# Dynamical Resource Reservation Scheme in an ATM Network Using Neural Network-Based Traffic Prediction

Fabrice Clérot    Pascal Gouzien    Samy Bengio*    Annie Gravey
Daniel Collobert†

France Télécom BD-CNET, 2 avenue Pierre Marzin, 22307 Lannion Cedex, FRANCE
{bengios}@cirano.umontreal.ca
{clerot,gouzien,graveya,collober}@lannion.cnet.fr

## Abstract

*Using real traffic data, we show that neural network-based prediction techniques can be used to predict the queuing behaviour of highly bursty traffics typical of LAN interconnection in a way accurate enough so as to allow dynamical renegotiation of a DBR traffic contract at the edge of an ATM network.*

*The performances of predictor-based in service renegotiation are evaluated in terms of renegotiation errors and reserved bandwidth for the the DBR traffic handling capability and are shown to be very encouraging for the use of connectionist prediction techniques for the management of bursty traffics in ATM networks.*

**Keywords**: *neural networks, traffic prediction, leaky bucket, LAN interconnection, ATM networks.*

## 1 Introduction

In order to realize its promises as the B-ISDN transfer mode, ATM has to fulfill two conflicting requirements, namely *"Bandwidth on Demand"* and *"Guaranteed Quality of Service (QoS)"*, for various types of traffic. This is particularly challenging in the case of variable bit rate ($VBR$) traffics, such as compressed video or LAN interconnection, where the behaviour of the sources is not well-defined in terms of bandwidth requirements.

In order to fulfill the *"Guaranteed QoS"* requirement, traffics should not be allowed to access the network without control, and such a control (traffic policing) is specified in terms of continuous state *leaky buckets* (also known as generic cell rate algorithm or virtual scheduling algorithm) at the network edges [1, 11]. This implementation supposes a traffic contract between the source and the network which defines the behaviour of the source in terms of mean cell inter-arrival time and cell delay variation tolerance. The enforcement of this traffic contract at the User-Network Interface (UNI) protects the network against bursts of uncontrolled length and intensity and such a traffic characterization allows to reserve necessary resources inside the network so as to guarantee the required QoS. Various schemes can be used to reserve those necessary resources and one of them, namely the Deterministic Bit Rate (DBR) traffic handling capability, will be studied below. In the following, we shall be concerned with a restrictive definition of the quality of service in terms of cell loss mainly as we only address the problem of data traffic.

The *"Bandwidth on Demand"* requirement can then be implemented by renegotiating (periodically or upon request from the source) the traffic contract and using, for instance, a *Fast*

*now at INRS-telecommunications, 16 place du Commerce, Ile-des-Soeurs (QC) CANADA, H3E 1H6.
†whom correspondance should be addressed

*Reservation Protocol (FRP)* [2]. However fast they can be, resource reservation protocols cannot be based on the instantaneous characteristics of the traffic to be carried: reservation of the resources involves a latency of the order of the network round trip time at least and, moreover, the operation of these protocols should not overload the network in terms of processing time. This points out the need for the source to be able to efficiently predict its traffic descriptor over a typical inter-negotiation period.

Although this access control scheme based on both resource reservation and enforcement of the declared traffic descriptors allows an efficient use of the network resources, it may be quite difficult to implement from a source point of view, specially in the case of very bursty traffics as is the case for LAN interconnection: such bursty sources cannot efficiently negotiate their traffic contract for the next period without being able to accurately predict their own behaviour during this period. Such a prediction capability is indeed an essential requirement to the realization of ATM promises.

Although predicting traffic with neural networks has been advocated for compressed video [5], we are not aware of such a study for data traffics or for the time-scales considered below. In this contribution, we shall show, *using real bursty traffic data*, that such a prediction of the queuing behaviour of such traffics is indeed possible with neural networks.

This may seem in disagreement with the conclusions of recent studies of LAN and WAN traffic which have evidenced the wide intensity variations and long term correlations existing in such traffics [14, 16]. It should be recalled that we are not in any way trying to predict the behaviour of the traffic itself, but we rather try to predict the extreme behaviour of a queue driven by the traffic so as to define an appropriate traffic descriptor in the ATM framework for the next period. In this respect, while leaving the question of modeling data traffic open, this study aims at giving a pragmatic answer to the problem of "fitting" such traffic into the rather rigid requirements of traffic policing at the edge of an ATM network.

The framework of this study is summed up in Figure 1: a pair of LANs is interconnected through a pair of VCs inside an ATM WAN; note that *individual sources belonging to a LAN are multiplexed at the VC level*. The prediction function is implemented on this multiplexed trafic, at the ingress of the ATM-WAN only (on the LAN side of the UNI) and is used to periodically renegotiate the usage parameters of the outgoing VC with the CAC (Call Acceptance Control). The conformance of the traffic to the negotiated usage parameters is enforced on the WAN side of the UNI by the UPC (Usage Parameter Control).

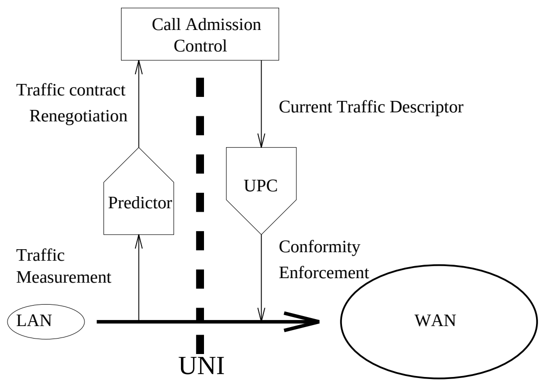


Figure 1: Framework of this study: the LAN traffic is multiplexed on a single VC and the prediction function is implemented at the ingress of the ATM-WAN

In this study we shall not address the problem of the influence of rejected renegotiations at the CAC level (i.e. we assume that predicted usage parameters are always accepted by the CAC) and confine ourselves to the prediction problem.

