HDAX: Historical Symbolic Modelling of Delay Time Series in a Communications Network

Hooman Homayounfard **Paul J. Kennedy**

Centre for Quantum Computation and Intelligent Systems,
Faculty of Engineering and Information Technology,
University of Technology, Sydney,
PO Box 123, Broadway NSW 2007, AUSTRALIA,
Email: hoomanhm@it.uts.edu.au, paulk@it.uts.edu.au

Abstract

There are certain performance parameters like packet delay, delay variation (jitter) and loss, which are decision factors for online quality of service (QoS) traffic routing. Although considerable efforts have been placed on the Internet to assure QoS, the dominant TCP/IP - like the best-effort communications policy - does not provide sufficient guarantee without abrupt change in the protocols. Estimation and forecasting end-to-end delay and its variations are essential tasks in network routing management for detecting anomalies. A large amount of research has been done to provide foreknowledge of network anomalies by characterizing and forecasting delay with numerical forecasting methods. However, the methods are time consuming and not efficient for real-time application when dealing with large online datasets. Application is more difficult when the data is missing or not available during online forecasting. Moreover, the time cost in statistical methods for trivial forecasting accuracy is prohibitive. Consequently, many researchers suggest a transition from computing with numbers to the manipulation of perceptions in the form of fuzzy linguistic variables. The current work addresses the issue of defining a delay approximation model for packet switching in communications networks. In particular, we focus on decision-making for smart routing management, which is based on the knowledge provided by data mining (informed) agents. We propose a historical symbolic delay approximation model (HDAX) for delay forecasting. Preliminary experiments with the model show good accuracy in forecasting the delay time–series as well as a reduction in the time cost of the forecasting method. HDAX compares favourably with the competing Autoregressive Moving Average (ARMA) algorithm in terms of execution time and accuracy.

Keywords: Time Series Data Mining, Piecewise Linear Approximation, Perception-based Approximation, Delay Forecasting.

1 Introduction

Rapid increases in the number of Internet users and services have prompted researchers within academia and industry to contemplate smart ways of supporting applications with the required Quality of Service (QoS) (Rankin et al. 2005). QoS is the measure of transmission quality and service availability of a network (or internetworks). Service availability is a crucial part of QoS and the network infrastructure must be designed so as to provide high availability to meet QoS. The target of 99.999% availability permits five minutes of downtime per year. Some important metrics for measurement of QoS in the network are:

Delay – The finite amount of time it takes a packet for an end-to-end transmission from sender to receiver. These (sender and receiver) are both edge routers in the same autonomous system in our case.

Delay Variation (Jitter) – The difference in the end-to-end delay between packets. For instance, if one packet and the following packet requires 125 ms and 100 ms, respectively, to traverse their end-to-end trip, then the delay variation would be 25 ms.

Packet Loss – A relative measure of the number of data packets lost during transmission compared to the total number of packets transmitted. Loss is typically a function of availability. That is, if the network is highly available, then loss during periods of non–congestion is zero. However, QoS decision–makers prioritise packets to be transmitted to a link to reduce the likelihood of loss for a specific service level agreement.

By prioritising Internet traffic and the core network more efficiently, QoS and traffic engineering functions can address performance issues related to emerging Internet applications such as real–time voice and video streaming. Consequently, technologies such as those based on software agents[1] are expected to become key tools for the development of future software (Debenham & Simoff 2007). This type of technology may be used in distributed telecommunication environments such as mobile computing, e–commerce and routing management. An effective routing mechanism and its management is crucial to satisfactorily support diverse services in such networks. Routing tables, as the maps in packet delivery throughout the network, are dynamic and get updated by network state–based events (Lau & Woo 2007). Typical network events include node failure, link failure and congestion. A major problem with current routing mechanisms is that they generate routing tables that do not reflect the real–time state of the network and ignore factors like local congestion.

Although considerable efforts have been placed on the Internet to assure QoS, the dominant TCP/IP – like the best–effort traffic handling policy – does not provide sufficient QoS guarantee without abrupt change in the protocols (Bui et al. 2007). Estimation, approximation and forecast of delay variations are essential tasks in network routing management for detecting anomalies. A large amount of research has been done to provide foreknowledge of the

[1] The term *agent* is employed throughout this paper as the automated tool (software) which is capable of acting independently in the network without human monitoring.

network anomalies by characterizing and forecasting delay variation through using numerical forecasting methods. However, the methods are time consuming and/or not efficient for real–time application whilst we are dealing with online and large datasets particularly when data is missing or unavailable. Moreover, the time cost in statistical methods for trivial forecasting accuracy is prohibitive. Zadeh (2001), Keogh et al. (2005), and Batyrshin & Sheremetov (2008) suggest a transition from computing with numbers to the manipulation of perceptions in the form of linguistic variables defined in fuzzy logic.

According to Zadeh (2001), what Lord Kelvin, Rudolph Kalman, William Kahan, and many other outstanding minds did not acknowledge is the fundamental human ability of performing physical and mental works without any complicated numerical computations. Common instances of these works are parking a car; driving in heavy traffic; playing guitar; understanding speech, and summarizing a journal paper. In all cases, the brain works with fuzzy perceptions – perceptions of size, distance, weight, speed, time, direction, smell, colour, shape, force, likelihood, truth and intent, among others – provided by five human senses instead of running a complicated algorithm in the background. In our especial case of delay forecasting model, we aim to simulate this remarkable ability of the brain by memorising the patterns' frequency in the delay (jitter) time series within sliding windows of time. We may later use the provided foreknowledge and manipulate perceptions of delay trends and current delay value to forecast the next trends and values – instead of running a statistical algorithm each time.

The majority of packet loss in the Internet occurs due to congestion in the links. Real–time multimedia applications over the Internet such as Voice over IP (VOIP), video/audio conferencing and streaming are sensitive to packet loss, and packet retransmission is not an acceptable solution with these sorts of application. Predicting packet loss with some certainty from network dynamics of past transmissions is crucial knowledge to inform smart routers to make better decisions. Network applications such as wireless need to identify congestion in the path of packets so as to take suitable rate control actions (Roychoudhuri & Al-Shaer 2005). We propose a data mining model for classification of links that have a high probability of packet loss. The model is intended to contribute to making informed decisions within smart edge routers where the quality of transmissions should be controlled and is primarily determined by the level packet loss.

