Adaptive Sampling for Network Management

Edwin A. Hernandez, Matthew C. Chidester, and Alan D. George

High-performance Computing and Simulation (HCS) Research Laboratory
Department of Electrical and Computer Engineering, University of Florida
P.O.Box 116200, Gainesville, FL 32611-6200

Abstract *– High-performance networks require sophisticated management systems to identify sources of bottlenecks and detect faults. At the same time, the impact of network queries on the latency and bandwidth available to the applications must be minimized. Adaptive techniques can be used to control and reduce the rate of sampling of network information, reducing the amount of processed data and lessening the overhead on the network. Two adaptive sampling methods are proposed in this paper based on linear prediction and fuzzy logic. The performance of these techniques is compared with conventional sampling methods by conducting simulative experiments using Internet and videoconference traffic patterns. The adaptive techniques are significantly more flexible in their ability to dynamically adjust with fluctuations in network behavior, and in some cases they are able to reduce the sample count by as much as a factor of two while maintaining the same accuracy as the best conventional sampling interval. The results illustrate that adaptive sampling provides the potential for better monitoring, control, and management of high-performance networks with higher accuracy, lower overhead, or both.*

KEY WORDS: Adaptive sampling; fuzzy logic; linear prediction; network management; SNMP.

1. INTRODUCTION

In network management, accurate measures of network status are needed to aid in planning, troubleshooting, and monitoring. For example, it may be necessary to monitor the bandwidth consumption of several hundred links in a distributed system to pinpoint bottlenecks. If the monitoring is too aggressive, it may create artificial bottlenecks. With too passive a scheme, the network monitor may miss important events. Network query rates must strike a balance between accurate performance characterization and low bandwidth consumption to avoid changing the behavior of the network while still providing a clear picture of the behavior. This balance is often achieved through sampling. Sampling techniques are used to study the behavior of a population of elements based on a representative subset. In general, the samples are taken periodically at some fixed interval or in some random distribution. Such sampling reduces bandwidth and storage requirements for the monitored data.

In a high-performance network, sampling overhead must have a minimal impact on the low-latency application transactions while high network throughputs require frequent sampling to capture transient behavior. For example, many distributed applications (e.g. parallel discrete-event simulation, database systems, etc.) may require frequent synchronization, large data transfers, or both. Network management queries could delay critical data on a single link and thus reduce the efficiency of the entire application.

Under some traffic loads, simple periodic sampling may be poorly suited to the monitoring task. For example, during periods of idle activity or low network loads, a long sampling interval provides sufficient accuracy at a minimal overhead. However, bursts of high activity require shorter sample intervals to accurately measure network status at the expense of increased sample traffic overhead. To address this issue, adaptive sampling techniques can be employed to dynamically adjust the sampling interval and optimize accuracy and overhead.

Adaptive sampling monitors network behavior by dynamically adjusting the sampling time interval for each parameter monitored. When levels of high activity are detected, a shorter sampling interval is employed to measure the behavior of the network with greater accuracy. When less activity is detected, the sampling interval is lengthened to reduce sampling overhead. The adaptive algorithms introduced in this paper are used to select the most representative samples of the population by polling and monitoring key fluctuations in the variables being measured. Consequently, adaptive sampling techniques allow the management to take place in a less-intrusive fashion by avoiding unnecessary queries.

This paper introduces two techniques for adaptive sampling. The first technique is based on Linear Prediction (LP) [1,2]. The LP sampler uses previous samples to estimate or predict a future measurement. The LP logic can be used in conjunction with a set of rules defining the sampling rate adjustments to make when the prediction is inaccurate.

The second technique for adaptive sampling makes use of fuzzy logic. Fuzzy logic mimics the decision processes employed by the human brain [3]. For example, a network administrator may reason that when network load is low, the sample interval can be increased. The fuzzy logic model relies on previous experiences for defining the fuzzy set of parameters and boundaries for the fuzzy variables.

In order to gauge the performance of the adaptive techniques, they are compared with systematic sampling. This technique uses a deterministic interval of time to query the agents. Other non-adaptive sampling techniques include

random sampling where samples are taken at intervals of time determined by a random distribution, and *stratified random sampling* where a sample is taken at a random point within a deterministic time interval [4,5].

This paper introduces the concept of adaptive network sampling and provides experimental results with the various sampling techniques comparing them in terms of accuracy and performance using Internet and videoconference traffic. In Section 2, the sampling techniques used in this study are defined. Section 3 describes the experiments and measurement techniques used to compare the performance of the sampling disciplines are described. The performance results are shown in Section 4. Related research is described in Section 5, while conclusions and directions for future research are provided in Section 6.

