

# Long Horizon End-to-End Delay Forecasts: A Multi-Step-Ahead Hybrid Approach

Vinh Bui and Weiping Zhu  
The University of New South Wales, Australia  
{v.bui, w.zhu}@adfa.edu.au

Antonio Pescapé and Alessio Botta  
University of Napoli “Federico II”  
{pescape, a.botta}@unina.it

## Abstract

*A long horizon end-to-end delay forecast, if possible, will be a breakthrough in traffic engineering. This paper introduces a hybrid approach to forecast end-to-end delays using wavelet transforms in combination with neural network and pattern recognition techniques. The discrete wavelet transform is implemented to decompose delay time series into a set of wavelet components, which is comprised of an approximate component and a number of detail components. Thus, it turns the problem of long horizon delay forecasting into a set of shorter horizon wavelet coefficient forecasting problems. A recurrent multi-layered perceptron neural network is applied to forecast coefficients of the wavelet approximate component, which represents the trend of the delay series. The k-nearest neighbors technique is used to forecast coefficients of the wavelet detail components, which reflect the burstiness of background traffic. The proposed approach has been verified in both simulation and over real heterogeneous networks showing promising results in terms of averaged normalized root mean square error. In addition, when compared to some existing and well known approaches it presents the superior performance.*

## 1 Introduction

Despite considerable efforts have been placed on the Internet to assure the Quality of Services (QoS), the dominance of TCP/IP and its best effort policy make it almost impossible to achieve a sufficient QoS guarantee without dramatically changing the protocols. An alternative approach is to move the QoS provision up to application level by building an overlay network on top of the physical network. As a consequence, various overlay networks and services, e.g. RON [1], SON [5], QRON [10], OverQoS

[25] have been proposed. Initial results show the flexibility and feasibility of the approach, which suggests the necessity of further studies. More recently, a new framework for Internet management and control architecture called 4D (*decision, dissemination, discovery and data*) has been proposed [6]. In the framework, traffic control has been moved from routers and switches to end-to-end mechanisms, which rely on packet delays and/or losses to adjust traffic intensity. In addition, if multiple paths are available, end-to-end loss/delay information also can be used to optimally route traffic from source to destination [12].

End-to-end delay estimation and forecasts are essential to the realization of network end-to-end control. For instance, in multiple paths QoS routing, forecasts of end-to-end delays can be used to compute the optimal routing policy. For this purpose, an accurate long horizon forecast is necessary. Despite a reasonable amount of research has been carried out recently aiming to forecast end-to-end delay behaviors [29, 27, 28, 20, 9, 11, 30, 8], there is a lack of efforts on long horizon end-to-end delay forecasts. To remedy this and consequently make end-to-end traffic control possible, we propose an approach to forecast end-to-end delays, which is capable of predicting hundreds steps ahead.

An end-to-end delay refers to the time taken by a packet to traverse from a source to a destination. In a network like the Internet, end-to-end delays are usually observed in the form of a noisy and non-stationary process [32].

In order to forecast such a process, we propose a hybrid approach, which is involved in a three-steps technique. Firstly, the approach uses the wavelet transform to decompose the process into a wavelet approximate component plus a set of wavelet detail components. Secondly, it uses a Recurrent Multi-Layered Perceptron (RMPL) neural network and the k-nearest neighbors pattern recognition technique to predict future coefficients of the wavelet approximate and detail components. Finally, the predicted coefficients are transformed back to a new delay series by means of inverse discrete wavelet transformation. In addition, the first step also decomposes a hundreds-step-ahead delay forecasting problem into a set of fewer-step-ahead wavelet

<sup>0</sup>This work is supported by University of New South Wales at Australian Defence Force Academy. This work has been partially supported by CONTENT NoE, OneLab and NETQOS EU project.

coefficient forecasting stages, which increase the forecasting horizon and accuracy. The proposed approach has been verified by using MATLAB Neural Network Toolbox [14], Wavelet Toolbox [15], NS-2 [16], and real Internet measurements over heterogeneous networks. The results show that it is feasible to forecast end-to-end delays for a few hundreds packets ahead with a low averaged normalized root mean square error. Also, when the forecast horizon is long enough e.g. 320 steps ahead, the forecast accuracy is significantly better than that obtained by using the best known delay forecasting method proposed by Parlos [20] (i.e. the Parlos's gives the averaged error of 0.37 when, under the same conditions, the proposed method gives 0.26).