The basic idea of our proposed model is to predict packet loss in a link by approximating the trends and values of the delay according to observed patterns. We propose a framework utilised by intelligent software agents for data mining, which are installed on edge routers. The agent predicts the likelihood of packet loss for the link, according to a predicted delay level, and periodically broadcasts the availability of the link to its neighbours. This will prevent other links from sending packets to links that report congestions within their network autonomous system. The task becomes intractable when there is no real–time data to accomplish online data mining. In this paper we describe our framework that approximates the probability of packet loss based on the predicted delay values.

The rest of this paper is organised as follows. Section 2 has two parts, the first describing related work in predicting performance traces focusing on delay forecasting and the next describing different network data sources, including the one used in this paper. In section 3 we explain our approach to forecasting network delay values and in section 4 we present results of our experiments on applying our model. Section 5 concludes the paper.

2 Background

2.1 Related Work

Internet packet–loss and delay variation show temporal dependency. If packet n is lost, packet $n + 1$ is likely to be lost. This leads the network to a "bursty" packet loss in real–time communications network (Jiang & Schulzrinne 2000) that may degrade quality of service and the effectiveness of recovery mechanisms such as Forward Error Correction (FEC). In this regard, various researches have studied the delay and packet loss, jitter, available bandwidth and other performance traces in the network so as to predict network anomalies or differentiate them from noise.

A quantitative study of delay and loss correlation patterns from off-line analysis of measurement data from the Internet has been done by Moon (2000), although it did not consider real–time prediction of packet loss from the delay variation data of online communications. A correlation between the bandwidth used and the amount of observed loss is also reported by the researchers while measuring performance of their Mbone system (Handley 1997, Hermanns & Schuba 1996). The path–load measurement in Jain & Dovrolis (2002) employs delay variations to calculate the available bandwidth. However, as confirmed in Roychoudhuri & Al-Shaer (2005), there is not a detailed analysis of online prediction of the packet loss from the delay variation.

Some researchers report techniques using delay variation and TCP window size for congestion avoidance (Brakmo et al. 1994, Jain 1989, Wang & Crowcroft 1991). Others use these results for classification of loss and noise in wireless transmission (Biaz & Vaidya 1998). Moreover, Tobe et al. (2000) discriminates various degrees of congestion according to the relative *one way delay* and *static delay thresholds*. They did not measure or predict delay trends and packet loss based on the congestion level (Roychoudhuri & Al-Shaer 2005).

Other research projects (Carter & Crovella 1996, Dovrolis et al. 2001, Paxson 1998) have employed packet routing techniques to estimate the path capacity and available bandwidth. Specifically, Paxson (1998) examined the correlation between delay and packet loss, but concluded that the linkage between delay variation and loss was weak, though not negligible. In contrast with Paxson's observations, Roychoudhuri & Al-Shaer (2005) attempted to predict packet loss based on delay variation patterns, rather than the overall amount of delay variation. They developed a technique to detect congestion and predict packet loss by means of inter–packet gap variations observations.

Our work addresses the issue of defining a symbolic delay forecasting model, which is based on historical frequencies of observed patterns of the delay variation in adjacent TCP windows. In particular, we will focus our research to produce an "informed" (Debenham & Simoff 2007, Rocha-Mier et al. 2007, Miloucheva et al. 2003) data mining model to be used in a smart network router for online routing management.

2.2 Data Network Scenarios

It is obvious that the type of data used in data mining is highly dependent on the case studies and research goals. For instance, customer data cannot be used in

lieu of network data for predicting faults in a network. However, when making real–time informed decisions before partial network congestion, use of other data repositories may be useful for the decision–making (Weiss et al. 1998). Weiss et al. (1998) also list three types of data used in telecommunications networks: call data, network data and customer data. However, the type of data needed for routing management is network data and there are two salient types mentioned in the literature for fault prediction and isolation: alarm data and QoS data traces.

Rocha-Mier et al. (2007) describe measurement and modelling sequences of various network variables that comprise of time series based on data network statistics. They have created a useful network scenario using OPNET Modeller. The modeller was employed for generating simulated statistics and values of network data variables. Although real network data variables could be derived from the data logs by the use of intelligent agents or manually by the system administrators, there may be violation in accessing data throughout the Internet. Therefore, they adopted the modeller to study various levels of the network traffic load as well as types of services and applications.

Network sequences derived from alarm databases have been used in Klemettinen (1999). Telecommunications databases contain event data from a number of interconnected components, such as switches. Traces used by Klemettinen were produced by the components to report abnormal situations. Klemettinen (1999) views an occurrence a of an alarm as a triple $a = (t, s, m)$, where t is the time of the alarm, s is the "sender" of the alarm, and m is the "alarm message". The alarm "episodes" have all the information about the problems (Weiss et al. 1998). Alarm data are also used by Hatonen et al. (1996) in their knowledge discovery system called telecommunications alarm sequence analyser.

Telecommunications network QoS data traces, on the other hand, are used in novel data mining methods for automation of pattern recognition. These data mining methods use similarity analysis and management of significant patterns in network performance datasets (Miloucheva et al. 2003). This involves analysing the time series of QoS data traces and is required for boosting performance of QoS data mining in network autonomous fault prediction and isolation.

In our experiments, we used delay traces in TCP–dump format from network traffic archives generated by Napoli University "Federico II". The data traces were simulated from real network topologies using Distributed Internet Traffic Generator (D-ITG) (Botta et al. 2007). D-ITG is a software platform that generates traffic at network, transport and application layers over Linux and Windows platforms. It is capable of producing IPV4 and IPV6 traffic traces at packet level accuracy, and replicating appropriate stochastic processes for both Inter Departure Time and Packet Size random variables. The random variables may be obtained from a variety of probability distributions such as exponential, uniform, Cauchy, Gaussian and Pareto.