2. SAMPLING TECHNIQUES FOR NETWORK MANAGEMENT

In network management, status information regarding load, latency, queue occupancy, and other parameters is frequently available in devices such as routers, switches, and network interfaces. Such information is often accessed through the Simple Network Management Protocol (SNMP) [6-9]. In SNMP, a Network Management Station (NMS) queries the network devices, or agents, to periodically assess the status of the network devices or links.

The period of the sampling determines the accuracy of the measured data. Transient activity may not be accurately detected when the sampling interval is large, while small intervals consume more bandwidth on the network and require greater storage capacity at the NMS. For example, a burst of high activity lasting only seconds is likely to go undetected with a sampling interval of several minutes. In an effort to balance accuracy with sampling overhead, several sampling disciplines have been applied to network managers.

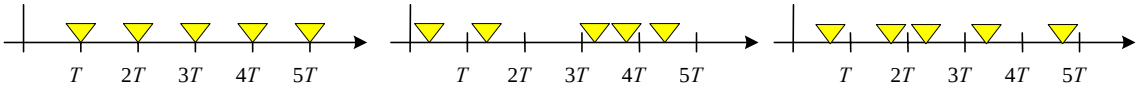
2.1. Conventional Sampling

Traditionally, network management has made use of simple, non-adaptive sampling techniques. Such techniques use a fixed rule to determine when to sample data in each agent. The sampling rule can be deterministic, such as in periodic sampling, or it can involve a random component. Introducing randomness has been shown to improve accuracy in situations where the monitored data is uniform in nature [4]. There are three conventional methods used by network management systems for sampling of agents:

a) *Systematic sampling*, or periodic sampling, deterministically samples data at a fixed time interval. Fig. 1(a) shows systematic sampling with a period of T seconds.

b) *Random sampling* employs a random distribution function to determine when each sample should be taken. The distribution may be uniform, exponential, Poisson, etc. As shown in Fig. 1(b), random sampling may take a varying number of samples in a given time interval.

c) *Stratified random sampling* combines the fixed-time interval used in systematic sampling with random sampling by taking a single sample at a random point during a given time interval. Fig. 1(c) shows stratified random sampling with a time interval of *T*.



(a) Systematic sampling (b) Random sampling (c) Stratified random sampling

Fig. 1. Conventional sampling methods.

Since network traffic is frequently aperiodic, the rate for any of these non-adaptive sampling techniques is typically set based on the expected average network load, the traffic distribution, or simply to a value that will yield an acceptably small amount of overhead. If the actual traffic differs from the expected pattern, the measurements may prove inaccurate or may take an excessive number of samples.

2.2. Adaptive Sampling

Adaptive sampling dynamically adjusts the sampling rate based on the observed sampled data. A key element in adaptive sampling is the prediction of future behavior based on the observed samples. The two adaptive sampling techniques presented in this paper differ by the technique used to predict the future behavior of the system. The Linear Prediction (LP) method attempts to predict the value of the next sample. If the prediction is accurate, the sampling rate can be reduced. Inaccurate predictions indicate a change in the network behavior and require an increased sampling rate to determine the new pattern. By contrast, the Fuzzy Logic Controller (FLC) adjusts the sampling rate based on experiences in past situations with similar sample data. A rule structure is defined that the FLC can follow to determine the most appropriate action to take given a certain input condition.

2.2.1. Linear Prediction (LP) Method

The LP method proposed for adaptive sampling is based on the Linear Prediction Coefficient (LPC) technique used by Jacobson and Karels in the congestion control protocol for the Transport Control Protocol (TCP) to predict the Round Trip Time (RTT) of a packet [10]. LPC employs a low-pass filter to predict future data values. This technique filters out transient behavior, using the average value of the non-transient data as the prediction for the next value. This predicted time value was used to adjust the transmission window base on the RTT of previous packets. The low-pass filter is of the form:

$$x_p = (1-\alpha)\times M + \alpha \times R \tag{2.1}$$

In this equation, x_p is the predicted value for the next sample, M is the most recently measured value, and R represents the average of the previous two samples. The coefficient α ranges between zero and one and is selected experimentally depending on the network load, where higher values of α reduce the impact of transient values on the average value.

A similar LP-based technique is used in this paper to control the sampling time. The main difference between the approach described in this paper and the one used by Jacobson and Karels is that here a variable number of previous samples are used to calculate the predicted value. Instead of using only the previous two samples, a window of N samples is used for the prediction. Moreover, the α coefficient for tuning to a specific network load is no longer employed.