To highlight the significance of the proposed approach, we underline that: (i) it enables a hundreds-step-ahead end-to-end delays forecast; (ii) it proposes a forecasting method, which incorporates the discrete wavelet transform, neural networks and the k-nearest neighbors technique to deal effectively with non-stationarity of end-to-end delays; (iii) the forecasting method has been verified in both simulation and over real heterogeneous networks, which confirm its accuracy, robustness and durability against the best known approach proposed so far; (iv) it has made an important step towards the realization of end-to-end traffic control.

The rest of the paper is organized as follows. In Section 2, a review of related works is presented. The new forecasting approach is discussed in Section 3. Section 4 proposes the new delay forecasting algorithm. In Section 5, the performance of the proposed algorithm is illustrated through simulation and experimental studies. Section 6 provides concluding remarks.

## 2 Related Works

End-to-end delay forecast with different focuses has been addressed in a number of papers. In particular, problems of delay boundary prediction were studied in [9] and [11]. While [9] proposed using time-series ARIMA technique to carry out the prediction, [11] improved the prediction accuracy by introducing the Maximum Entropy Principle (MEP). In other directions, [30] tried to predict playout delays of VoIP packets by modeling them with a Hidden Markov Model (HMM), whereas in [8] the authors propose a HMM based approach to jointly model and predict delay and losses. Also using ARIMA technique, a short-term round trip time delay forecast was considered in [29]. However, more closely related to our work are [20] and [28], which focused on long-term delay forecasts.

In [20], Parlos proposes the use of a RMPL neural network training subsequently with both supervised and reinforcement learning algorithms to perform a multi-step-ahead prediction of end-to-end delay changes. The reinforcement learning is carried out by means of a so-

called global feedback (GF) to improve the network long-term forecast capability. The performance of the proposed method has been demonstrated in the numerical study through an accurate 100-step-ahead prediction. However, a forecaster purely based on neural networks is hardly to be robust for long-term end-to-end delay forecasts. The reason is that end-to-end delays as a noisy and non-stationary process have not been systematically studied and poorly understood. For such a process, a neural network usually can be a good approximator only at a segment it has been trained for. As far as the network moves away from the trained segment, the approximation would be less accurate. Although the network training can be taken on-line, the long training time could prevent us from using neural networks for hundreds-step-ahead end-to-end delay forecasts. Apart from that, Parlos uses the packet inter-departure time as an input to his neural predictor, which certainly improves the accuracy of the forecast. Nonetheless, in practice the packet inter-departure time may not be available without the support of the operating systems. We, on the other hand, do not require this information to enhance the forecast accuracy.

More recently, Yang et. al., in [28], use a series of parallel Auto-Regressive (AR) models constructed using vector quantization to provide a multi-step-ahead RTT delays forecast. The use of the parallel models allows the approach to overcome the limitation of a single model in forecasting non-stationary RTT delay. At each time slice, only a number of the models is made active based on the actual measurements of RTT delays. The model selection is a Markov-based process, in which  $\pi_{ij} = P(m_{k+1}^j | m_k^i)$  denotes the probability the model  $m^j$  will be activated at time  $k + 1$  given the model  $m^i$  is in effect at time  $k$ . Outputs of the active models are dynamically combined to give an overall prediction. The numerical study using probing flows with a packet inter-departure time of 0.5 second has shown the approach is capable of forecasting up to 10 seconds, which are equivalent to 20 packets ahead. The performance of the forecast however is calculated based on the root mean square error (RMSE), which does not give a real indication of the forecast accuracy since the absolute values of RTT delays are small.