The generated archives are kept in tar.gz format, each of which contains sample datasets of QoS parameters - packet loss, delay and jitter. The archives are related to several end-to-end paths. Botta et al. (2007) report that samples are generated by adopting an active measurement approach and sending probe packets of 64, 512 and 1024 bytes with a packet rate of 100 packets per second. For each generation experiment for production of One way Delay, Round Trip Time, packet loss and jitter traces it is possible to initialise the random variables seed.

We use non-stationary stochastic delay time series of TCP, UDP and SCTP probe packets sent at regular time intervals to characterise the end-to-end packet delay behaviour of the Internet. The time series are ON/OFF background traffic sources, calculated using non-overlapping windows of 10ms length, for wired, wireless and ADSL network. The data are described in more detail in section 4.

3 Proposed Forecasting Algorithm: Historical Symbolic Delay Approximation (HDAX)

In this section we describe our approach to forecasting the delay values from previously observed patterns of the time series of delay values. Our approach consists of two phases: training and simulation. The training phase uses a time series dataset of delay values to recognise patterns and make the look–up pattern matrix (PDB) mapping the trend patterns to their observed frequency. This phase moves a sliding window over the training data and makes patterns consisting of 3 consecutive trends (previous, current and next) together with their frequency. PDB is then used in the simulation phase on real data to predict the next trend and associated delay value from the current and previously observed trend patterns.

The patterns are described in terms of linguistic variables as a perception–based function (PBF). According to Zadeh (2001) and Batyrshin & Sheremetov (2008), a perception–based function is a fuzzy function resulting from human perception and is given by the rule sets of R_i: "if X is T_i then Z is S_i" where T_i is a categorical (linguistic) term describing some fuzzy intervals A_i on the domain of real numbers and S_i are the descriptions of the shape of $Y(x)$ function on these intervals. For example, regarding the network expert empirical knowledge about delay trends in the network, the PBF rules set is defined as

$$R: \text{"if } X \text{ is } T_k \text{ then } Y_k \text{ is } S\text{"}, (k = 1, \ldots, l) \qquad (1)$$

where Y_k is the trend of the delay time series in the time window k (intervals with 10 seconds length each) and S is the perception–based categorical value of the delay PBF trend defined as *Plain, Increase, Sharply Increase, Decrease, Sharply Decrease* and *Outliers*.

In our case, we represent possible future trends of the QoS time series y_t of delay values at time intervals t and $t-1$ with categorical terms. Note that we use y_t to denote the delay value at time t and Y_k to denote the trend (linguistic variable) for window k. For simplicity, the linearly ordered scale in our experiment also has six linguistic grades (defined above) each of which is a categorical term (assigned to the case number of zero to five respectively).

$$Alphabet = \langle SI, I, P, D, SD, OUT \rangle \qquad (2)$$

The interpretation of these categorical grades is described in Table 1 which relates the case number, the

Table 1: The scale of delay trends (PBF).

Case	Id	Description	$Y = y_t - y_{t-1}$
0	P	Plain	$Y = 0$
1	I	Increase	$0 < Y \leq max/2$
2	SI	Sharply Increase	$max/2 < Y \leq max$
3	D	Decrease	$-max/2 \leq Y < 0$
4	SD	Sharply Decrease	$-max \leq Y < -max/2$
5	OUT	Outlier	$Y < -max$ or $Y > max$

abbreviation, description of the categorical abbreviation and its meaning in terms of the change in values between two consecutive values of the time series Y.

Another notion in our algorithm is the delay level function, which shows the level of delay observed at time t. This value is needed because we might see two kinds of pattern frequency, but in different contexts, as shown in the Fig. 1.

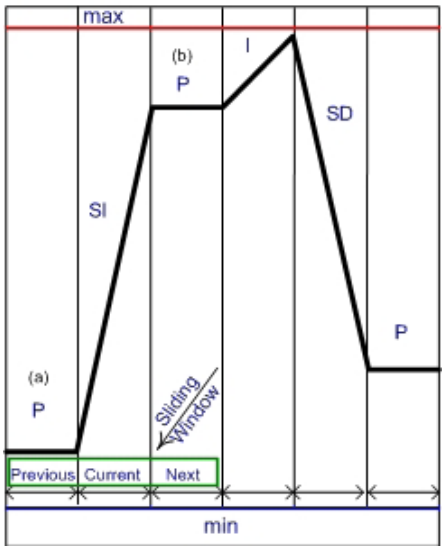

Figure 1: A sample string of symbolic values for the average delay variation time series $\{P, SI, P, I, SD, P\}$

Figure 1 shows that we may have two (or more) contexts of having "plain" trend (labelled as P) in slope sequences. For instance, we may have two cases: a) when the value of y_t is near the minimum and b) when the value of y_t is near the maximum. Therefore, we also need to store the delay level value (ie. whether it is close to the minimum or maximum value) so as to have a finer classification between these two different cases. Consequently, we define a delay level function $F(Y_t)$ as

$$F(Y_t) = \begin{cases} 0 & \text{if } 0 \leq Y_t < max/4 \\ 1 & \text{if } max/4 \leq Y_t < 2 \times max/4 \\ 2 & \text{if } 2 \times max/4 \leq Y_t < 3 \times max/4 \\ 3 & \text{if } 3 \times max/4 \leq Y_t \leq max \\ 4 & \text{if } max \leq Y_t \end{cases} \quad (3)$$

The max parameter used in Table 1 and equation (3) is defined by the user and in this work was empirically set to 60000 ms based on the training dataset used.

After defining the delay level function, we introduce the pattern database (PDB) which is a matrix with a row for each pattern and 6 columns as listed in Table 2. The PDB relates a triplet of values for trends (as listed in Table 1) for the previously observed trend, the current trend and the next trend together with the frequency of the pattern.

Next we describe the training and simulation phases in more detail.