Eq. (2.2) below defines the LP logic for a sampler of order N. As in Eq. (2.1), the value x_p represents the predicted value for the next sample. The vector x holds the value of the previous N samples, where $x[N]$ is the most recent sample and $x[1]$ is the oldest sample. *Sample* refers to the value of the current sample. Immediately prior to taking the next sample, the values in the vector x are shifted such that $x[1]$ is discarded and replaced with $x[2]$, $x[2]$ is replaced with $x[3]$, and so forth with the value of *Sample* replacing $x[N]$. Eq (2.2) is designed to work with non-decreasing SNMP byte counters, and therefore the difference between any two values in x will always be non-negative. A second vector, t, records the time that each sample is taken and is shifted in the same manner as x, with the time at which *Sample* was taken replacing $t[N]$. Since the period between samples is not necessarily constant, the next value is predicted based on the average rate of change in the previous N samples.

$$x_p = x[N] + \frac{\Delta T_{Current}}{N-1} \sum_{i=1}^{N-1} \left(\frac{x[i+1] - x[i]}{t[i+1] - t[i]} \right) = x[N] + \frac{\Delta T_{Current}}{N-1} \left(\frac{x[N] - x[1]}{t[N] - t[1]} \right) \tag{2.2}$$

The prediction is then used as shown in Fig. 2. The predicted output, x_p, which has been derived from the previous N samples, is then compared with the actual value of *Sample*. A set of rules is applied to adjust the current sampling interval, $\Delta T_{Current} = t[N] - t[N-1]$, to a new value, ΔT_{Next}, which is used to schedule the next management query. The rules used to adjust the sampling interval compare the rate of change in the predicted sample value, $x_p - x[N]$, to the actual rate of change, $Sample - x[N]$. This rate, m, is given by Eq. (2.3).

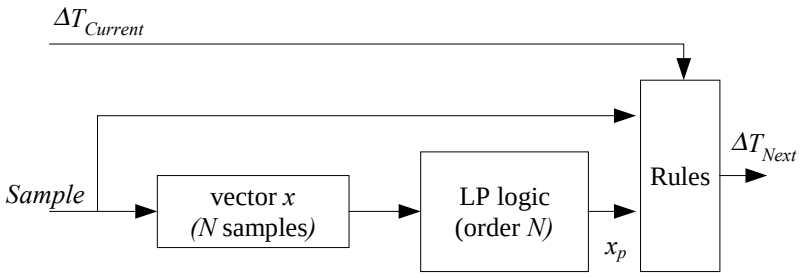


Fig. 2. LP-based adaptive sampling.

$$m = \left| \frac{x_p - x[N]}{Sample - x[N]} \right| \tag{2.3}$$

The rate of change provided by Eq. (2.3) will take on a value near unity when the predicted behavior is close to the actual behavior. The range of values that satisfy this condition is defined as $m_{min} < 1 < m_{max}$. If m is below m_{min}, the measured parameter is changing faster than the prediction. Such behavior indicates more activity than predicted, so the sampling interval should be decreased to yield more accurate data values on which to base future predictions. Conversely, if m is above m_{max}, the measured value is changing more slowly than the prediction so the sampling interval can be increased. Note that m can be undefined if the current value of *Sample* is unchanged from that of the previous sample, $x[N]$. Since this condition is indicative of an idle network, the sampling interval is increased exponentially.

Table I. LP rules for adjusting sample interval.

Calculated m Value	Next Sample Interval
$m < m_{min}$	$\Delta T_{Next} = m \times \Delta T_{Current}$
$m_{min} \leq m \leq m_{max}$	$\Delta T_{Next} = \Delta T_{Current}$
$m_{max} < m$	$\Delta T_{Next} = \Delta T_{Current} + 1$ sec
m undefined	$\Delta T_{Next} = 2 \times \Delta T_{Current}$

Table I lists the specific rules for generating ΔT_{Next} given the current value of m. Based on the results of experiments described later, m_{min} and m_{max} are set to 0.9 and 1.1 respectively. These values were selected because they provided good performance over a range of traffic types. An additional constraint can be used to limit the range of possible values for ΔT_{Next}. For example, in this paper, the sampling interval is restricted to be between 1 and 10 seconds regardless of the output of the LP logic. The lower bound is applied to limit the sampling to a rate the manager and agents can realistically service, while the upper bound ensures there is some minimum set of samples on which to base future predictions.