However, to effectively control traffic, a long horizon delay forecast, which comprises a few hundreds steps ahead, would be needed. The lack of research in this aspect, as well as the limitation of the previous ones, indicate the necessity of research on this direction. In the following sections, we present a hybrid approach to overcome the limitation of the previous approaches in long horizon delay forecasting. In specific, we incorporate the discrete wavelet transform, RMLP neural networks, and the k-nearest neighbors pattern recognition technique to deal with challenges in forecasting noisy and nonstationary delays for such a long horizon. First, we reduce the forecast horizon, hence the

effect of nonstationarity, by “transforming” the problem of hundreds-step-ahead delay forecasting into a set of fewer-step-ahead wavelet coefficients forecasting stages. We then try to eliminate the effect of noise by applying a suitable technique to forecast coefficients of each wavelet component. As a result, we could maintain a reasonable accuracy in hundreds-step-ahead delay forecasts.

### 3 Delay Forecasting Approach

End-to-end delays of a TCP flow is a noisy and non-stationary process [32]. From our observations, end-to-end delays strongly reflect the dynamics of the traffic control. Under low intensity and even background traffic, the delays are mostly stationary as the traffic control is relatively stable. The non-stationarity and the noise become obvious with the increase of the interaction between the traffic control and short-term bursty background traffic. If the delays were fully non-stationary it would be very challenging to forecast them for such a long horizon. Fortunately, in many cases end-to-end delays exhibit long-term non-stationary but short-term stationary behaviors [3, 31]. Therefore, by reducing the forecast horizon, hence the effect of delay non-stationarity, it is possible to realize a long horizon delay forecast with a reasonable accuracy. To perform the forecast horizon reduction, multi-level Discrete Wavelet Transforms (DWT) [21] will be applied.

#### 3.1 DWT: An Analytical Basis

Measured samples of end-to-end delays, from the signal processing point of view, are a discrete-time signal  $x(t)$ , which can be transformed into a sum of a countably-infinite set of wavelet functions by dilating and translating a wavelet prototype function or the mother wavelet. In principle, Wavelet transforms are very similar to Fourier transforms. The main difference between the two is that, in place of infinite *sin wave* functions, Wavelet transforms use *wavelets*, which are short-lived wave functions. The use of wavelets allows the transforms to cope effectively with non-stationary signals. Mathematically, the  $s$ -level discrete wavelet transform of  $x(t)$  can be written as follows [21]:

$$x(t) = \sum_{\tau \in Z} a_{s,\tau} \phi_{s,\tau}(t) + \sum_{k=0}^s \sum_{\tau \in Z} d_{k,\tau} \psi_{k,\tau}(t) \quad (1)$$

where  $a_{s,\tau}$  are wavelet *approximate coefficients* at level  $s = 2^k$  ( $k \in Z$ ), obtained from  $x(t)$  and the complex conjugations  $\phi_{s,\tau}^*(t)$  of the scaling functions  $\phi_{s,\tau}(t)$ ;  $d_{k,\tau}$  are wavelet *detail coefficients* of  $x(t)$  at detail level  $k$ , obtained from  $x(t)$  and the complex conjugations  $\psi_{k,\tau}^*(t)$  of the wavelet functions  $\psi_{k,\tau}(t)$ ;  $\phi_{s,\tau}(t) = 2^{-s/2} \phi(2^{-s}t - \tau)$  are scaling functions at level  $s$ , obtained by dilating and transforming the scaling function  $\phi(t)$ ;  $\psi_{k,\tau} = 2^{-k/2} \psi(2^{-k}t - \tau)$  are wavelet functions at level  $k$ , obtained by dilating and transforming the mother wavelet function  $\psi(t)$ .

The first sum in equation (1) is the approximation of  $x(t)$  at the coarsest time scale  $2^s$ , which represents the low frequency part of  $x(t)$ . In many situations, the approximation is the dynamics of the signal. The second sums are the detail part of  $x(t)$ , which often exhibits the noise. In practice, the approximation of  $x(t)$  can be obtained by passing the signal through a low-pass filter. On the other hand, the detail part of the signal, which is high-frequency can be filtered out using a high-pass filter. Therefore, the multi-level DWT can be obtained in a much faster manner by using a bank of cascaded filters [13] as illustrated in Fig. 1.