3.1 Training Phase

In this phase, we slide three adjacent windows (previous, current and next) each of length 10 along the training data and increment the frequency value for the associated pattern in the PDB. The window length of 10 was chosen empirically as a balance between performance of the algorithm and generalisation of trends. If the value is too high, there may be too long a time to detect anomalies in delay values. If the value is too low, too much overhead may be placed on the router. The 10 values in a window are averaged to calculate the y_t values, and y_{t-1} and y_t are used to calculate the trend using the scheme outlined above and in Table 1.

Table 2: Description of the fields in the Pattern Database

Field Name	Field Type	Description
Index	Integer	Index of the table
Prev	Categorical {0, 1, 2, 3, 4, 5 }	Categorical based on the value of $y_t - y_{t-1}$
Current	Categorical {0, 1, 2, 3, 4, 5 }	Categorical based on the value of $y_t - y_{t-1}$
Next	Categorical {0, 1, 2, 3, 4, 5 }	Categorical based on the value of $y_t - y_{t-1}$
fyt	Categorical {0, 1, 2, 3, 4 }	Categorical based on the $F(Y_t)$ value
Frequency	Long integer	N/A

For simplicity, we implemented the PDB as a MATLAB matrix, but alternatively it may be created as a physical table in a database. Our PDB matrix contained 1080 rows and 6 columns, as we have an alphabet containing 6 symbols (P, I, SI, D, SD, OUT) and 5 Y_t level ($6 \times 6 \times 6 \times 5 = 1080$).

3.2 Simulation Phase

Once the PDB is populated by training it over a sufficiently long sequence of delay values, it can be used to predict delay trends and values for a new sequence of delay values. We call this the simulation phase.

In this phase we observe the previous 2 delay average values, grouped into two adjacent windows (previous and current patterns) each of length 10, to forecast the next average delay value and trend. The trends for the previous and current windows are calculated in the same way as described in the previous section. The two consecutive (previous and current) patterns (Y_{t-1} and Y_t) are looked up in the PDB and the associated "next" pattern with the highest frequency is chosen as the expected next trend (ie. Y_{t+1}). The next delay value y_{t+1} is estimated from Y_{t+1} (ie. the "next" trend from the PDB) and the "current" delay value y_t using the $F(Y_{t+1})$ function defined as

$$F(Y_{t+1}) = \begin{cases} y_t & \text{if } Y_{t+1} = 0 \\ y_t + max/4 - \sigma & \text{if } Y_{t+1} = 1 \\ y_t + 3 \times max/4 - \sigma & \text{if } Y_{t+1} = 2 \\ y_t - max/4 + \sigma & \text{if } Y_{t+1} = 3 \\ y_t - 3 \times max/4 + \sigma & \text{if } Y_{t+1} = 4 \\ y_t \pm max \pm \sigma & \text{if } Y_{t+1} = 5 \end{cases} \quad (4)$$

where y_t is the current delay value, max is a threshold for detecting the outliers and maximum delay without packet loss and σ is the standard deviation of the y_t values in the training set. The max value was set as in Table 2. In the last condition of equation (4) the max and σ values are added if $y_t > max$, otherwise they are subtracted.

3.3 Measuring the quality of HDAX

An alternative approximation method defined in Box et al. (2008) to approximating the time series value is the Autoregressive Moving Average. An abbreviation of $ARMA(p,q)$ is written in different literature for the mixed autoregressive moving average model with p autoregressive and q moving average terms. As shown in Chatfield (2001), given a time series of data X_t where t is an integer index (indicating time intervals in the experiments) and the X_t are real numbers, then an $ARMA(p,q)$ model is written as

$$(1-\sum_{i=1}^{p}\phi_i)X_t = (1+\sum_{i=1}^{q}\theta_i)\epsilon_{t-1} \quad (5)$$

where p and q are the orders and the ϕ_i and θ_i are the coefficients of the autoregressive and the moving average parts, respectively. The ϵ_t are residuals in prediction, defined by the following recurrent expressions ($\hat{Y}_t$ and Y_t are respective forecasted and original delay values at time t):

$$\epsilon_t = \left|\hat{Y}_t - Y_t\right| \quad (6)$$

The version of ARMA used here has four arguments as its input and generates one step ahead of the entered ARMA time series. The four arguments are the time series value X_t, the ARMA coefficients ϕ and θ, and a control argument *yesmean* that takes one of the two values 1 and 0 – whether the mean is subtracted from the input time series or not.

Since the ARMA model is autoregressive we have to compute the impulse response, which are the coefficients of the equivalent moving average representation. The p and q orders, of ARMA sequence, X_t, may be estimated by minimising the criterion reported in Hannan & Rissanen (1982). However, in our experiments, we set the p and q orders of the ARMA time series empirically. Since $\hat{Y}_t$ is a function of ϕ and θ we estimate ϕ and θ by minimising the logarithm of the sum of squared errors between the actual and predicted values over a window w

$$\log\sum_{t=1}^{w}{\epsilon_t}^2 \quad (7)$$

where ϵ_t is the forecasting error at time t. Based on this, the optimum values of ARMA coefficients be set to $\phi = (1,-0.6)$ and $\theta = (0.6^0, 0.6^1, \ldots, 0.6^j, \ldots, 0.6^{10})$.

To compare the HDAX predicted delay values with the known delay values (from test data) we define a distance function to measure their similarity. For two time sequences $S = s_1,\ldots,s_n$ and $Q = q_1,\ldots,q_n$ with the same length of n, the Euclidean distance of S and Q is

$$D(S,Q) = \sqrt{\sum_{i=1}^{n}(s_i - q_i)^2} \quad (8)$$

In our experiments, we do need to calculate a distance function so as to estimate the accuracy of the algorithms. The HDAX and ARMA results may be compared to test data based on the number of forecasting steps, window sizes, elapsed time and the normalized root mean squared error (NRMSE). This latter is defined as

$$NRMSE = \sqrt{\frac{\sum_{t=1}^{l}(y_t - \overline{y}_t)^2}{\sum_{t=1}^{l} y_t^2}} \quad (9)$$

where y_t and $\overline{y}_t$ are delays and forecasted delays, respectively, and $l < n$ is the length of the subsequence of the original time series. To make the comparison more efficient, we used a comparable, but less complex, error defined as

$$NRMSE = \sqrt{\frac{\sum_{t=1}^{l}(y_t - \overline{y}_t)^2}{n^2 \times (max(y_t) - min(y_t))^2}} \quad (10)$$

4 Simulation Results

In this section we present simulation results for HDAX and ARMA in terms of the accuracy and speed of the algorithm. The experiments ran on a Compaq nx7010 laptop (CPU: Intel Pentium M 1.70GHz; RAM: 1280MB) and a server cluster node with 2x 3.33 Ghz 8MB cache Quad Core Xeon W5590 6.4GT/sec QPI, 24GB(6x4GB) DDR3-1333 ECC Memory and 2 x 300GB 15,000 RPM SAS Hard Drive (Raid 1).