2.2.2. Fuzzy Logic Controller (FLC) Method

Rather than attempting to predict the exact values of future samples, fuzzy logic applies "rules of thumb" to the observed behavior to adjust the future sampling rate in a reasonable fashion. Fuzzy logic has shown promising results in non-linear systems, especially those that are difficult to model or where an exact model is impossible. The FLC mimics human reasoning by applying a set of rules based on one or more premises and a single implication. For example, if the premises "network load is very high" and "the sampling interval is somewhat low" are found to be true, the implication "reduce the sampling interval by a medium amount" might be taken. In this section, an FLC is used to adaptively control the sampling rate of the network monitor.

The proposed method for adaptive sampling using fuzzy logic is shown in Fig. 3. The inputs of the FLC are the current sampling interval, $\Delta T_{Current}$, and the difference between the last two sample values, ΔX. The FLC output, F_{out}, corresponds to the amount of time to increment or decrement the current sampling interval, $\Delta T_{Current}$. This adjustment yields a new value for the next sampling interval, ΔT_{Next}. The following subsections describe the FLC, including the definition of the membership functions, the fuzzy rules, and the defuzzification process.

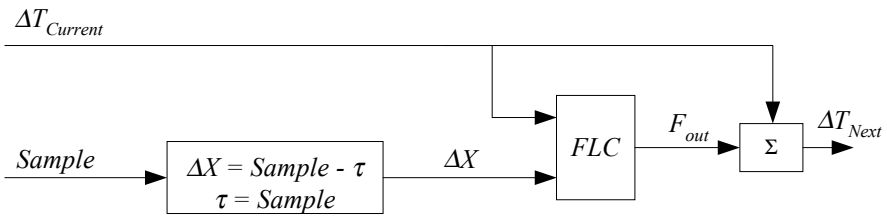
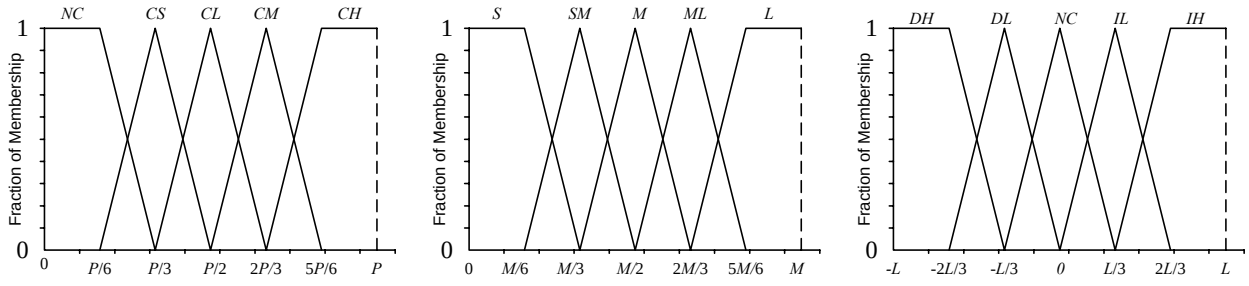


Fig. 3. Adaptive sampling based on fuzzy logic.

2.2.3. FLC Membership Functions

Unlike traditional digital systems, fuzzy logic differentiates between several levels such as "high," "medium," and "low" when making decisions. The numerical ranges assigned to each of the levels are defined in a membership function. The FLC makes use of a set of membership functions such as those illustrated in Fig. 4 to determine a suitable output value given the state of the input values. In this case, there are two membership functions for the inputs and one for the output. For the two input membership functions, ΔX and $\Delta T_{Current}$, the horizontal axis corresponds to the value of the input. A given value of the input is interpreted as being in one or more fuzzy states. For the ΔX input, the fuzzy states are *No-Change (NC), Change-Slight (CS), Change-Low (CL), Change-Medium (CM), and Change-High (CH)*. The $\Delta T_{Current}$ input falls into the categories of *Small (S), Small-Medium (SM), Medium (M), Medium-Large (ML),* and *Large (L)*.

The output of the FLC is also defined in terms of fuzzy variables. In Fig. 4, the output of the FLC, F_{out}, falls into one of five fuzzy states: *Decrease-High (DH), Decrease-Low (DL), No-Change (NC), Increase-Low (IL),* and *Increase-High (IH).*



(a) ΔX (b) $\Delta T_{Current}$ (c) F_{out}

Fig. 4. Membership functions used by the FLC.

In Fig. 4, the labels at the top of each peak in the graph represent the fuzzy states for the given input or output variable. The y-axis shows the fractional membership in each state for a given value along the x-axis. The x-axis value corresponds to the numerical value of the input or output variable. The membership function for input ΔX describes the amount of change in measured throughput (in terms of SNMP byte-count) between successive samples and ranges from 0 to P bytes. Similarly, the membership function for input $\Delta T_{Current}$ denotes the current sampling interval and ranges from 0 to K seconds. Finally, the membership function for output F_{out} indicates the amount of increase or decrease to be applied to the sampling interval and ranges from -L to +L seconds.