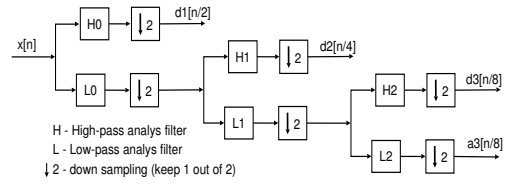
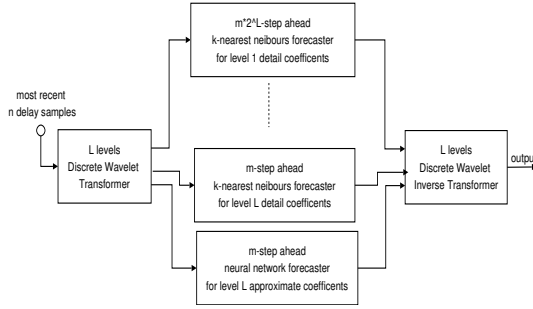


Figure 1. 3-level discrete wavelet transform.

The required number of decomposition levels is always a concern of any application applying wavelet transforms. Although a signal can be repeatedly decomposed until no information gain can be achieved, a few levels of decomposition is usually needed in practice. For a discrete time signal consisting of  $N$  samples, the maximum number of decomposition levels is less than  $\log_2(N)$ . However, the actual number of decomposition levels is problem specific, which can be determined using a rule called Minimum-Entropy [15]. By waveletly transforming the delay time series, we have reduced the forecast horizon since the transforms involve down sampling (see Fig. 1). In addition, we have also filtered out the dynamics of the delay series from the noise. Owing to the differences in essence between the dynamics of the series and the noise, neural networks and the  $k$ -nearest neighbors technique are applied to respectively forecast the coefficients of the wavelet approximate and wavelet detail components.

#### 3.2 Wavelet Coefficients Forecasts

Neural networks have been implemented for modeling the dynamics of complex processes [18, 2] including end-to-end delays [32]. The ability to learn, adapt and generalize allows a neural network with an appropriate structure to approximate virtually any measurable function [7]. However, when the function is complicated the approximation is usually correct around the trained segment. Therefore, instead of using neural networks to predict the coefficients of all components, we implement neural networks to model and



**Figure 2. The hybrid delay forecaster.**

forecast the coefficients of the wavelet approximate component only. The implementation is based on a RMLP network, which is more suitable for multi-step-ahead prediction problems compared to standard feed-forward MLP networks [19]. Thanks to the wavelet transform, one step in the wavelet approximate coefficient forecast would be equivalent to  $2^k$  delay values after the wavelet inverse transformation ( $k$  is the number of the wavelet decomposition levels). Thus, the neural network has to perform only a few-step-ahead forecast instead of a hundreds-step, which has significantly increased the forecast accuracy of the approach.

To forecast the coefficients of the wavelet detail components, which represent the dynamics of short-term bursty e.g. short-lived TCP and ON/OFF UDP background traffic we implement the k-nearest neighbors technique, which predicts a future behavior of an object by searching for a similar behavior of the object in the history. This type of prediction is applicable when the forecasting object exhibits some repeated behaviors over time e.g. seasonal effects in the Internet traffics [17]. Interestingly, the dynamics of the background traffics also exhibit such type of behaviors. Although several pattern recognition based techniques are available, we choose the k-nearest neighbors due to its simplicity and speed [4]. The structure of the proposed hybrid delay forecaster comprised of a neural network forecaster and k-nearest neighbors forecasters is presented in Fig. 2.