We note here that the usefulness of traffic prediction is not restricted at the UNI as described above; as a matter of fact, renegotiation of resources, either using signalling protocols or in band reservation schemes is also performed at other interfaces (typically NNIs), so that traffic prediction, if indeed efficient, could be implemented ubiquitously in ATM networks.

The paper is organized as follows: after a presentation of the DBR traffic handling capability, we shall shortly discuss the possible benefits of periodically renegotiating the resources needed in the case of a bursty traffic; we shall then present the connectionist models for time series prediction, describe our predictor implementation and discuss the results.

# 2 Ressource Allocation Overview

Among the various ways of allocating resources in an ATM network while protecting the QoS defined by the ITU-T (DBR, SBR, ABR, ABT, see [15, 11] for more details), we shall concentrate on DBR (Deterministic Bit Rate) and on the implementation of in-service parameter renegotiation for this capability.

## 2.1 Description of the DBR Capability

Hereafter, we briefly describe the DBR ATM layer traffic handling capability as currently standardized [11], that is without in-service renegotiation of the parameters.

For this capability, the source simply declares a peak cell rate ($PCR$) and a cell delay variation tolerance ($\tau_{pcr}$) for the duration of the call, and reservation will be attempted on the basis of $PCR$.

The algorithmic definition of the peak cell rate is related to a virtual queue: the actual rate of a source is considered to be below the negotiated $PCR$ as long as the buffer level of a (virtual) queue that is emptied at $PCR$ is below a threshold $L_{max}$ which is related to the negotiated cell delay variation ($CDV$) tolerance $\tau_{pcr}$ by

$$L_{max} = PCR \times \tau_{pcr}$$

This definition is summed up in Figure 2.

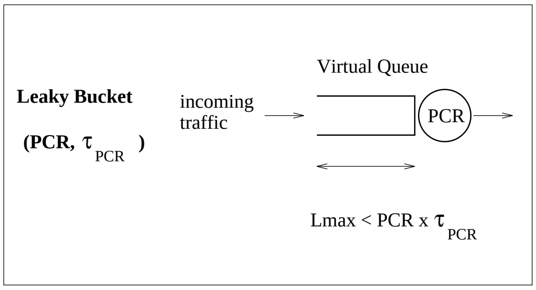


Figure 2: Definition of the DBR traffic descriptor parameters

The algorithm used at the UPC so as to enforce the conformity of the incoming traffic to the traffic descriptor is known as the generic cell rate algorithm (GCRA).

## 2.2 In-Service Renegotiation

Obviously, it may be quite difficult to set the parameters defined above for the duration of the call, specially in the case of data traffic.

In this contribution, we propose to take advantage of the prediction ability of neural networks to renegotiate these parameters during the call (in-service renegotiation), so as to follow more closely the needs of the traffic. It should be noted that this renegotiation will not be performed in band, as in the ABT-IT capability, but will involve signalling automata as being currently standardized at the ITU-T [17].

We shall only consider periodic renegotiation of the parameters: instead of negotiating the parameter $PCR$ for the duration of the call, negotiation will be carried for the next period under the assumption that $\tau_{pcr}$ is fixed.

Hereafter, we shall use for the peak cell rate the *minimum* value satisfying the conditions imposed by the GCRA, hence requiring the maximum precision from our predictor. In a real situation, some kind of safety margin might be allowed of course but, even under this most stringent requirement, we shall see that our predictor behaves very well since the notion of safety margin can be included in the construction of the predictor.

We present below the traffic trace which has been used for this study.

## 3 Description of the Traffic Traces

As explained below, in order to get reliable results about the prediction capabilities of neural networks, it is necessary to use large real traces. The traces we have used are made TCP traffic recorded at the Berkeley and CNET Lannion gateways to the Internet. The traces are recorded on a packet per packet basis, each packet being characterized by its arrival time and the amount of TCP data transferred.

One should be careful when using traffic traces recorded on existing networks for studies of mechanisms to be implemented in future networks: obviously, using real traffic traces to design and test new congestion management mechanisms for instance may be misleading since the characteristics of the trace itself can be strongly dependent on already existing protocols (TCP in our trace, for instance). The present situation is different: the trace we use certainly includes inter-network TCP dynamics but as the application we are aiming at is mainly private networks interconnection by ATM links this is not a drawback since traffic originating from such networks (which often are inter-networks themselves) will also contain such dynamics, TCP/IP being likely to stay as the main protocol stack for the next future in the area of data communications.

The Berkeley trace, hereafter referred as the LBL-PKT3 trace, has been thoroughly studied by other groups [16, 20] and has been shown to exhibit a very high variability (the average rate of the trace is 0.35 Mb/s with peak rates up to 1.7 Mb/s even when the rates are averaged on a time window as large as 10 s) and strong long-range correlations or non-stationary behaviour (see [20] for a discussion of this issue).

The traces recorded at the CNET Lannion gateway exhibit similar characteristics and consist of 12 hours of TCP traffic.

Such traces should be representative of data traffic which ATM shall have to carry so as to support virtual private networks and "wide area LANs".

As intuitive from Figure 3, the resources needed by the traffic wildly vary in time (even when averaged on a 10 s time scale), indicating potential resource savings *if such variations can be predicted.* We shall now turn to connectionist models for time series prediction.