In terms of data for the preliminary experiments with HDAX and ARMA, we used archives that had been generated by Napoli University "Federico II" using the I-DTG platform. As introduced before, the generated archives were ON/OFF background TCP traffic sources for wired, wireless and ADSL network with 100Kbps transfer rate, 500ms burst time and 500ms idle time. Each archive contains a time series of delay values with packet sizes of 64, 256, 512 and 1024 bytes. Each sample is calculated using non-overlapping windows of 10ms length. The non-stationary traces are collected with no time synchronization between sender and receiver computers. Thus, the delay samples can not be used to estimate the average delay, but rather the delay distribution and variance.

In the first simulation phase, to provide more visibility for comparing the forecasting results of both HDAX and ARMA, we used traces of one wired network traffic archive with the packet size of 64 bytes in TCP–dump format. To start using the data for HDAX and ARMA simulations, the original dataset was transformed (3160 delay values) into 316 windows of equal length 10 and the average values of each segment computed. These average values are used in HDAX as described above. The window length of 10, used in these experiments is not suitable for all real–world network scenarios. A network expert should make a trade–off between the algorithm overhead and the performance of the anomaly detection to define the proper window length.

We divided the 316 average delay values into two datasets for training and simulation phases. The first 30% of the data is used in the training phase to build and populate the PDB as described in section 3.1. This comprised 100 y_t delay values. The remaining 70% of the data (216 y_t values) was used for 4 continuous simulation runs each with 54 values. We ran HDAX simulation phase for each set of 54 values to forecast delay values. We then calculated the overall NRMSE using equation (10). Predicted and original delay values for each of the four HDAX and ARMA simulation runs in simulation phase 1 are shown in Figures 2 to 5. These results show that the predicted delay values are close to the expected values in most cases. Figures 2 to 4 show that HDAX sometimes predicts a higher delay than expected when there are sharp increases. Similarly, figures 3 and 4 show lower than expected delay values when there are sharp decreases in delay value. Future work will examine the causes for these misjudgements.

Table 3 shows the normalised root mean square errors (NRMSE) for the four simulation runs in sim-

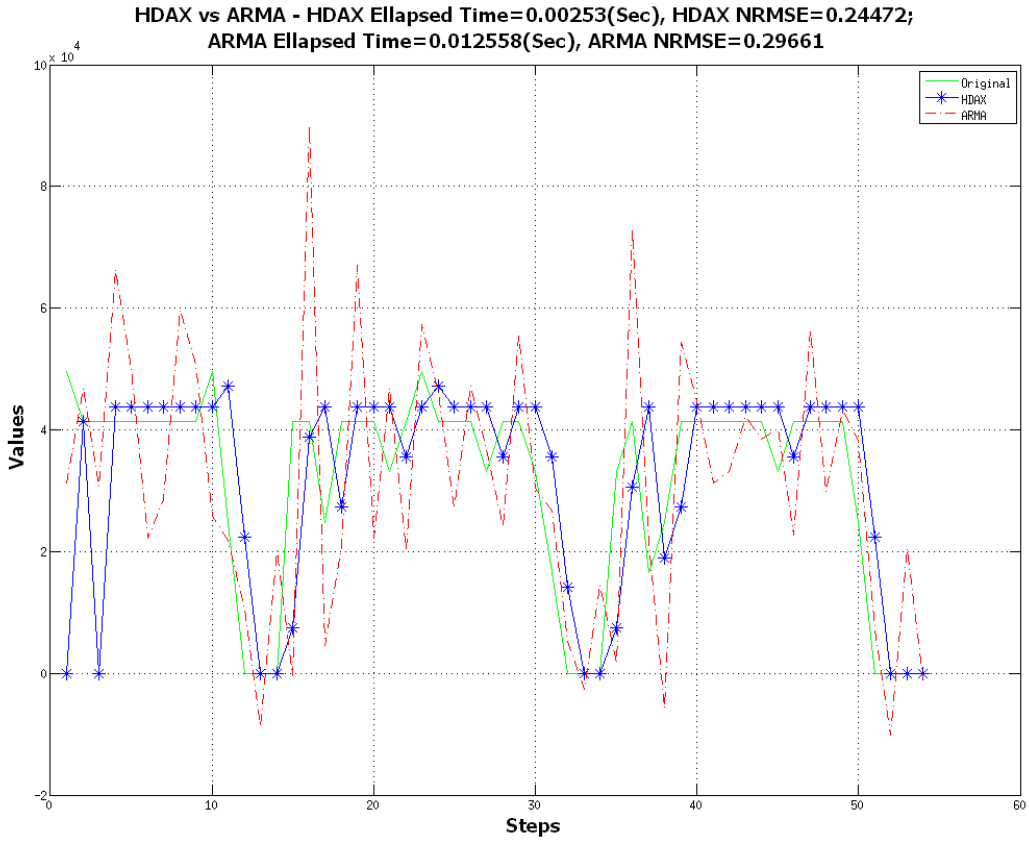

Figure 2: Original (solid line), HDAX predicted (star-dashed line) and ARMA predicted (dot-dashed line) delay values for simulation phase 1 run 1.