Table II. Parameter values selected for the membership functions in the FLC.

Type of Traffic	P (bytes)	K (sec)	L (sec)
Internet	$0.4 \times S_{Internet}$	12	2.0
Videoconference	$4.0 \times S_{video}$	12	0.5

The appropriate selection of the membership parameters P, K, and L requires some understanding of the traffic behavior. When determining the values for these FLC parameters, it is desirable to select values that lead to a relatively uniform spread in frequency of occurrence of inputs across the states of the membership function. For instance, with parameter P, the goal is select the value that distributes the input data in a relatively uniform fashion across all five of the stages, from *NC* to *CH*. Based on several tuning experiments with the Internet and videoconference traffic models described in the next section, the values selected for this controller were determined as shown in Table II. In this table, the variables $S_{Internet}$ and S_{video} refer to the peak throughputs (i.e. the maximum change in SNMP transmitted byte-count measured over a span of one second) in the Internet and videoconference data, respectively. These values are multiplied by a scaling factor to produce the desired value for parameter P. For Internet traffic, a small scaling factor is selected, which allows the FLC to respond to smaller changes in network behavior. Coupled with a larger value of L, the FLC can make large adjustments in the sampling interval in response to small changes in network behavior. The videoconference traffic employs a larger scaling factor and smaller value of L since the periodic nature of the traffic requires less drastic adjustments to sample rate. Although beyond the scope of this paper, the use of an FLC in a network with arbitrary traffic would require a training period during which the optimal spread of these three parameters across the fuzzy states in the membership functions could be ascertained.

2.2.4. FLC Rules

In addition to the membership functions for the inputs, the FLC needs a fuzzy set of rules to map the input values to an output response. The twenty-five statements shown in Table III represent a proposed set of fuzzy rules for the FLC. Each row in this table provides two *premises* along with a single *implication*. In the table, the first and second columns are correlated using the logic operator AND, which is analogous to the intersection in set theory. For example, the fuzzy logic expression shown in the first row of the table indicates that given no measured changes in the input, ΔX, and a small sampling interval, $\Delta T_{Current}$, then F_{out} should cause the sampling interval to be increased by a high amount.

Table III. Rules for the fuzzy controller.

Rule	ΔX		$\Delta T_{Current}$		F_{out}	
1	*No-Change*	*(NC)*	*Small*	*(S)*	*Increase-High*	*(IH)*
2	*No-Change*	*(NC)*	*Small-Medium*	*(SM)*	*Increase-High*	*(IH)*
3	*No-Change*	*(NC)*	*Medium*	*(M)*	*Increase-Low*	*(IL)*
4	*No-Change*	*(NC)*	*Medium-Large*	*(ML)*	*Increase-Low*	*(IL)*
5	*No-Change*	*(NC)*	*Large*	*(L)*	*No-Change*	*(NC)*
6	*Change-Slight*	*(CS)*	*Small*	*(S)*	*Increase-High*	*(IH)*
7	*Change-Slight*	*(CS)*	*Small-Medium*	*(SM)*	*Increase-Low*	*(IL)*
8	*Change-Slight*	*(CS)*	*Medium*	*(M)*	*No-Change*	*(NC)*
9	*Change-Slight*	*(CS)*	*Medium-Large*	*(ML)*	*Decrease-Low*	*(DL)*
10	*Change-Slight*	*(CS)*	*Large*	*(L)*	*Decrease-Low*	*(DL)*
11	*Change-Low*	*(CL)*	*Small*	*(S)*	*Increase-Low*	*(IL)*
12	*Change-Low*	*(CL)*	*Small-Medium*	*(SM)*	*No-Change*	*(NC)*
13	*Change-Low*	*(CL)*	*Medium*	*(M)*	*Decrease-Low*	*(DL)*
14	*Change-Low*	*(CL)*	*Medium-Large*	*(ML)*	*Decrease-High*	*(DH)*
15	*Change-Low*	*(CL)*	*Large*	*(L)*	*No-Change*	*(NC)*
16	*Change-Medium*	*(CM)*	*Small*	*(S)*	*Decrease-Low*	*(DL)*
17	*Change-Medium*	*(CM)*	*Small-Medium*	*(SM)*	*Decrease-Low*	*(DL)*
18	*Change-Medium*	*(CM)*	*Medium*	*(M)*	*Decrease-High*	*(DH)*
19	*Change-Medium*	*(CM)*	*Medium-Large*	*(ML)*	*Decrease-High*	*(DH)*
20	*Change-Medium*	*(CM)*	*Large*	*(L)*	*Decrease-High*	*(DH)*
21	*Change-High*	*(CH)*	*Small*	*(S)*	*Decrease-Low*	*(DL)*
22	*Change-High*	*(CH)*	*Small-Medium*	*(SM)*	*Decrease-High*	*(DH)*
23	*Change-High*	*(CH)*	*Medium*	*(M)*	*Decrease-High*	*(DH)*
24	*Change-High*	*(CH)*	*Medium-Large*	*(ML)*	*Decrease-High*	*(DH)*
25	*Change-High*	*(CH)*	*Large*	*(L)*	*Decrease-High*	*(DH)*