## 4 The Proposed Forecasting Algorithm

### 4.1 Algorithm

The forecasting process is started with the decomposition of the most recent delay data series into wavelet components. Afterward, coefficients of the approximate component are fed to the neural network forecaster while coefficients of the detail components are fed to the corresponding k-nearest neighbors forecasters. The new forecasted coefficients are then inversely transformed to a new delay series. The forecasting algorithm is implemented as follows:

1. Form a vector  $D$  of  $n$  most recent delay values:

$$D = \{d(k+n-1), d(k+n-2), \dots, d(k)\}$$

2. Perform  $L$  levels discrete wavelet decomposition of the delay vector  $D$ .
3. Train the neural network forecaster with coefficients of the wavelet approximate component using time series sliding window technique with the window size of  $w$ .
4. Perform  $m$ -step ahead forecast for the approximate component using the trained neural network.
5. Perform  $m * 2^{L-i}$  step ahead coefficient forecast for the  $i$ -th level Wavelet detail component ( $1 \leq i \leq L$ ) using the k-nearest neighbors technique.
6. Perform the inverse discrete wavelet transform to get a new delay vector.

### 4.2 Algorithm Parameters

The above algorithm has been given in a generic form for simplicity. However, there are a number of parameters, e.g. size of the delay vector, wavelet decomposition levels, and structure of the neural network, have to be clearly specified.

Since network end-to-end delay behaviors are noisy and non-stationary, the delay vector must be chosen large enough so it will be a representative of the delay process although a too large vector could include unnecessary historical information that may bias the forecast. The size of the delay vector also depends on the number of the wavelet decomposition levels. Since the wavelet transform involves down sampling, too few samples would result in not enough data to train the neural network predictor. At this stage, we form a delay vector of 10000 samples, which is a common size used by a number of previous works on Internet loss and delay modeling [24, 26]. In our experiments, a longer delay vector does not improve the forecast accuracy while increased the forecasting time.

Another parameter is the number of wavelet decomposition levels. In principle, a signal can be repeatedly decomposed until no information gain can be achieved. Thus, the Minimum-Entropy rule [15], which computes the information gain on entropy basis can be used to estimate the optimal number of wavelet decomposition levels. In our experiments, the number of the decomposition levels usually varies between 1 and 6, in most cases is 3.

Similarly to the delay vector, structure of the neural network also has to be decided empirically. Indeed, a network with more neurons and hidden layers is more flexible and adaptable. However, it also requires more time and data for training. In addition, a larger network does not necessarily guarantee a better generalization capability, which decides the forecast performance of the network. The number

of network inputs is chosen to match the size of the sliding window, which shows how deep the current delay depends on the previous ones. Although a number of works has been carried out on determining the optimal size of the sliding window, it remains an empirically decided parameter in most cases. In our experiments, we have constructed RMLP networks with 1 hidden layer, 20-30 hidden neurons, 10-15 inputs and 1 output. Networks with more hidden layers and neurons per layer have been tested but do not show a significantly better performance.

Apart from the neural network parameters, two parameters of the k-nearest neighbors technique have to be selected are the reference window (*Ref. Wnd.*) and k. If the size of the reference window has to be chosen in the same manner as the sliding window, k is usually taken a value less than 5 mainly for anti-noise purposes. In our experiments, we use the reference window of size 5 to 30 and k equals 2.

The last parameter has to be decided is m - the number of forecasting steps. In our case, m taking a value between 5 and 15 would give a good balance between the number of forecasting steps and the forecast accuracy. The forecast error is computed using a normalized root mean square error (NRMSE), which is defined as follows:

$$NRMSE = \sqrt{\frac{\sum [x(t) - \bar{x}(t)]^2}{\sum x^2(t)}}$$

where  $\bar{x}(t)$  is the forecast value of  $x(t)$ .

## 5 Performance Evaluation

In this paper we report the performance of the proposed approach by using both simulation and real experimentations over real heterogeneous networks.

