Kalman and Neural Network Approaches for the Control of a VP bandwidth in an ATM Network

Raphaël Feraud [*], Fabrice Clérot [**], Jean-Louis Simon, Daniel Pallou, Cyril Labbé, Serge Martin

France Télécom C.N.E.T., Technopole Anticipa, 2 av. Pierre Marzin, BP 40
22307 Lannion Cedex France

Abstract. Allocating resources to data traffic in telecommunication networks is a difficult problem because of the complex dynamics exhibited by this kind of traffic and because of the difficult trade-off between the delivered quality of service and the wasted bandwidth.
We describe and compare the performances of two controllers of different designs (a Kalman filter and a neural network) to adaptatively control the bandwidth of a VP connecting two network nodes so as to keep the quality of service close to a given target.
Simulations are carried out on a hardware emulator which allows a fast and faithful simulation of the switch functionalities.
The Kalman filter obtains good performances and the original (and specific to this problem and its implementation constraints) neural network training strategy allows to obtain a faster and more accurate control.

1 Introduction

Managing data traffic in multiservice networks is a significant challenge for network operators since the data traffic volume is already equivalent to the classical telephony volume and is increasing at a much faster rate and it is known that data traffic is not well represented by classical traffic models, so that standard dimensioning techniques fail [4].

More precisely, the study of real data traffic traces [9], [5], [10] has shown that such traffic either is stationnary and should be modelled by processes with long range dependence and complex short time-scale dynamics [5] [10], or is not stationary and should be locally modeled by short range processes whose parameters have to be tracked on-line [9].

Dimensioning the network resources is a significant challenge in both cases: if the traffic is non-stationary, on-line tracking and prediction is necessary so as to allocate the resources; in the other case, static dimensioning from long range dependent models often leads to the allocation of very large resources protecting the network from very large deviations; such large deviations are

[*] Email: raphael.feraud@cnet.francetelecom.fr
[**] Email: fabrice.clerot@cnet.francetelecom.fr

however very rare and such static dimensioning wastes a lot of resources [3]. This is why, even assuming that traffic is stationnary and long-range dependent, on-line estimation of the parameters and prediction of the resources needed, that is adaptive dimensioning, seems a more promising approach.

In this work, we shall investigate two approaches for controller design; in the first approach, our a priori knowledge of the system is used to *select state variables* which sum up the state of the system as it may be observed (measured) and the controller design problem reduces to learn the relationship ("mapping") between the state variables and the behaviour of the system. This approach is implemented by a neural network. In the second approach, our a priori knowledge of the system is used to *build a model* of this system depending on a few parameters and the controller design problem reduces to fit the model to the behaviour of the system. This approach is implemented by a Kalman filter.

Since this work is strongly constrained by implementation considerations, we shall first present in some details the mechanisms used to guarantee a minimum QoS and fairness in the network nodes. The design of the controllers is then described and the experimental environment and results are discussed.

2 The CMS switch

The CMS switch is an ATM switch developed at CNET with the aim of optimizing transfer of high speed data in an ATM network. It takes into account the specificity of data service: large semantic constraints and weak temporal constraints. A single connection-oriented data service is implemented in the backbone. In each switch, connections are isolated by the use of a per connection FIFO, and active connections share dynamically the bandwidth. Each connection is identified by its Virtual Connection (VC), its Minimum Guaranteed Rate (MCR), its Peak Cell Rate (PCR). The connections are multiplexed in a Virtual Path (VP) between two CMS nodes.

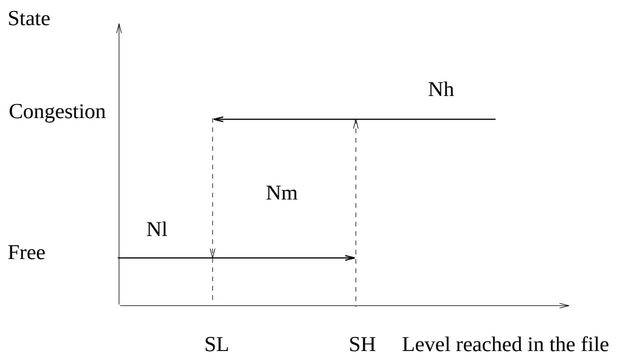


Fig. 1. The free/congested cycle. See text for explanations.