The "fuzziness" of the FLC stems from the fact that a given input value may correspond to multiple fuzzy states. For example, referring to Fig. 4(a), if ΔX has a value between $P/3$ and $P/2$ then it would be a partial member in the *CS* state and a partial member in the *CL* state. Such fuzziness makes it hard to determine a fixed expression for the output value from Table III. In order to find the non-fuzzy output, the defuzzification process takes place. This process is composed of an inference method that makes use of the rules and membership functions to produce the control output. The correlation-product method is used here for defuzzification with the FLC [3].

2.2.5. FLC Defuzzification Process

Consider the example in Fig. 5 using the correlation-product method for defuzzification. The figure assumes the input values of ΔX_0 and ΔT_0 for ΔX and $\Delta T_{Current}$, respectively. The membership functions indicate two degrees of membership for each input: a value of α in the *CS* state and β in the *NC* state for the input ΔX_0, and a value of δ in the *ML* state and ε in the *M* state for T_0. There are three steps in the defuzzification process using the correlation product method. The first step is to find the set of all rules that correspond to the input membership states by considering all permutations of the states. In this example, the rules 3, 4, 8, and 9 from Table III apply. The second step is to choose the minimum value of the degree of membership of the inputs among the matching rules. For instance, rule 3 involves the fuzzy states *NC* and *M* in the membership functions ΔX and $\Delta T_{Current}$, respectively. The degree of membership for ΔX_0 is β and for ΔT_0 is ε. Since $\varepsilon < \beta$, then ε is the minimum input value for rule 3. The third step scales the shape of the membership function for the F_{out} value of each rule by the minimum input value determined in the previous step. In the example below, rule 3 defines the *IL* state for F_{out}, so the *IL* shape is scaled down by ε. This process is repeated for all the rule graphs and the resulting scaled shapes are combined. The center of mass of the combined graph is calculated using Eq. (2.4) [3,11]. The resulting outcome, z, corresponds to the numerical value used for F_{out}.