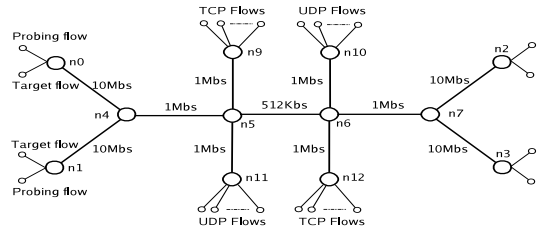
### 5.1 Simulation

In this section, we illustrate the performance of the proposed forecasting method by simulation using NS-2 and MATLAB with Wavelet and Neural Network Toolboxes. The simulation network contains 1000 FTP, CBR and Pareto ON/OFF background traffic sources the parameters of which are reported in TABLE 1. The target flow is a TCP (FTP) flow with the packet size of 512 bytes traversing from node n0 to node n2. All links implement DropTail queuing policy. We simulate the network for 1000 seconds i.e. approximately 20000 packets in the target flow with the averaged loss rate of 1.7%. The network topology is presented in Fig. 3.

After the network simulation, a series of the observed end-to-end delays of the target flow is divided into two equal parts. The first part is used for training (training set) and the second part is for validating the method. The training set

**Table 1. Background traffic sources.**

	FTP	CBR	ON/OFF
Num of Source	700	150	150
Activation	Rand[0-1000]	Rand[0-1000]	Rand[0-1000]
Packet size	64 bytes	512 bytes	128 bytes
Rate	N/A	100Kbps	100Kbps
Send size	Rand[500-5000]	N/A	N/A
Burst time	N/A	N/A	500ms
Idle time	N/A	N/A	500ms
Random/Shape	N/A	True	Pareto(1.6)



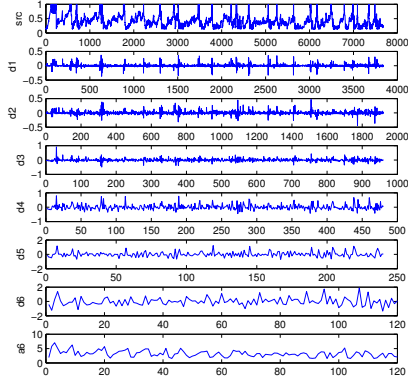
**Figure 3. Simulation topology.**

is subsequently decomposed into wavelet components. The number of wavelet decomposition levels obtained following the Minimum-Entropy rule in this case is 6. The wavelet decomposition of the training set is illustrated in Fig. 4 where src is the original delay series, d1 to d6 are the coefficients of the wavelet detail components and a6 is the coefficients of the approximate component. As shown, d1 – d6 exhibit relatively noisy behaviors with repeated patterns, which are better handled by the k-nearest neighbors technique. On the other hand, a6 shows more predictable behaviors, which can be captured using a neural network.

To model and forecast the coefficients of the wavelet approximate component, we implement a RMLP network with 10 inputs, 1 hidden layer of 30 *tansig* non-linear neurons, 1 output with a *purelin* linear neuron. The network is trained using a similar approach proposed by [20]. An offline training is carried out using *traindxdx* backpropagation algorithm. The learning rate is set to 0.1. The network is trained for 1000 epochs with MSE equals 0.001 and activation of the early stopping mechanism. An on-line training using global feedback as proposed by [20] is taken regularly after each multi-step forecast. On the other hand, to forecast the coefficients of the wavelet detail components, we use the k-nearest neighbors forecasters with k equals 2 and the reference windows of the forecasters 1-5 in correspondence are 28, 24, 18, 10, and 5. The forecasting step m is subsequently set to 5, 10 and 15 in accordance with 320, 640 and 960 packets ahead. For each m, we run the forecasting algorithm 30 times. The results are presented in TABLE 2. From our estimation, for each forecast, it is required from 10ms to 20ms in single run to forecast up to 960 steps ahead exclude the neural network training time.

**Table 2. Forecast results.**

Number of forecast steps	NRMSE	Standard Deviation
320	0.257	0.025
640	0.293	0.042
960	0.321	0.059