A hop by hop flow control is implemented between the transit nodes, ensuring the absence of packet losses in transit. The implementation of this hop by hop flow control relies on the watermarking of each individual queue. Two watermarks (high and low marks) are defined for each queue, depending on the parameters described below (section 3). The flow control operates as described on Figure 1:

1. free regime: starting from an empty queue, the connection is in its free regime, as long as the queue length does not reach the high mark SH. The rate allocated to this source by the upstream node is only constrained by the upstream node fair queuing algorithm (see below);
2. congested regime: when the queue reaches the high mark SH, a RM cell is sent upstream requesting the fair rate for this connection to be set at its minimun guaranteed rate, MCR_i, by the upstream node. The connection stays in the congested regime as long as its queue length stays above the low mark, SL. When the queue reaches the low mark SL, a RM cell is sent upstream allowing this connection to go back to its free regime.

The VP (Virtual Path) bandwidth is allocated to the active sources by a Weighted Round Robin mechanism, which guarantees that each active source is served at least at its minimum guaranteed rate. The bandwidth in excess of the sum of the minimum guaranteed rates is then allocated fairly among the active sources, according to their declared peak cell rates. The fair rates locally allocated to the sources depend on their free or congested state as notified by the downstream node:

- if the source is in the congested mode, its fair rate is its minimum guaranteed rate: $R_i = MCR_i$
- if the source is in the free mode, its fair rate is given by

$$R_i = MCR_i + \frac{PCR_i}{\sum_{j*} PCR_{j*}}(VP - \sum_j MCR_j)$$

where $\sum_j *$ means a summation on the free sources only.

Too large an excess bandwidth will lead to a low VP bandwidth utilization without any significant enhancement of the Quality of Service (QoS). Our purpose is to adapt periodically the VP bandwidth on each ATM node in order to maximize the VP bandwith utilization without degrading the Quality of Service. The QoS indicator is chosen as the ratio of congested sources. This ratio is set to 20 %. The adaptation period is set to 5 seconds.

3 Definition of the state variables

As explained above, the neural network predictor requires a description of the state of the system ("state variables") to compute its prediction. In addition to the classical trade-off between the accuracy of the description and the complexity of the training, we had to take into account that

- the limited communication bandwidth left to collect the state variables in the interface cards and transmit them to the processing unit did not allow to collect state variables on a queue by queue basis with a sampling time compatible with our objectives of fast control of the VP bandwidth because of the possibly large number of VCs (up to 1024 VCs per VP);
- the limited processing power and storage space available on the interface cards did not allow to pre-process the raw queue data locally.

We had to rely on a sampling of the average state of the queues. Such a description requires a communication bandwidth independent of the number of VCs and does not require any processing power on the interface cards. The system is described by the ratio of the number of files found in a given state at sampling time to the total number of files: N_l, the ratio of files under the first mark, N_m, the ratio of files between the first and the second mark, N_h, the ratio of files upper the second mark and N_{cl}, the ratio of sources locally congested (i.e. sources for which a congestion notification has been sent to the upstream node).

The description of the system, $D_t = (N_l(t), N_m(t), N_h(t), N_{cl}(t))$, is transmitted to the processing unit at sampling time. Storage and pre-processing are done in the processing unit. Pre-processing consists of an averaging on jumping windows of size T: $E_t = \frac{1}{T}\sum_{t-T+1}^{t} D_u$. These averages are the state variables used by the neural network. In the following, the sampling period is 1 millisecond and the averaging period is 1 second.

4 A simple controller

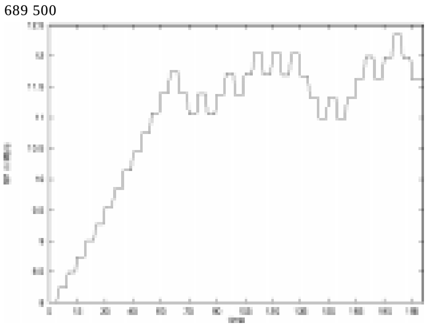
Since the Connection Admission Control (CAC) and the Usage Parameter Control (UPC) guarantee the conformity of a VC to its traffic contract, the connection blocking probability (N_{cl}) can be increased without affecting the Quality of Service (QoS). Therefore, the method to control the VP bandwidth can be based on a connection blocking probability target (ON_{cl}) [8]:

$$VP(t+1) = \begin{cases} (1+r)VP(t) & if \quad N_cl > ON_{cl} \\ (1-r)VP(t) & if \quad N_cl \leq ON_{cl} \end{cases}$$