## 4 Connectionist Models for Time Series Prediction

Let a given one-variable time series be represented by the $N$ values $\{x_1, x_2, \cdots, x_N\}$. Prediction then consists to find the future values $\{x_{N+1}, x_{N+2}, \cdots\}$. Takens [19] has shown that if the series is obtained from a deterministic dynamical system, there exists a scalar $d$ (which is called the *embedding dimension*), a scalar $\tau$ (which is an arbitrary delay) and a function $f$ such that for every $t > d \cdot \tau$:

$$x_t = f(x_{t-\tau}, x_{t-2\tau}, \cdots, x_{t-d\tau}) \tag{1}$$

The prediction problem consists, given the first $N$ values of a time series, to find the appropriate $d$, $\tau$ and $f$. Of course one usually cannot be sure that a given series is deterministic. Actually, statistical methods do exist to verify if a series is deterministic and to find $d$ as well as $\tau$ but they require the size of the series to be on the order of $10^d$ which is rarely the case in

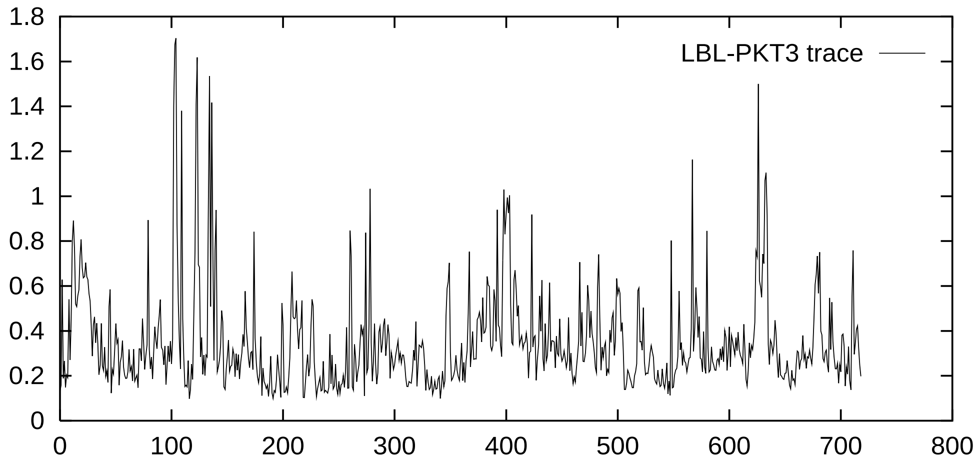


Figure 3: Evolution of the LBL-PKT3 trace. Each dot represents the mean input rate (in Mb/s) during a period of 10 s. The whole trace last 2 hours and has a mean input rate of 0.351Mb/s.

practical problems. For the moment, let us assume that we know $d$ and $\tau$ and that we want to find $f$. This is where *neural networks* come in: it is a well known fact that they can be used as universal function approximators [10].

The time series is cut into three non-overlapping sets: a *training set*, a *validation set* and a *test set*. The training set is used to find the weights of the neural network by minimizing a cost function using an iterative learning algorithm such as the backpropagation algorithm [18], the validation set is used to monitor the learning process (by cross-validation) and the test set is used to verify the real prediction performance of the network (that is, an estimated prediction error on future time series values).

In prediction problems, we train the network with *past examples* (thus, we minimize a *training error*) but we really want our network to perform well on *future examples* (thus, have a minimal *generalization error*). We use the validation set to estimate generalization error (note that the data in the validation set are not used to minimize the cost: minimization is only performed for the data in the training set). Training is stopped when the generalization error estimated on the validation set starts to increase (even if the training error is still decreasing), indicating that the training process begins to over-fit the training set.

The best heuristics used to select $\tau$ are based on the hypothesis that two successive values of the input data vector must be the least related in order to maximize information. For instance, one can choose the first zero of the autocorrelation function, or the first minimum of the mutual information function. In both cases of course, $\tau$ must be as small as possible.

The neural networks used in this study are multilayer perceptrons with one hidden layer. The architecture of such multilayer perceptrons is defined by the number of neurons in the input layer (i.e. the embedding dimension of the data) and in the hidden layer.

Many heuristics exist to determine these architectural parameters, but this is still a hard problem. We also use cross-validation to select the neural network architecture.

## 5 Predicting the traffic descriptor for the next period

We wish to implement a prediction-based renegotiation of the DBR contract. We are thus looking for a mapping with the following inputs:

- the current queue size, the current bit rate,
- some kind of information characterizing the past traffic,

and which would give as output the $PCR$ consistent with a given $\tau_{pcr}$ (in this work, $\tau_{pcr}$ is fixed for the whole trace) and the future traffic on the next $H$ seconds.

This is not a simple prediction problem. In fact, the predictor should not only predict the future characterization of the traffic (this is the prediction part), but also deduce, for a given future traffic characterization and initial queue size, what would be the maximum queue size reached in the next period (this is the function approximation part). As it is known that neural networks are good for prediction **and** function approximation, they are good candidates to solve this problem.

The information which characterizes the past traffic and the learning strategies are key issues for this prediction problem. They are described below (see Section 8).

## 6 Framework of the experiments

The following parameters have been used in our simulations:

- leaky bucket dimensioning : $\tau_{pcr} = 0.1\,$s (which is consistent with the fact that data transmission are only lightly sensitive to delays);
- we chose a value of 10 s for the negotiation period. The various ATM layer traffic handling capabilities and signalling mechanisms being still under discussion inside the standardizing bodies, this figure, although reasonable, should only be considered as indicative. We note here that in a different context, a renegotiation period of 1s was estimated to allow as much as 40,000 calls [9]; therefore a value of 10 s should not stress the signalling mechanisms beyond their limits even for a large number of calls.

Hence, in this experiment, $PCR$ is predicted for the next 10 s period, and reservation is carried out on the basis of $PCR$ only. Hereafter, we refer to this experiment as DBR-10s.