**Figure 4. Decomposition of the training set.**

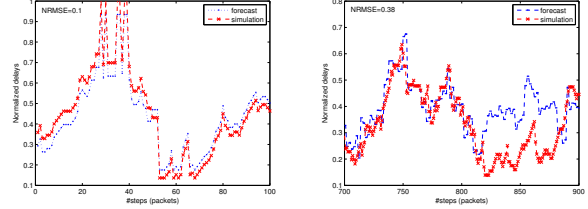
As illustrated, the forecast error increases along with the number of forecasting steps. For the first 100 steps, the forecasted and the actual delay values are close (see Fig. 5 top). As the forecast horizon increases, the forecast values start to move away from the actual ones. A zoomed figure of the last 200 steps in the 960-step-ahead forecast (see Fig. 5 bottom) shows considerable differences between the last 100 forecasted delays and the actual ones. We have tried a longer forecast horizon with  $m = 20$  i.e. 1280 steps ahead. However, the forecast error has grown dramatically from 0.32 to 0.49 as the neural forecaster contributed a big error. In contrast to the neural forecaster, the k-nearest neighbors forecasters do not experience a sudden increase in forecast error since they do not suffer from the error accumulation effect in multi-step-ahead forecasting. Nonetheless, the k-nearest neighbors technique also cannot guarantee a high accuracy in a short-term forecast.

In order to verify the advantage of the proposed hybrid forecasting method, we have built two alternative forecasting methods, one based on pure RMLP neural network forecasters similarly to what proposed in [20] and the other based on pure k-nearest neighbors forecasters. Results from 30 simulation runs presented in TABLE 3 show that the hybrid method performs significantly better than the other two especially for a long forecast horizon. The pure neural network based method performs worst since the neural networks were seriously affected by the noise. Like other forecasting techniques, which rely on function approximation e.g. ARIMA neural networks usually fail to forecast for a long horizon due to the accumulation of forecast errors.

We also compare our method with the method proposed

**Table 3. Comparison of forecast algorithms.**

	NRMSE (320)	NRMSE (640)	NRMSE (960)
Hybrid	0.26	0.29	0.32
Neural	0.31	0.37	0.62
k-nearest	0.42	0.48	0.55

**Figure 5. Zoomed first 50 steps (top) and zoomed last 200 steps.**

by Parlos in [20]. We implement the Parlos's approach in MATLAB. The implementation is based on the code provided by Anderson<sup>1</sup>. We create a RMLP 10-15-1 network with 4 delayed inputs for inter-departure time and 6 delayed inputs for the previously forecasted values. The result from 100 simulations shows that, Parlos's approach performs better than our approach for the first 64 steps (see Fig. 6 top). The reason is due to the use of the k-nearest neighbors technique, which does not perform better than neural networks when the forecast horizon is short. For a longer forecast e.g. 320 and 640 steps, our approach performs significantly better compared to the Parlos's as illustrated in Fig. 6 (bottom). In addition, since Parlos uses inter-departure time as inputs for the neural predictor, his approach is hardly to be applicable in traffic control as these input values are not known in advance. The comparison is presented in TABLE 4.

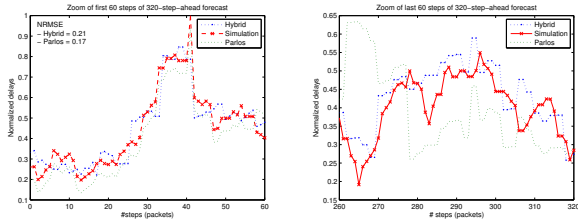
**Table 4. Comparison of forecast algorithms.**

	NRMSE (64)	NRMSE (320)	NRMSE (640)
Hybrid	0.21	0.26	0.29
Parlos	0.17	0.37	0.45

## 5.2 Internet Trace-Based Study

Besides the simulations, we also evaluate the performance of the proposed forecasting method using real Internet measurements performed by using Distributed Internet Traffic Generator (D-ITG) [22]. The data is collected on a real heterogeneous testbed that allows to reproduce several operative conditions. Indeed, it comprises different operating systems (Linux and Windows), access networks (Ether-

<sup>1</sup><http://www.cs.colostate.edu/anderson/code/>



**Figure 6. The proposed method and the Parlos's method in comparison.**

net, IEEE 802.11, ADSL, GPRS, ...), user devices (Workstation, Laptop, Palmtop), and transport protocols (TCP, UDP, ...). Therefore, it allows to assess the impact of several parameters on the forecasting algorithm and to evaluate the robustness of our proposal. To achieve this goal, we avoid the simultaneous variation of more than one parameter, i.e. in each measurement stage, just one parameter is tuned. Data sets are freely available at [23].