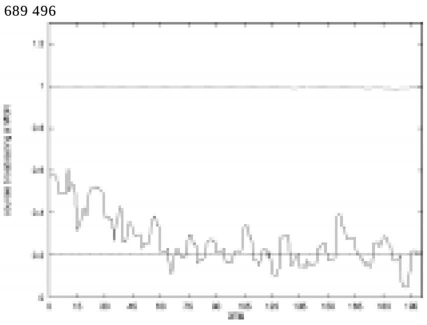
The drawback of this step by step algorithm is the tuning of the parameter r. For a large value of r, the algorithm converges quickly with strong oscillations. For a small value of r, the convergence is slow. An improvement of this algorithm is to calculate r according to the value of N_{cl} [7]: $r = \max(\gamma, \frac{N_{cl} - ON_{cl}}{1 - N_{cl}})$, where γ is an a priori upper bound for the increment r.

The convergence of this algorithm is slow and it has only been proposed for adaptation periods of the order of the hour [8, 7].

Data traffic cannot be considered stationary on such a long period; indeed, the study of real traffic traces has shown that the local stationarity hypothesis holds on time scales of the order of a second only [9]. The adaptation of the VP bandwidth must cope with non-stationarities and therefore must converge on a time scale of a few seconds.



(a) Value of the VP bandwidth according to time



(b) Value of the ratio of congested files according to time

Fig. 2. Results of the step by step algorithm for $ON_{cl} = 0.2$.

5 A neural network controller

5.1 Learning a function

To control the VP bandwidth, a function using the state variables, E, as inputs and the VP bandwidth, VP, as output is needed (Figure 4). The regression consists to find a function $f_\alpha \in \Gamma$ minimizing the functional risk $R(f_\alpha) = \int (f_\alpha(E) - VP)^2 P(VP|E)dE$.

In learning case, the conditional distribution of probability $P(VP|E)$ is unknown and we approximate the functional risk by the empirical functional risk, using a set of examples $R(f_\alpha) = \frac{1}{N}\sum_i^N (f_\alpha(E_i) - VP_i)^2$, where N is the number of examples.

The learning machine defines the space of functions Γ. Neural networks (Figure 3) are powerful learning machines since they are non-linear models with efficient and robust training techniques.

5.2 Learning the optimal value of the VP bandwidth

Let F be the control function we want to learn:

$$VP^*(t+H) = \max((1 + F(E_{t-4T}, E_{t-3T}, E_{t-2T}, E_{t-T}, E_t))VP(t), \sum_i MGR_i)$$

where H is the period of control ($H = 5$ seconds in this study), E_t the averaged state variable at time t, T an averaging period ($T = 1$ second) and MGR_i the minimum guaranteed rate of the source i. To capture the temporal correlations, the inputs consists of a series of five consecutive values of the state variables: $E_{t-4T}, E_{t-3T}, E_{t-2T}, E_{t-T}, E_t$.

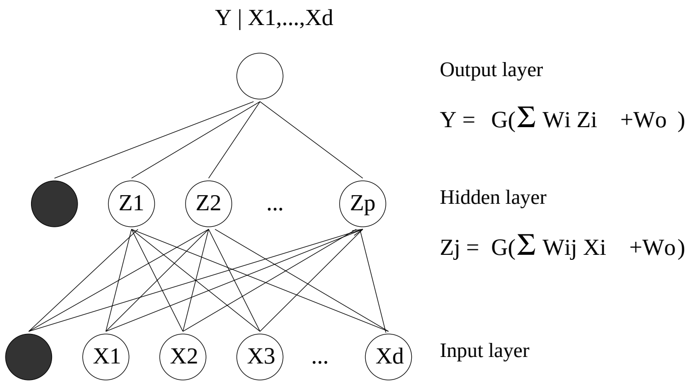


Fig. 3. The multilayer perceptron with one hidden layer allows to approximate non-linear functions. $G(x) = \frac{1}{exp(-kx)+1}$ is the sigmoid function.