We would like to stress here that, as we are trying to predict the behaviour of a constrained *extremum*, the problem is all the more difficult as the prediction horizon increases. Therefore, a 10 s horizon represents a significant challenge.

The performance of the prediction machine is compared to the performance of an "oracle" who perfectly knows the future for the next negotiation period: the oracle does not attempt any "prediction" but simply calculates the parameters from the data of the next 10 s; it is used to test the performance of the predictor, and its performance itself is also interesting since it shows what can be expected from optimal renegotiation when applied to a real bursty traffic.

We shall first use the oracle to show the benefits brought by renegotiation; then we shall present the performance of our predictor for DBR using various learning strategies.

## 7 Oracle Results

We first want to illustrate the importance of being able to dynamically negotiate the bandwidth in the case of a bursty traffic; Table 1 shows the resources in terms of buffer size needed if one aims, while not renegotiating the $PCR$, at getting the same performances than DBR-10s in terms of mean rate ( $R_{mean}$ fixed, i.e. standard DBR case). Also given are the rates needed to get the same performances in terms of mean queue length ($L_{mean}$ fixed) and maximum queue length ($L_{max}$ fixed).

Obviously, an *optimal* dynamical negotiation of the bandwidth allows to save resources. We shall show below that, *although not optimal*, prediction-based dynamical negotiation is indeed possible and also allows to save resources.

## 8 Results for DBR-10s Using the Neural Network Predictor

In this section we present our results for different learning strategies and characterizations of the past traffic.

| when | Resources needed using... | |
| --- | --- | --- |
| | DBR-10s | Standard DBR |
| $R_{mean} = 0.9$ Mb/s | $L_{max} = 0.4$ Mb | $L_{max} = 23.1$ Mb |
| $L_{mean} = 0.09$ Mb | $R_{mean} = 0.9$ Mb/s | $R_{mean} = 5.5$ Mb/s |
| $L_{max} = 0.4$ Mb | $R_{mean} = 0.9$ Mb/s | $R_{mean} = 3.7$ Mb/s |

Table 1: Comparisons between the use of DBR-10s with an oracle and standard DBR (no renegotiation).

Analysis of the time series characterizing the traffic lead us to choose $\tau = 1$ and, from cross-validation, we determined $d = 20$ but the precise value appeared not to be crucial (if large enough).

## 8.1 A first "heavyweight" experiment

For a first experiment, the characterization of the past traffic was chosen to be the traffic means and variances of the volume of data arriving in 0.1 s jumping windows, for the last 2 seconds.

Using LBL-PKT3, we thus generated 72000 points of a time series characterizing the traffic behaviour, which was cut into three equal and non-overlapping sets (training, validation and test). The test set corresponds to the last 40 minutes of the trace.

The learning strategy was the following: for each time frame of 10 seconds, we furthermore generated 9 fictive initial conditions (3 current queue sizes $\times$ 3 current bit rates), which were chosen around the initial conditions obtained by the oracle for this time frame. We then computed for each situation, given we knew the future of the trace, the minimum bit rate consistent with $\tau_{pcr}$ for the next time frame of 10 seconds. Hence, a sample is made of

- the current file length
- the current bit rate
- the 20 means and 20 variances characterizing the past traffic
- the target value of $PCR$ which is used for the training of the neural network.

This finally gave us a training set and a validation set of 216000 samples each.

The results of this learning strategy were reported and discussed in [7]. As shown on Figure 4, the reservation made by the neural network are consistent with the activity of the source, and the negotiated traffic contract is violated only once on the whole trace. See [7] for more details.

The main drawbacks of this approach are;

1. a very large learning set leading to very long trainings,
2. a difficult choice of the correct initial conditions to be generated: for the neural network used above, these initial conditions were chosen around the values obtained by the oracle, a choice which, *post facto*, did not appear so good since the NN-predictor tends to use systematically greater bit rates than the oracle (which is quite natural) and hence generates smaller queues, so that the system driven by the NN-predictor evolves in a part of the phase space significantly different from the part where it was taught (i.e. when the system is driven by the oracle),
3. a lack of "intuitive" control of the learning process: once the training and validation sets are generated, we have no control of what is happening.

The main conclusion of this expriment is that a "blind and heavyweight" approach to our problem is indeed effective; in the following we shall investigate learning strategies which avoid the above drawbacks, the main drawback being in our opnion the third one.

Inspecting the weights of the neural network, we also noticed that the variances we used to characterize the past traffic were given weights so small that they were virtually useless.

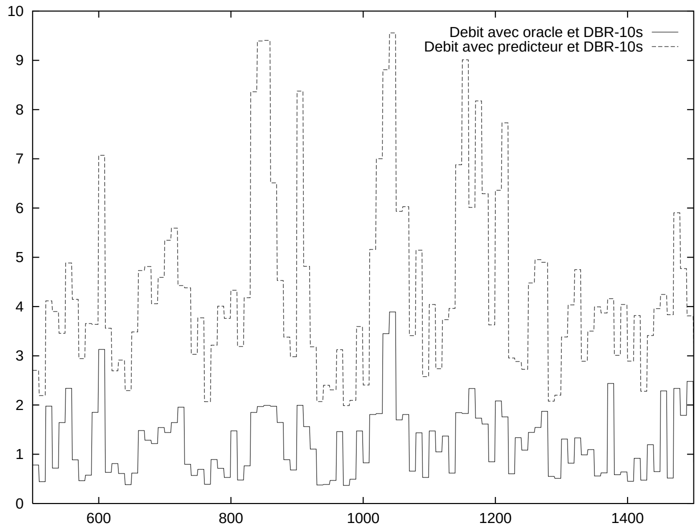


Figure 4: Results of the "heavyweight" learning strategy for the LBL-PKT3 trace.The solid line shows the bit rate when the oracle is used; the dotted line shows the bit rate when our predictor is used.