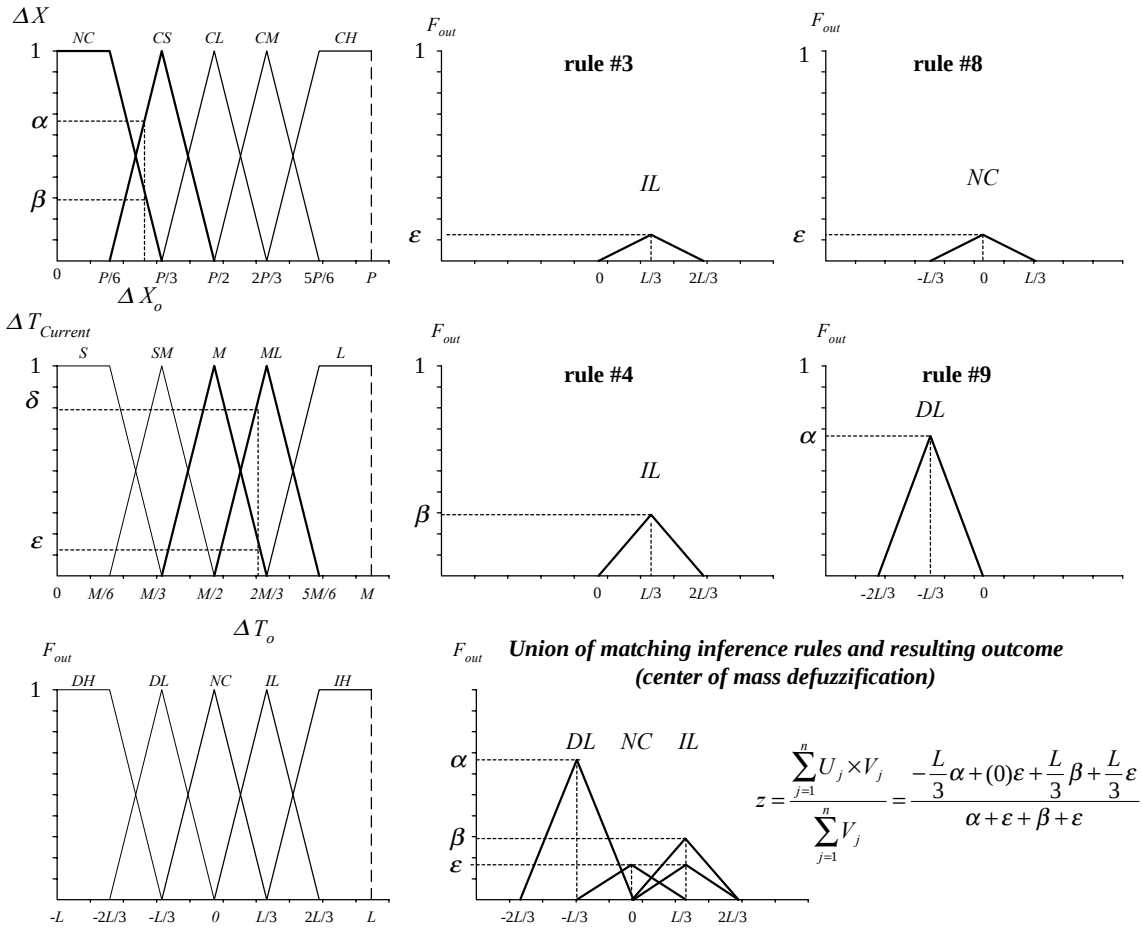


Fig. 5. Correlation-product inference method and defuzzification process.

$$z = \frac{\sum_{j=1}^{n} U_j \times V_j}{\sum_{j=1}^{n} V_j} \tag{2.4}$$

For each of the *n* scaled shapes in the combined graph, the center of mass equation uses U_j and V_j to represent the values for the peak of the shape on the x-axis and y-axis, respectively. For example, Fig. 5 shows the combination of the scaled rule graphs where the U_j values are $L/3$, $L/3$, 0, and $-L/3$, and the V_j values are ε, β, ε, and α, from rules 3, 4, 8, and 9, respectively. The result of this equation represents the numerical (i.e. non-fuzzy) output for membership function F_{out}. Future sampling queries are rescheduled based upon this scalar result as shown in Fig. 3.

3. EXPERIMENTS AND MEASUREMENT TECHNIQUES

This section describes a set of simulative experiments to compare the adaptive and systematic sampling techniques for network management. The computing testbed that generated the traffic in the simulation is described in this section, as well as the metric used to compare the sampled data sets.

3.1. Traffic Models and Sampler Simulation

Two traffic traces of network activity are used as data models for the experiments. The two were selected to represent the opposite ends of the spectrum of potential traffic that might be encountered in a high-performance network, one characterized by bursty, aperiodic traffic and the other by streaming, periodic traffic.

The first trace is a two-hour sample taken from the outgoing 100 Mb/s Fast Ethernet port on a Fore Systems PowerHub 7000 router providing Internet connectivity to a laboratory with dozens of computers during a busy part of the day. This *Internet* trace is representative of random, bursty TCP/IP network data and includes traffic from Hypertext Markup Language (HTML), File Transfer Protocol (FTP), and Telnet activity. This type of data should be ideally suited to adaptive sampling because the sample rate can be adjusted to compensate for the level of loading on the network at any given time. Thus, the goal with adaptive sampling for this type of traffic is to reduce the sample count while maintaining the accuracy achieved by the equivalent systematic sampler. Or, conversely, the goal is to increase the accuracy while maintaining the same sample count as the equivalent systematic sampler.

A second trace was captured from a port of a Fore Systems ASX-200BX Asynchronous Transfer Mode (ATM) switch that was connected directly to a Fore Systems AVA-300 ATM audio/video decoder. This connection hosts a continuously running videoconference feed transferring variable-bit-rate compressed video and uncompressed audio at 30 frames/second over a dedicated 155 Mb/s ATM link. The resulting *videoconference* trace is representative of a highly periodic pattern of network traffic. The periodic nature of this data may not be better suited to adaptive sampling techniques than the systematic sampler. However, the adaptive sampling algorithm can dynamically select a sampling interval that accurately measures the data. By contrast, a systematic technique requires the proper interval to be correctly selected *a priori* by the network administrator, which is not easily realized.