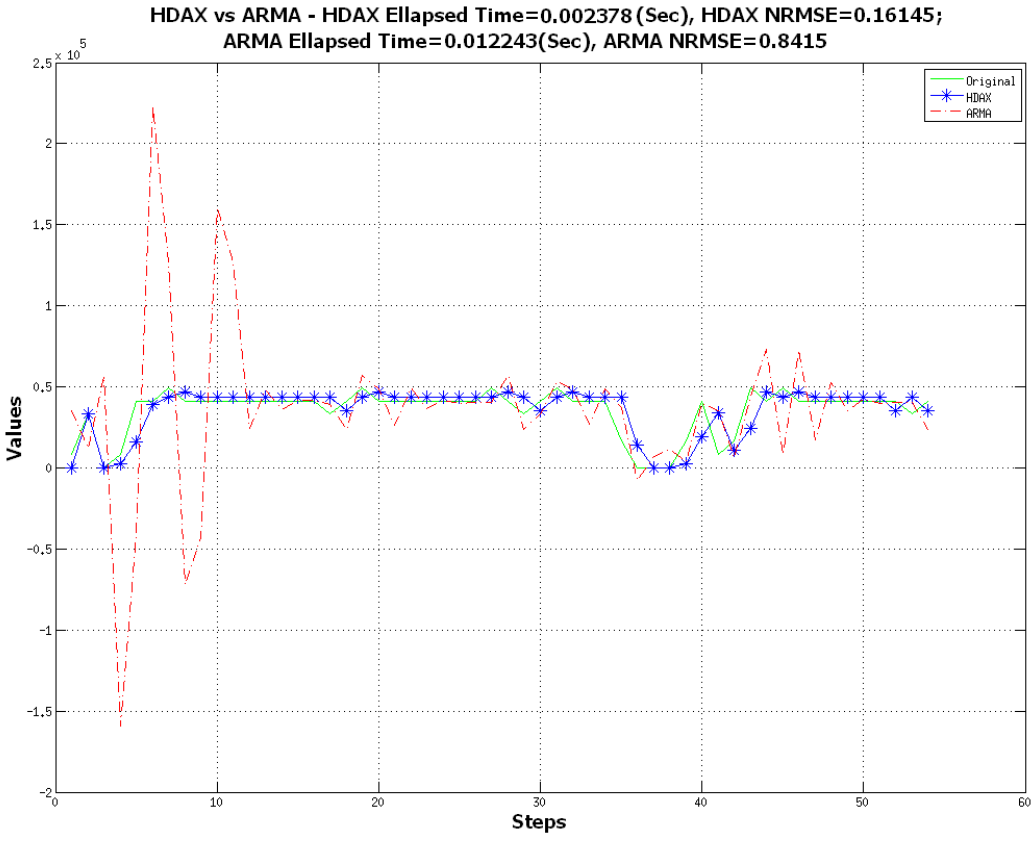

Figure 3: Original (solid line), HDAX predicted (star-dashed line) and ARMA predicted (dot-dashed line) delay values for simulation phase 1 run 2.

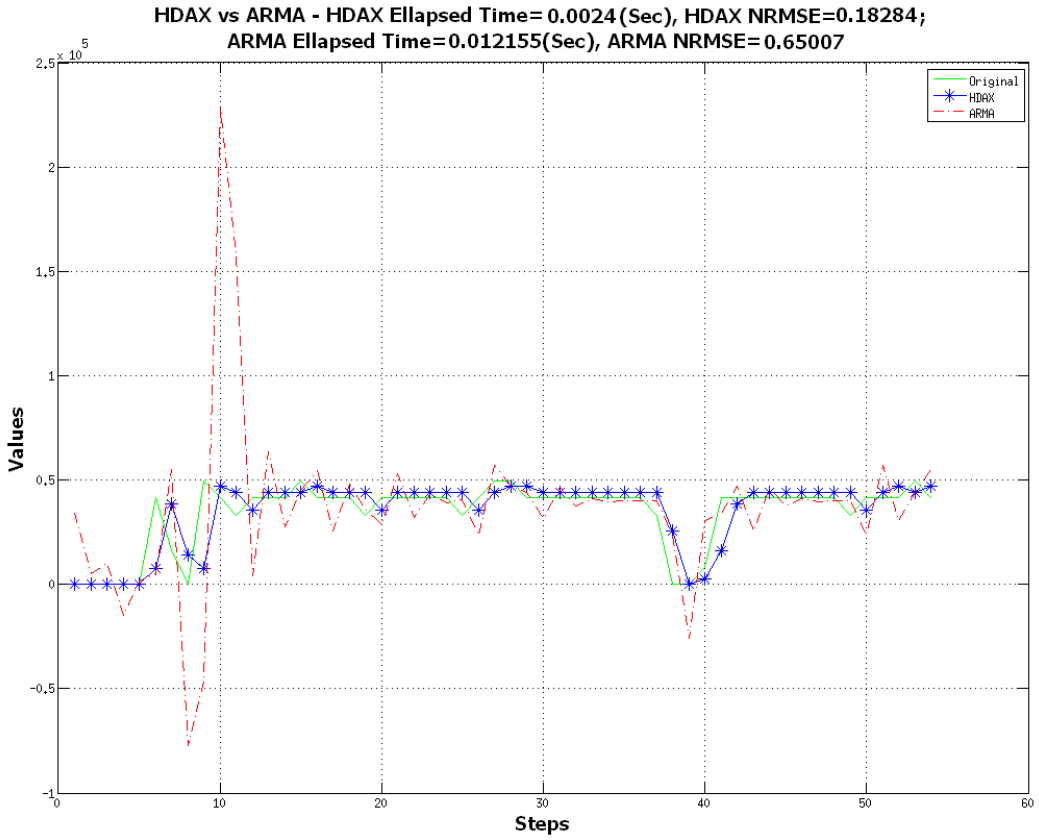

Figure 4: Original (solid line), HDAX predicted (star-dashed line) and ARMA predicted (dot-dashed line) delay values for simulation phase 1 run 3.