## 8.2 "Lightweight" learning strategies

### 8.2.1 Characterization of the past traffic

Keeping the same neural network architecture, we modified the characterization of the past traffic so that the input layer now receives:

- the current file length, the current bit rate
- the quantity of data of the last 2 s agregated in 100 ms windows (20 values)
- the quantity of data of the last 20 s agregated in 1 s windows (20 values)

It should be noticed that the characterization of the traffic we use does not require any fine-grained dynamical information (such as the interarrival times statistics for instance), but is only built of agregated quantity of data in fixed size windows. As the windows are indeed large, such a characterization should be implementable rather easily, without requiring accurate time-stamping.

### 8.2.2 Basic learning algorithm

For the three learning strategies described below, the learning algorithm is made of four steps: the training set is read sequentially and for each new renegotiation period (period $N+1$) we have

1. *a prediction step*

   calculate the bit rate predicted by the neural network, $D_{pred}$

2. *a trace-driven simulation step*

   - feed the trace for period $N+1$ in a file emptied at $D_{pred}$
   - calculate the maximum file length $L_{max}$ in period $N+1$

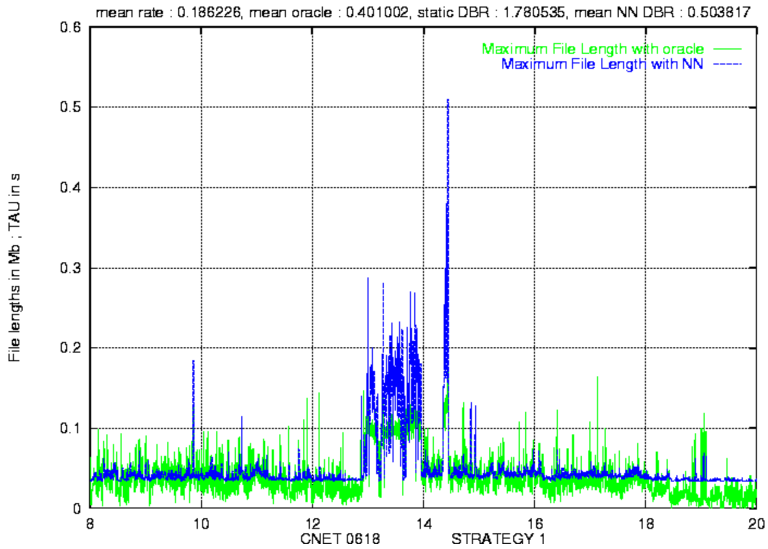


Figure 5: Results for Strategy 1 on a trace collected at CNET on June $18^{th}$ 1996. The training set is made of the first 2 hours, the validation set of the next 2 hours and the rest of the trace (8 hours) is used as the test set. The solid line is the maximum file length predicted by the oracle; the dashed line is the maximum file length predicted by the neural network.

- note that the initial conditions for period $N+2$ are $D_{pred}$ and the file length obtained from the trace-driven simulation at the end of period $N+1$

3. *an error evaluation step*
   - calculate the effective jitter tolerance $\tau_{eff} = \frac{L_{max}}{D_{pred}}$
   - note that $\tau_{eff} > \tau$ indicates a violation of the traffic contract

4. *a backpropagation step*

   we investigated three different possibilities for backpropagating the error $(\tau - \tau_{eff})^2$; they are detailled below

The main difference with the "heavyweight" learning strategy (Section 8.1) is that the initial conditions are now determined on the fly *from the dynamics of the system driven by the neural network*. Hence we can more efficiently explore the part of the state space spanned by the system driven by the neural network, which should lead to faster training times.

#### 8.2.3 Learning strategy 1: a simple-minded approach

As is usually done, we backpropagate the error $(\tau - \tau_{eff})^2$ for every sample in the training set until the validation error starts increasing.

This stategy converges extremely fast (typically less than a hundred iterations on the whole training set; an iteration involves backpropagation on all samples of the training set).

The results of this approach are given on Figure 5.

The performance of the predictor is obviously quite poor in terms of renegotiation; however, it must be noted that the neural network shows excellent generalization properties: in particular, it does react to the burst of activity between 13:00 and 14:00, although this burst fully lies in the test set and no such level of activity occurs in the training or validation sets.

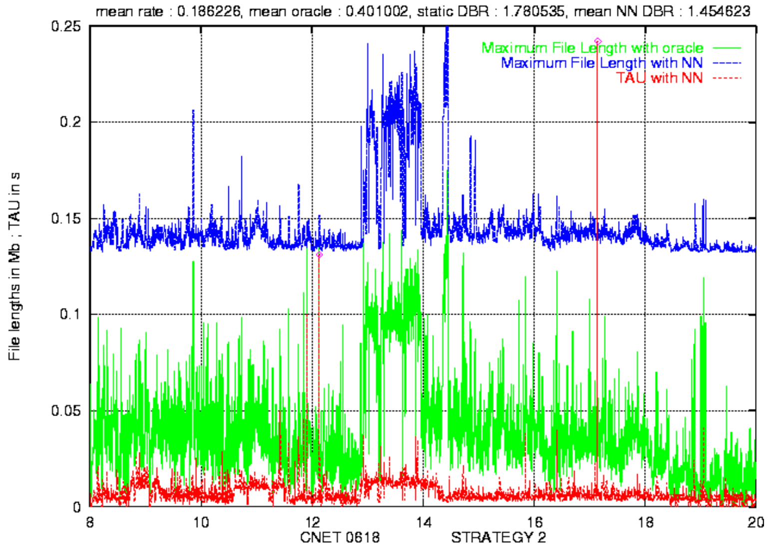


Figure 6: Results for Strategy 2 on a trace collected at CNET on June $18^{th}$ 1996. The training set is made of the first 2 hours, the validation set of the next 2 hours and the rest of the trace (8 hours) is used as the test set. The dashed curve (upper curve) is the maximum file length predicted by the neural network, the solid line (middle curve) is the maximum file length predicted by the oracle; the dotted curve (bottom curve) is the effective jitter tolerance $\tau_{eff}$ obtained by the neural network ($\tau_{eff} > 0.1$ s means a contract violation in the considered period).