Note that the output of the controller cannot be lesser than the sum of minimum guaranteed rates so as to fulfill the traffic contract.

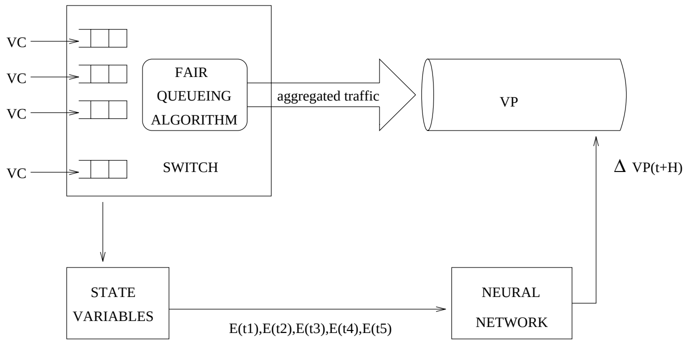
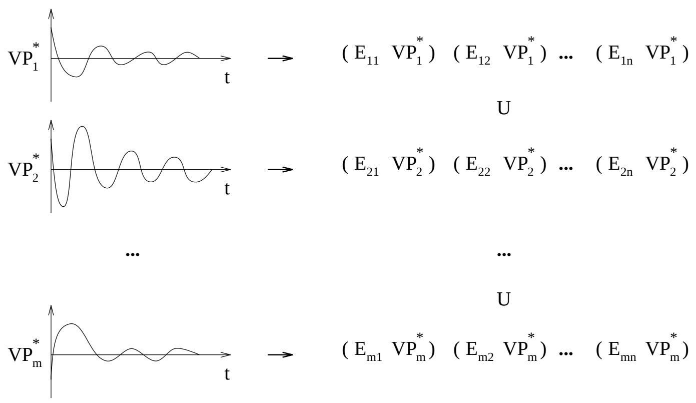


Fig. 4. Schema of the control process. The series of state variables E_t are the inputs of a neural network. The neural network returns the variation of the VP bandwidth needed to reach the optimal VP bandwidth fot the next period H.

To obtain a fast control, we use the step by step algorithm previously described to generate data along a trajectory (Figure 2). Each trajectory generates several input examples with the same output: the optimal VP bandwidth (Figure 5). If such a function can be learned, it leads to a very fast convergence of the control process: anywhere on the trajectory, the function will return the optimal VP bandwidth. To obtain various outputs, we generate several trajectories, corresponding to different traffic contracts, source rates and initial VP bandwidths. The set of examples consists of 13000 examples. Each example consists of an input/output couple:

$$(E_{t-4T}, E_{t-3T}, E_{t-2T}, E_{t-T}, E_t; \Delta VP(t))$$



VP^*_1
t
(E_{11} VP^*_1) (E_{12} VP^*_1) ... (E_{1n} VP^*_1)
U
VP^*_2
t
(E_{21} VP^*_2) (E_{22} VP^*_2) ... (E_{2n} VP^*_2)
...
...
U
VP^*_m
t
(E_{m1} VP^*_m) (E_{m2} VP^*_m) ... (E_{mn} VP^*_m)

Fig. 5. The value of the optimal VP bandwidth is obtained at the end of the convergence of the step by step algorithm. For a trajectory i, at each control period j the state variables E_{ij} are collected to generate n input examples with the same output: VP^*. The use of m trajectories allows to generate $N = mn$ different input/output couples.

We define the optimal value of the VP bandwidth, VP^*, according to the target congested rate ON_{cl}: $VP^* = \{VP$ such that: $N_{cl} = ON_{cl}\}$, as obtained at the end of the convergence of the step by step algorithm.

Finally, to minimize the functional risk between the learning machine and the target function F, we use:

- a multilayer perceptron as learning machine consisting of an input layer of twenty neurons corresponding to the series of state variables (five state variables, one state variable being a vector with four components), a hidden layer of five neurons and one output neuron,
- the standard back-propagation algorithm [6] to minimize the empirical functional risk on a training set of 10000 examples,
- a validation set to control the minimization of the functional risk: the learning is stopped when the empirical functional risk increases in the validation set composed by 3000 examples (examples which are not used to minimize the empirical functional risk).