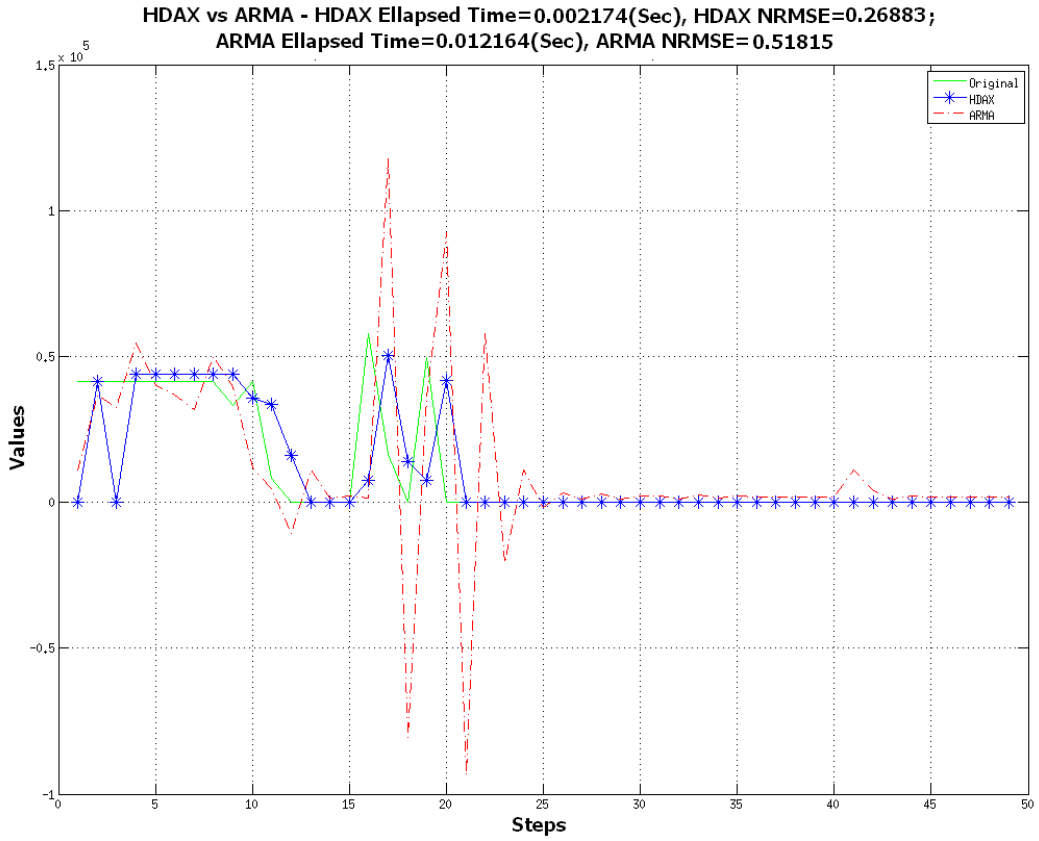

Figure 5: Original (solid line), HDAX predicted (star-dashed line) and ARMA predicted (dot-dashed line) delay values for simulation phase 1 run 4.

ulation phase 1 experiments together with their speed of calculation.

Table 3: Accuracy of HDAX and ARMA (benchmark) on first phase of simulation runs together with speed of calculation.

Simul. Run	HDAX NRMSE	Speed (sec)	ARMA NRMSE	Speed (sec)
1	0.24472	0.00253	0.29661	0.012558
2	0.16145	0.002378	0.8415	0.012243
3	0.18284	0.0024	0.65007	0.012155
4	0.26883	0.002174	0.51815	0.012164

In the second phase of simulations we ran the HDAX and ARMA algorithms with all the Napoli archives. The algorithms' input had variable length between 999 to 1203 average delay values after running the average transformation. Each experiment was repeated at least 5 times to optimise parameters and to show the optimum results in terms of accuracy (NRMSE) and performance (elapsed time). Similar to phase 1, we used one third of each dataset to train and populate HDAX trends lookup table and used the remaining data for running the simulation. As shown in Table 4, HDAX compares favourably with ARMA. It shows an average accuracy of 89% while ARMA model showed 82% accurate. Moreover, HDAX shows slightly better performance in comparison to the used benchmark.

Table 4: Accuracy of HDAX and ARMA (benchmark) in the phase two of simulation runs together with speed of calculation.

MODEL	Accuracy %	Performance (sec)
HDAX	88.725941	0.014780647
ARMA	82.181929	0.016310118

5 Conclusions

This paper presented a delay forecast framework: a novel mechanism to predict delay in real–time streams by observing the delay variation and trends. The framework is part of a predictive model, which is using the forecasted QoS traces such as values of delay and jitter to predict packet loss assigned to a network node (usually an edge router). The proposed approach in this paper determines trends for variation in delay by measuring the delay variation characteristics from ongoing traffic. The motivation is to use this predicted value to indicate the current degree and severity of congestion and likelihood of packet loss, and to use it as a vital component in sender–based error and rate control mechanisms for multimedia.

We presented simulation results showing that the method can predict delay values accurately and quickly. We used the ARMA algorithm, as a benchmark, to test the accuracy and performance of HDAX in comparison to ARMA model. The HDAX algorithm showed average accuracy and speed of 89% and 0.0148 (sec), respectively.

As future work, we need to refine the algorithm to enable forecasting QoS traces with missing values. Then, the whole model in our project, including HDAX, can be installed as a software agent within a smart router on OPNET modeller to run further simulations. We will also refine HDAX metrics using statistical techniques such as estimation theory to improve the accuracy and efficiency of the algorithm. The metrics like maximum delay threshold and window size should be determined dynamically to reflect the prevailing online status of the network. This will enable us to deploy HDAX more reliably. Finally, more formalisation is also required for specifically studying of HDAX delay time series forecaster.

References

Batyrshin, I. Z. & Sheremetov, L. B. (2008), 'Perception–based approach to time series data mining', *Applied Soft Computing Journal* **8**(3), 1211–1221.

Biaz, S. & Vaidya, N. H. (1998), Distinguishing congestion losses from wireless transmission losses: A negative result, *in* '7th Int. Conf. Computer Communications and Networks', Lafayette, LA, pp. 722–731. Proceedings of the Seventh International Conference on Computer Communications and Networks (IC3N).