### 8.2.4 Learning strategy 2: a conservative approach

For this strategy we try to get a conservative behaviour of the predictor by progressively specializing the learning process on the worst samples of the training set. The learning strategy can be described as follows:

- until no error is made in the training set
  - run the simulation for the whole training and validation sets
  - backpropagate the error $(\tau - \tau_{eff})^2$ for the worst sample in the training set (ie the largest $\tau_{eff}$) for that run
- until no error is made in the validation set
  - lower $\tau$ to $\tau'$
  - run the simulation for the whole training and validation sets
  - backpropagate the error $(\tau' - \tau_{eff})^2$ for the worst sample in the training set (ie the largest $\tau_{eff}$) for that run

This strategy also converges extremely fast, typically less than a thousand iterations on the training set (note that an iteration involves only one backpropagation on the worst sample of the training set).

As can be seen from Figure 6, the results in terms of renegotiation are more satisfactory; we get only two renegotiation errors, indicated by diamonds, on the whole test set (8 hours) and it is clear that specializing the learning process on the worst samples of the training set makes the neural network predictions conservative.

The drawback of this approach is that the learning process very fast gets specialized to only one sample of the training set; surprisingly, such a strong specialization does not lead to a very poor generalization of the network and this puzzling result is left for further research (we note here that a somewhat similar result was obtained in [3, 4] in a different context).

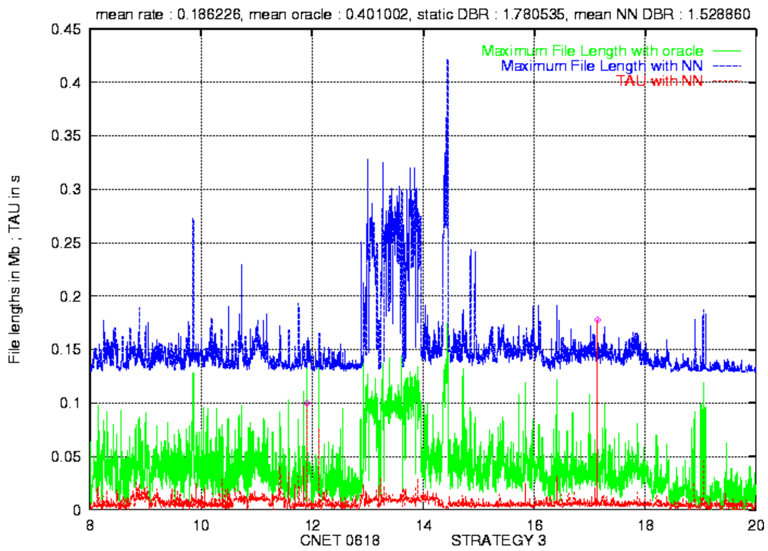


Figure 7: Results for Strategy 3 on a trace collected at CNET on June $18^{th}$ 1996. The training set is made of the first 2 hours, the validation set of the next 2 hours and the rest of the trace (8 hours) is used as the test set. The dashed curve (upper curve) is the maximum file length predicted by the neural network, the solid line (middle curve) is the maximum file length predicted by the oracle; the dotted curve (bottom curve) is the effective jitter tolerance $\tau_{eff}$ obtained by the neural network ($\tau_{eff} > 0.1$ s means a contract violation in the considered period).

### 8.2.5 Learning strategy 3: the best of both worlds

Despite its good results, we felt that Strategy 2 lead to a too sharp specialization of the training which could be detrimental to the generalization abilities of the neural network. We therefore investigated a new strategy which aims at combining the avantages of the two strategies above:

- apply strategy 1, then
- apply strategy 2

Hence, the neural network is taught the entire phase space before being made conservative by specializing on the worst samples of the training set.

As can seen from Figure 7, we do get the best of both stategies 1 and 2 with this approach: the system is conservative as was the case for Strategy 2 and is more adaptive as was the case in Strategy 1.

The performances in terms of renegotiation are indeed excellent, as there is only one contract violation in the whole test set (8 hours). This burst can be considered as an example of a rare (hence, "unforeseenable") event which may lead to a traffic contract violation. Of course, the occurence of such an event is unavoidable when predictors are used to renegotiate the traffic contracts.

If the prediction was to be implemented by the network as a service to the sources, the unavoidable occurence of such events means that these traffic contracts fall into the category of "predictive services": no "hardcore" QoS guarantees are possible (whatever the technique, prediction is indeed a risky business !), but the QoS should be "almost always" as required by the source [13].

If the prediction was to be implemented by the source itself, the network only guarantees the QoS corresponding to the renegotiated contract and any violation of this contract is of the sole source responsability.