6 A Kalman controller

To build the Kalman filter, we consider a simple model of the ATM node (Figure 6). The incoming traffic is aggregated. It is divided by the ATM node in two parts: a part of the incoming traffic passing through the VP and a blocked and lost part. Under the assumption that all the sources send at least at their minimum guaranteed rate, we can write $x = N_{cl}x + y + \sum_i MGR_i$.

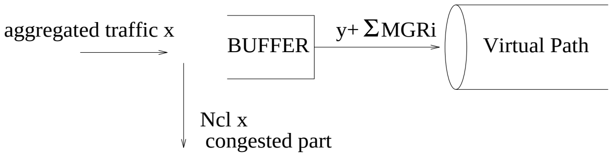


Fig. 6. The incoming aggregated traffic is divided in two parts. The first part, y and the sum of the minimum guaranteed rates, passes through the VP and the second part, the congested part $N_{cl}x$ is blocked and lost.

Let V be the part of the VP bandwidth above the sum of the MGR: $V = VP - \sum_i MGR_i$.

We assume that the VP bandwidth utilization is high: $y \approx V$. Assuming Gaussian and centered noises, the following non-linear model defines the extended Kalman filter:

$$N_{cl} + \epsilon_{N_{cl}} = 1 - \frac{V}{x + \epsilon_x}$$

Where the incoming aggregated bandwidth x is the state variable, the excess bandwidth V the control variable, the ratio of congested sources N_{cl} the measured variable, and $\epsilon_{N_{cl}}$ and ϵ_x are respectively the measurement and the process noises. At each period H, the rate of congested sources $N_{cl}(t)$ is measured, the noises $(\epsilon_x, \epsilon_{N_{cl}})$ and the state variable $x(t)$ are estimated by the Kalman filter . Then, the optimal VP bandwidth is given by: $VP^*(t + H) = \hat{x}_t(1 - ON_{cl}) + \sum_i MGR_i$ where $\hat{x}_t$ is the estimated state of the system by the Kalman filter and ON_{cl} the target congestion rate.

The initial value of $\epsilon_{N_{cl}}$ is set according to the behaviour of N_{cl} after convergence of the step by step algorithm and the initial value of ϵ_x is set according to the traffic contracts, assuming that the instantaneous rate of each source is drawn independently and uniformly between MGR_i and PCR_i.

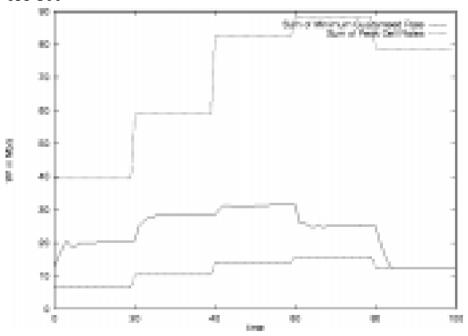
7 Experimental results

7.1 Simulation environment

Traffic and queue lengths measurements required functionalities which could not be implemented in the current version of the CMS switch and we had to rely on simulations. In order to measure the state variables, we had to simulate all the details of the implementation of the CMS switch as described above (per VC queuing, bandwidth sharing algorithm, flow control algorithm, see Section 2). A software-based simulator could not have allowed the implementation of our training strategy in reasonable time; instead, we have developped a simulator of the CMS switch itself and mapped it on a hardware architecture.

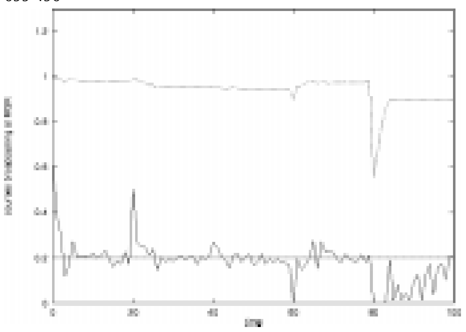
With tools currently used in integrated circuit development such as hardware emulators, it is possible to reproduce a reliable image of the CMS switch by saving only useful functions for our implementation. Advantages of the hardware approah are [2] the simulation speed, close to the real switch, and the great number of VC that can be instanciated, leading to realistic traffic loads without slowing down simulation.

699 500



(a) Optimal VP bandwidth according to time.

699 496



(b) Ratio of congested sources according to time, at the top the VP bandwidth utilization curve.