Botta, A., Dainotti, A. & Pescap, A. (2007), Multi-protocol and multi-platform traffic generation and measurement, *in* 'IEEE INFOCOM 2007', Anchorage, Alaska, USA.

Box, G. E. P., Jenkins, G. M. & Reinsel, G. C. (2008), *Time series analysis: forecasting and control*, Wiley series in probability and statistics, 4th edn, John Wiley, Hoboken, N.J.

Brakmo, L. S., O'Malley, S. W. & Peterson, L. L. (1994), 'TCP Vegas: New techniques for congestion detection and avoidance', *ACM SIGCOMM Computer Communication Review* **24**(4), 24–35.

Bui, V., Zhu, W., Pescape, A. & Botta, A. (2007), Long horizon end–to–end delay forecasts: A multi–step–ahead hybrid approach, *in* '12th IEEE Symposium on Computers and Communications, 2007. ISCC 2007', IEEE, pp. 825–832.

Carter, R. L. & Crovella, M. E. (1996), *Measuring bottleneck link speed in packet–switched networks*, Vol. 27, 28 of *Performance evaluation*, Boston University, Boston, MA, USA.

Chatfield, C. (2001), *Time-series forecasting*, Chapman & Hall.

Debenham, J. & Simoff, S. (2007), Informed agents: Integrating data mining and agency, *in* C. Boukis, L. Pnevmatikakis & L. Polymenakos, eds, 'International Federation for Information Processing–IFIP', Vol. 247, Springer–Verlag, Boston, pp. 165–173.

Dovrolis, C., Ramanathan, P. & Moore, D. (2001), What do packet dispersion techniques measure?, *in* 'INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE', Vol. 2, pp. 905–914.

Handley, M. (1997), An examination of MBone performance, Technical report, ISI Research Report ISI/RR-97.

Hannan, E. & Rissanen, J. (1982), 'Recursive estimation of mixed autoregressive-moving average order', *Biometrika* **69**(1), 81–94.

Hatonen, K., Klemettinen, M., Mannila, H., Ronkainen, P. & Toivonen, H. (1996), TASA: Telecommunication alarm sequence analyzer or how to enjoy faults in your network, *in* 'Network Operations and Management Symposium, 1996., IEEE', Vol. 2, pp. 520–529.

Hermanns, O. & Schuba, M. (1996), 'Performance investigations of the IP multicast architecture', *Computer Networks and ISDN Systems* **28**(4), 429–439.

Jain, M. & Dovrolis, C. (2002), End–to–end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput, *in* 'Proceedings of the 2002 SIGCOMM conference 4', Vol. 32, ACM New York, NY, USA, pp. 295–308.

Jain, R. (1989), 'A delay–based approach for congestion avoidance in interconnected heterogeneous computer networks', *SIGCOMM Comput. Commun. Rev.* **19**(5), 56–71. 74686.

Jiang, W. & Schulzrinne, H. (2000), 'Modeling of packet loss and delay and their effects on real-time multimedia service quality'. ACM Network and Operating Systems Support for Digital Audio and Video.

Keogh, E., Lin, J. & Fu, A. (2005), Hot SAX: Efficiently finding the most unusual time series subsequence, *in* 'Fifth IEEE International Conference on Data Mining (ICDM'05)', IEEE, Houston, pp. 1–8. Fifth IEEE International Conference on Data Mining.

Klemettinen, M. (1999), A knowledge discovery methodology for telecommunication network alarm databases, PhD thesis, University of Helsinki.

Lau, H. Y. K. & Woo, S. O. (2007), 'An agent–based dynamic routing strategy for automated material handling systems', *International Journal of Computer Integrated Manufacturing* **21**(3), 269–288.

Miloucheva, I., Hofmann, U. & Gutirrez, P. A. A. (2003), Spatio–temporal QoS pattern analysis in large scale internet environment, *in* G. Ventre & R. Canonico, eds, 'Interactive Multimedia on Next Generation Networks, LNCS', Vol. 2899, Springer, Berlin, p. 282. Interactive Multimedia on Next Generation Networks: First International Workshop on Multimedia Interactive Protocols and Systems, MIPS 2003, Napoli, Italy, November 2003: Proceedings.

Moon, S. B. (2000), Measurement and analysis of end–to–end delay and loss in the Internet, PhD thesis, University of Massachusetts, Amherst.

Paxson, V. (1998), 'Measurements and analysis of end–to–end internet dynamics', *University of California at Berkeley, Berkeley, CA* .

Rankin, J., Christie, G. & Kondratova, I. (2005), Mobile multimodal solutions for project closeout, *in* 'Proceedings of the CSCE 6th Construction Specialty Conference,', Toronto, Ontario, pp. 2–4.

Rocha-Mier, L. E., Sheremetov, L. & Batyrshin, I. (2007), 'Intelligent agents for real time data mining in telecommunications networks', *Lecture Notes in Computer Science* **4476**, 138.

Roychoudhuri, L. & Al-Shaer, E. (2005), 'Real–time packet loss prediction based on end–to–end delay variation', *IEEE Trans. Network Service Manager* **2**(1).

Tobe, Y., Tamura, Y., Molano, A., Ghosh, S. & Tokuda, H. (2000), Achieving moderate fairness for UDP flows by path–status classification, *in* 'Local Computer Networks, 2000. LCN 2000. Proceedings. 25th Annual IEEE Conference on', pp. 252–261.

Wang, Z. & Crowcroft, J. (1991), 'A new congestion control scheme: slow start and search (Tri-S)', *SIGCOMM Comput. Commun. Rev.* **21**(1), 32–43. 116033.

Weiss, G., Eddy, J., Weiss, S. & Dube, R. (1998), Intelligent telecommunication technologies, *in* L. C. Jain, R. Johnson, Y. Takefuji & L. Zadeh, eds, 'Knowledge–Based Intelligent Techniques in Industry', CRC Press, Boca Raton, pp. 249–275.

Zadeh, L. A. (2001), 'From computing with numbers to computing with words from manipulation of measurements to manipulation of perceptions', *Annals of the New York Academy of Sciences* **929**(1), 221–252.