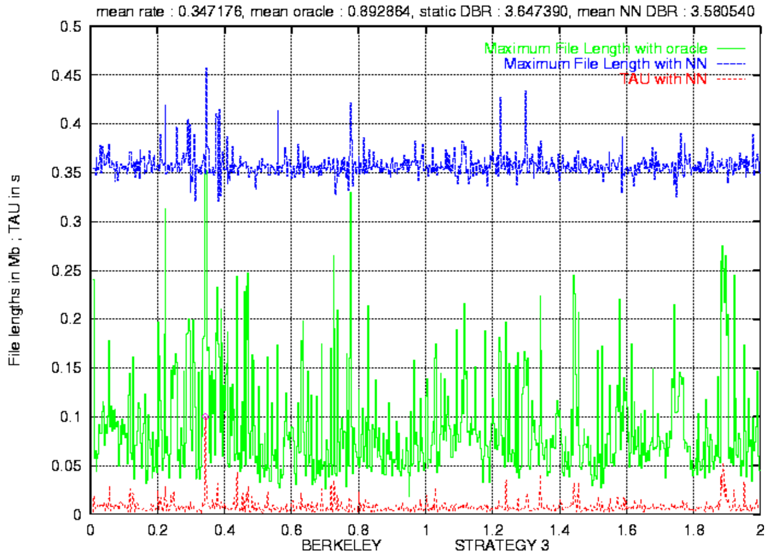


Figure 8: Results for Strategy 3 on the LBL-PKT3 trace collected at Berkeley. The training set is made of the first half hour, the validation set of the next half hour and the rest of the trace (1 hour) is used as the test set. The dashed curve (upper curve) is the maximum file length predicted by the neural network, the solid line (middle curve) is the maximum file length predicted by the oracle; the dotted curve (bottom curve) is the effective jitter tolerance $\tau_{eff}$ obtained by the neural network ($\tau_{eff} > 0.1$ s means a contract violation in the considered period).

For the sake of completeness, Figure 8 shows the results of Strategy 3 when applied to the LBL-PKT3 trace collected at Berkeley. There are no renegotiation errors on the whole test set (last hour of the trace).

### 8.2.6 Another experiment with Strategy 3

In order to test the long-term validity of our predictor, we ran another experiment; we kept the network as it was taught above (i.e. training was performed on a trace collected on the $18^{th}$ June 1996) and used it as a pure predictor on a trace collected two days later (i.e. on the $20^{th}$ June 1996). The results are reported on Figure 9, and we obtain excellent results, with no renegotiation error on the whole trace (12 hours).

Such a result shows that the characteristics captured in the neural network predictor by our training strategy are not strongly dependent on the trace it was taught and are still valid on timescales of days. This, combined with our fast training process and the simple measurements required for the training make the neural network approach to traffic descriptor prediction a perfectly viable technique [1].

## 9 Discussion

Also given in the top line of the above figures are mean bit rates (in Mb/s) characterizing various aspects of the experiments:

- *mean rate* is the mean bit rate of the traffic;
- *mean oracle* is the mean bit rate reserved by the oracle for DBR-10s;

[1] There is no "magic" involved in this however ! We also tried this predictor on the LBL-PKT3 trace and, although the adaptivity was surprisingly good, we got very poor results in terms of renegotiation.

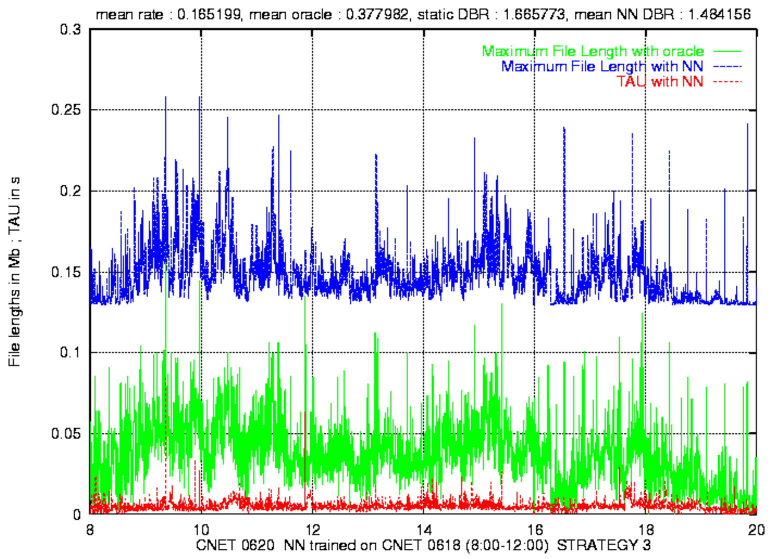


Figure 9: Results for Strategy 3 on a trace collected at CNET on June $20^{th}$ 1996. The neural network was trained as described above with a trace collected on June $18^{th}$ 1996. The dashed curve (upper curve) is the maximum file length predicted by the neural network, the solid line (middle curve) is the maximum file length predicted by the oracle; the dotted curve (bottom curve) is the effective jitter tolerance $\tau_{eff}$ obtained by the neural network ($\tau_{eff} > 0.1$ s means a contract violation in the considered period).

- *static DBR* is the minimum bit rate reserved for the whole trace in the case of a standard static DBR;
- *mean NN DBR* is the mean rate reserved by the neural network predictor for DBR-10s.

### 9.1 Comparison with the oracle

It is clear that the reservation made by the neural network predictor is much larger than the reservation made by the oracle. This is easily interpreted since the neural network indeed tries to predict the *worst* future behaviour of the queue from the characterization of the past traffic and *all* the behaviours it has seen during the learning phase; the oracle knows perfectly the future, so that it makes its reservation on the basis of only one particular instanciation of the future behaviour of the queue, which is not necessarily a worst case instanciation.

Hence, the quantitative comparison between the oracle and the predictor is not very informative. The main comparison should be a qualitative one as we already discussed: the oracle is closely taylored to the needs of the source and, at least for our Strategy 3, the comparison between the behaviours of the reservation of the oracle and of the neural network predictor shows that the neural network predictor indeed follows the big features of the activity of the source with some kind of safety margin.