Due to space constraints we can not present here all the results we obtained. Therefore, we present those related to 2 scenarios which data sets characteristics are summarized in TABLE 5. The data sets have been collected using TCP, Linux operating system, and Laptop computers. Data traces have been sanitized from the clock skew and therefore they are suitable to verify the goodness of our approach as it is not affected by an offset in the sample values.

**Table 5. Characteristics of the data sets.**

Name	Sender/Rcv.	Pkt size	Bit rate	Proto.	Sample
Eth2Adsl	Ether/ADSL	64bytes	51.2Kbps	TCP	10000
Eth2Wifi	Ether/802.11b	64bytes	51.2Kbps	TCP	3000

For Eth2Adsl data set, we use the first 8000 samples for training and the rest 2000 samples for verifying the forecast. Accordingly, this ratio is 2400 by 600 for Eth2Wifi set. The training procedure is identical to that used in the simulation. TABLE 6 summarizes the parameters of the forecaster used with each data set. The meaning of the parameters in the table can be interpreted as follows: i) *Neural network x-y-z (t)*: means the network has 3 layers, each layer has  $x$ ,  $y$  and  $z$  neurals respectively. The first layer is the input layer and the last layer is the output one. All layers in between are hidden layers.  $t$  is the threshold function e.g. sigmoid, tansig etc. ii) *Wavelet decomposition levels L*: means the number of levels used to waveletly decomposed the signal is  $L$ . iii) *k-neighbor Ref. Wnd. r1-r2-r3-r4*: means the size of the reference window used to forecast the wavelet coefficients at each level of the wavelet decomposition.  $r1$  is for detail level 1,  $r2$  is for detail level 2 and so on.

The results of the experimentations are presented in Fig-

**Table 6. Forecaster Parameters.**

Data set	Neural Network	W.D.Level (L)	k-neighbor
Eth2Adsl	10-20-1(tansig)	6	20-18-15-10-10
Eth2Wifi	2-10-1(tansig)	5	15-15-10-10

ure 7. In the Eth2Wifi case (Fig. 7 bottom) the number of samples (i.e. 3000) did not allow us to forecast too many steps ahead. As we can see, for the steps we evaluated, the forecast error (reported inside the figures) is low and the delay trend is correctly predicted. This is true also for the Eth2Adsl case (Fig. 7 top and middle) for which we were able to test the forecasting until 640 steps ahead. These results confirm those obtained in simulation.

### 5.3 Discussion

Since we intend to use the proposed approach for on-line delay forecasts, the forecasting time is a big concern. Therefore, we propose to use a matrix of neural networks where each network in the matrix is trained with a particular range (patterns) of delays in advance. Depending on input patterns of delays, we can select an appropriate network in the matrix to carry out the forecast. The number of neural networks in the matrix could be relatively small e.g. 50 since delays observed in a particular path or network segment often exhibit some long-term e.g. daily periodic behaviors due to the networking habits of the users.

## 6 Conclusion and Future Work

In this paper, we have proposed a hybrid forecasting method, which incorporates the discrete wavelet transform, neural networks and the k-nearest neighbors technique to fulfill the task of multi-step-ahead flow level delay forecasting. To improve the forecast accuracy, we have limited the forecast horizon of the neural network forecaster to a few steps ahead. A longer horizon forecast is carried out by k-nearest neighbors forecasters, which do not suffer from the accumulation of forecast errors. Thus, we could maintain a reasonable forecast accuracy for a hundreds-step-ahead thanks to the transformation and an appropriate combination of forecasting techniques. We have tested the proposed approach in both simulation and over real heterogeneous networks. As the method is capable of providing a forecast of end-to-end delays for few hundreds packets ahead with a low average NRMSE in a relatively small time, we believe that our contribution can be applied in traffic control to improve Internet Quality of Service.

We are currently working on improving the forecast accuracy by tuning some of the algorithm parameters, which have been so far determined empirically.