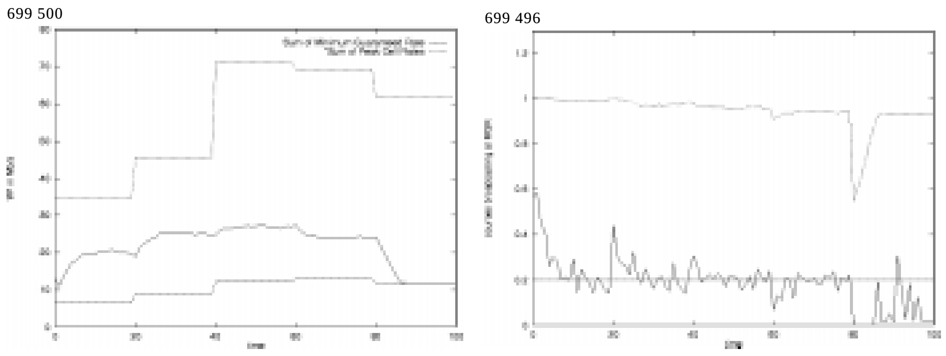
Fig. 7. Results of the neural network controller for $ON_{cl} = 0.2$. Every minute (20 time units) all the traffic contracts and rates of sources are randomly reset. On the last period, the optimal VP bandwidth is lower than the sum of minimum guaranteed rate.

Indeed, with a reconfigurable hardware machine, speed is governed by the clock frequency from 1 to 10 MHz and this speed does not depend on the model complexity as for a pure software model. This allowed us to instanciate many sources simulating various VC. This hardware implementation can be achieved using usual tools in CAD design: a VHDL (VHSIC Hardware Description Language) description of the chip is used to describe the system in terms of concurrent processes, then a VHDL synthesiser translates the VHDL description into combinational logic and registers, the basic elements used in electronic systems, and finally, a compiler computes the link to map this hardware description on the sea of gates of the emulator. The use of an emulator is original in this approach. This machine is composed of a sea of gates that can be dynamically linked to reproduce an electronic design. The M500 emulator from Metasystems comprises 500,000 programmable logic gates, connectable to each other, 17 Mbytes of memory, single or double port, an adjustable clock frequency from 1 to 10 MHz. The M500 emulator and the associated software has important features for debugging: it stores signal values of the last 7000 clock cycles, allows the user to start, stop, and run clock step by clock step simulation as desired. Reconfigurable hardware machines are therefore very appealing simulation tools for complex systems such as the dynamical coupling between the bandwidth sharing and flow control as implemented in the CMS node.

The simulator comprises three main blocks [1]:

- Traffic sources, most widely used statistical models are implemented: periodic sources with a period modified regularly, uniform traffic and geometric process, Markov modulated Bernouilli process with on-off sources, which simulated bursty traffic.
- Switch model: the CMS model can receive up to 64 sources or VC to fit in the hardware emulator.
- Traffic analysis: several quantities (such as cell number, lost cell number, inter-arrival process, queue occupancy, dates when SL and SH marks are reached) can be stored "on the fly" to evaluate the state variables.

This simulator has been used for the generation of the training and validation sets and for the test experiments presented below.



(a) Value of the VP bandwidth according to time.

(b) Value of the ratio of congested files according to time, at the top the VP bandwidth utilization curve.

Fig. 8. Results of the Kalman controller for $ON_{cl} = 0.2$. Every minute (20 time units) all the traffic contracts and rates of sources are randomly reset.

7.2 Discussion

Corresponding to the implementation constraints, the prediction period H is 5 seconds the sampling period of the state variables is 1 ms and the target ratio of congested sources is $ON_{cl} = 0.2$. To evaluate and to compare the control of the Kalman filter and of the neural network filter, we generated 64 various periodic traffics on a period of one minute. After this minute, we introduced an important non-stationarity by randomly resetting the traffic contracts and rates of all the sources. The emission rates and traffic contracts of the sources were drawn from Gaussian distributions (with the constraint that the emission rate cannot exceed the declared peak cell rate; note that sources can emit at a lower rate than their minimum guaranteed rate). We introduced a increasing trend of the means of Gaussians on a first period and a decreasing trend on a second period.

The neural network experiment and the Kalman filter experiment were built using the same methods but with slightly different values of the mean of Gaussians generating the parameters of sources. The quality and stability of the convergence can be compared on the stationarity period and the speed of convergence on the non-stationarity period.