### 9.2 Comparison between DBR-10s and static DBR

A better quantitative information can be drawn from the comparison between the mean reservation made by the neural network predictor and the reservation of the best static DBR contract.

From Figure 9 it can be seen that the mean DBR rate renegotiated by the neural network predictor (1.48 Mb/s) is smaller than the best static DBR contract (1.67 Mb/s) which could be negotiated for the whole trace (note that in order to negotiate such a contract you need to know the whole trace beforehand whereas our predictor has never seen this trace during its training process !).

This indeed shows that neural network-based traffic contract renegotiation allows to save bandwidth while maintaining the quality of service.

### 9.3 Future work

Although excellent results have been obtained, our neural network is far from being optimal. In particular, it can be seen that the neural network does not seem to adapt correctly its behaviour in low activity parts of the trace (see the CNET results in the 18:00-20:00 range). We are planning to use more sophisticated neural network architectures recently developped for pattern recognition [12, 8] in order to solve this problem.

We are also currently extending this study to other traffic traces from different origins and to different sets of parameters. We also plan to extend this work to the SBR traffic handling capacity.

This presentation was restricted to the ATM context but, as the Internet evolves towards an Integrated Service Packet Network (ISPN) [6], it has also defined "traffic descriptors" based on leaky buckets which are used for the resource reservation in the network. Therefore the techniques developped here can also find applications in the Internet ISPN context. This may be even more natural since the in-service renegotiation capability is included in the signalling protocol RSVP [21].

## 10 Conclusion

In this contribution, we have shown that the use of neural networks indeed allows accurate predictions of the extremal behaviour of a queue driven by a real traffic trace; we presented fast and intuitively simple learning algorithms for this difficult problem and successfully applied them to the dynamic resource reservation in an ATM network with a prediction horizon as large as 10 s.

It has been shown that taking advantage of this prediction capability to periodically renegotiate the parameters of ATM layer traffic handling capabilities was benefitial in terms of reserved resources.

Such results are extremely encouraging for the use of connectionist prediction techniques for the management of a bursty traffic in B-ISDN networks.

## 11 Acknowledgements

The authors express their gratitude to Vern Paxson (Lawrence Berkeley Laboratory) who made the LBL-PKT3 data available to them.

## References

[1] P. Boyer, F. Guillemin, M. Servel, and J. Coudreuse, *Spacing cells protects and enhances utilization in ATM network links*, IEEE Communications Magazine, (1992), pp. 38–49.

[2] P. Boyer and D. Tranchier, *A reservation principle with applications to the ATM traffic control*, Computer networks and ISDN systems, 24 (1992), pp. 321–334.

[3] T. X. Brown, *Adaptive access control applied to Ethernet data*, in Advances in Neural Information Processing Systems, 9, MIT Press, 1997.

[4] ——, *Adaptive statistical multiplexing for broadband communications*, in Tutorials of the 5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, D. Kouvatsos, ed., 1997.

[5] S. Chong, S. Li, and J. Ghosh, *Predictive dynamic bandwidth allocation for efficient transport of real-time VBR video over ATM*, IEEE J. on Selected Areas in Communications, 13 (1995), pp. 12–23.

[6] D. Clark, S. Shenker, and L. Zhang, *Supporting real-time applications in an integrated services packet network: architecture and mechanism*, Computer Communication Review, 22 (1992), pp. 14–26.

[7] F. Clérot, S. Bengio, A. Gravey, and D. Collobert, *Dynamical resource reservation scheme in an ATM network using neural network-based traffic prediction*, in Proceedings of the 5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, D. Kouvatsos, ed., 1997.

[8] R. Feraud, *A modular face detection system*, in Neurap Proceedings, 1997.

[9] M. Grossglauser, S. Keshav, and D. Tse, *RCBR: A simple and efficient service for multiple time-scale traffic*, Computer Communication Review, 25 (1995), pp. 219–230.

[10] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Networks, 2 (1989), pp. 359–366.

[11] I.371, *Traffic control and congestion control in B-ISDN.*

[12] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, *Adaptive mixtures of local experts*, Neural Computation, (1991), pp. 79–87.

[13] S. Jamin, P. Danzig, S. Shenker, and L. Zhang, *A measurement-based admission control algorithm for integrated service packet networks*, Computer Communication Review, 25 (1995), pp. 2–13.

[14] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, *On the self-similar nature of Ethernet traffic (extended version)*, IEEE/ACM Trans. on Networking, 2 (1994), pp. 1–15.

[15] J. Mignault, A. Gravey, and C. Rosenberg, *A survey of straightforward multiplexing models for ATM networks*, in ATM Expert RACE Symposium, 1995.

[16] V. Paxson and S. Floyd, *Wide area traffic: the failure of Poisson modelling*, Proc. Sigcomm'94, Computer Communication Review, 24 (1994), pp. 257–268.

[17] Q.2963, *ITU-T SG11, Draft Recommendation.*

[18] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning internal representations by error propagation*, in Parallel Distributed Processing, D. E. Rumelhart and J. L. McClelland, eds., vol. 1, MIT Press, 1986.

[19] F. Takens, *Detecting strange attractors in turbulence*, in Dynamical Systems and Turbulence, D. A. Rand and L.-S. Young, eds., vol. 898 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1980, pp. 366–381.

[20] S. Vaton, E. Moulines, and H. Korezlioglu, *Statistical identification of WAN traffic data*, in Proceedings of the 5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, D. Kouvatsos, ed., 1997.

[21] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, *RSVP: a new ressource reservation protocol*, IEEE Networks, 7 (1993), pp. 8–18.