First, the purpose is to maximize the VP bandwidth utilization. In both cases, it is close to 1 (Figure 7b, 8b). The analysis of the curves shows that the control of the Kalman filter (Figure 8a) is slower than the one of neural network (Figure 7a): a stable value of the VP bandwidth is reached on one or two iterations (5 to 10 seconds) in the case of neural network and two or three iterations for the Kalman filter. Moreover, the quality and stability of the neural network controller (Figure 7b) is better than the Kalman controller (Figure 8b): the oscillations of the measured congested rate have lower amplitudes for the neural network controller than for the Kalman controller.

It may be argued that the switch model designed for the Kalman filter is grossly simplified as it implements almost nothing of the complex queue management, fair service and congestion control described above in Section 2. The problem is that modeling such a complex system in greater details leads very fast to untractable models.

This illustrates the difference between the two kinds of controllers when dealing with complex systems: the “mapping approach” (neural network controller in this work) allows us to describe the system in great details (the hardware simulator implements all the complexities of the CMS switch, so that the state variables describe the behaviour of true switch), to the expense of finding a good mapping between high dimensional spaces by an off-line training, whereas the “modeling approach” (Kalman controller in this work) forces us to use simplified models but allows us to adapt the model parameters on-line.

Training of the neural network is performed off-line and gives superior performances, but it must be emphasized that the performances will stay good only as long as the examples used for the off- line training will be representative of the behaviour of the system. This implicit stationarity hypothesis is certainly wrong in the long term and appropriate strategies (such as periodic re- training) have to be developped to cope with non-stationarities.

The modeling approach has good although slightly inferior performances but these performances will stay good as long as the model is representative of the behaviour of the system and non- stationarities are coped with by the adaptivity of the Kalman filter.

8 Conclusion

We have described, implemented and compared the performances of two controllers of very different designs (a Kalman filter and a neural network) so as to adapatively dimension the VP bandwidth between two nodes so as to keep the quality of service close to a target value.

Simulations were made possible by the use of a hardware emulator allowing to reproduce faithfully the complex behaviour of the switch.

Both controllers showed good performances with a faster and more accurate control by the neural network. We have discussed how this improved performance came at the expense of adaptivity to long-term non-stationarities.

Both controllers are currently being implemented in a prototype of the CMS switch so as to test their performances under real workload.

References

1. C. Labbé, V. Olive, and J.M. Vincent. Emulation on a versatile architecture for discrete time queuing network : Application to high speed network. In *ICT98, International Conference on Telecommunications*, 1998.
2. C. Labbé, F. Reblewski, and J.M. Vincent. Performance evaluation of high speed network protocols by emulation on a versatile architecture. *Recherche Opérationnelle/Operations Research*, 32(3), 1998.
3. I. Norros. On the use of fractional brownian motion in the theory of connectionless networks. *IEEE Journal on Selected Areas in Communications*, 13(6), 1995.
4. V. Paxson and S. Floyd. Wide aera traffic: the failure of poisson modelling. In *Sigcomm'94, Computer Communication Review*, volume 24, pages 257–268, 1994.
5. R. Riedi and Willinger W. Towards an improved understanding of network traffic dynamics. In *Self-similar Network Traffic and Performance Evaluation*, Wiley, 1999.
6. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distribued Processing: Explorations in the Microstructures of Cognition*, volume 1, 1986.
7. C. Shioda, H. Toyoizumi, H. Yokoi, T. Tsuchiya, and H. Saito. Self-sizing network: a new network concept based on autonomous vp bandwidth adjustement. In *ITC 15*, 1997.
8. S. Shioda. Evaluationg the performance of virtual path bandwidth control in atm networks. *IEICE Trans. Commun.*, E77-B(10):1175–1187, 1994.
9. S. Vaton, E. Moulines, and H. Korezlioglu. Statistical identification of lan traffic data. In *5th IFIP Workshop on Performance Modelling Evaluation of ATM Networks*, 1997.
10. W. Willinger, V. Paxson, and M.S. Taqqu. *"A Practical Guide to Heavy Tails : Statistical Techniques for Analysing Heavy Tailed Distributions.* Birkhauser Verlag Boston, 1998.