Each trace was taken by measuring the cumulative number of bytes received over a two-hour period using SNMP queries. The resolution of the measurements was 0.1 seconds, yielding a total of 72,000 samples. The adaptive and systematic sampling methods were then simulated and evaluated by resampling the data in each trace.

For example, a systematic sampling with T_s = 1 sec would select every tenth element from the original data, yielding a new trace with 7,200 samples. During the simulation, all sampling intervals were rounded to the nearest 0.1 seconds so that they correspond to a single measurement from the original trace. For simplicity, the throughput measurements gathered by the systematic and adaptive samplers with these traces are all normalized to the peak value stored in the baseline data model (i.e. either Internet or videoconference) used in the experiment.

3.2. Metric for Comparison

In order to compare the performance of the adaptive sampling techniques with the systematic baseline, a measure of accuracy is needed. The *sum of squared error* metric [2] for comparing two *N*-sample sets, as shown in Eq. (3.1), is used in Section 4.3 to compare the accuracy of the different techniques. This expression makes a point-by-point comparison between the reference and sampled signals using the normalized magnitude of the instantaneous throughput. The reference signal, $f_R(n)$, is a systematic sampling of the original trace of Internet or videoconference data with a period of T_s = 1 sec. The comparison signal, $f_C(n)$, is the output from one of the adaptive or systematic sampling methods applied to the original trace. Since Eq. (3.1) requires sets with an equal number of samples, both the reference and the comparison traces are resampled to 72,000 samples. Linear interpolation is used to produce the added points.

$$Error = \sum_{n=1}^{N} (f_R(n) - f_C(n))^2 \tag{3.1}$$

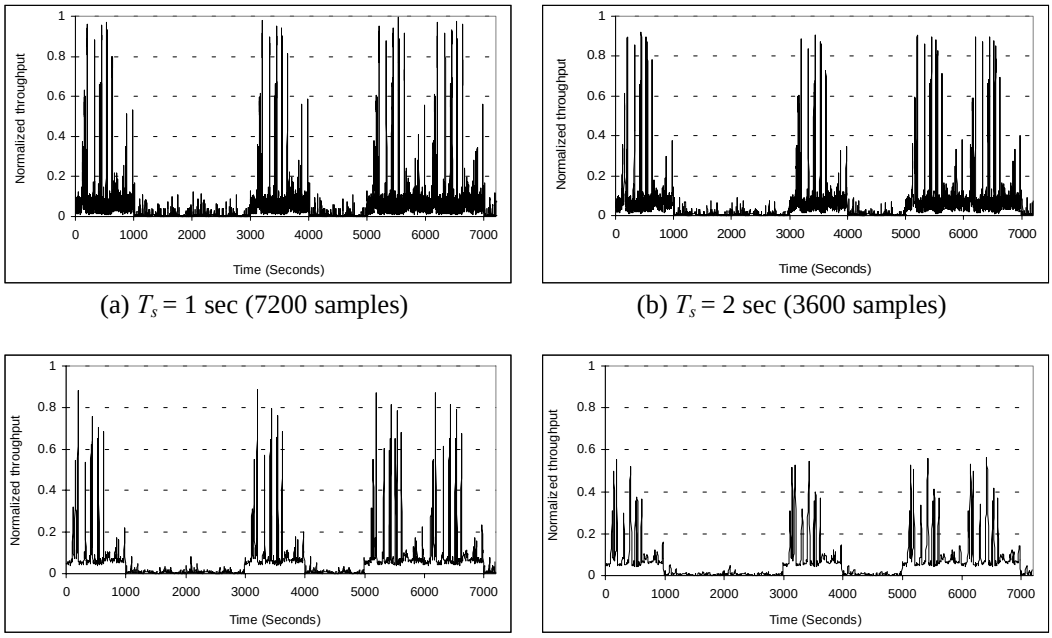
4. RESULTS AND ANALYSES

This section presents results and analyses from experiments conducted to ascertain the performance of the systematic and adaptive sampling algorithms as applied to the Internet and videoconference traces. In the next two subsections, a qualitative comparison of the signals produced by several systematic and adaptive sampling measurements is provided. Then, a quantitative assessment of the adaptive sampling techniques is conducted by comparing the adaptive techniques to an equivalent systematic baseline in terms of sample count and mean-squared error relative to the reference trace.

4.1. Systematic Sampling Measurements

In this section, the effect of sampling rate is qualitatively illustrated using traditional, systematic sampling techniques. Fig. 6 and Fig. 7 show the normalized throughput sampled using fixed sampling intervals (T_s) from one to ten seconds on the Internet and videoconference traces, respectively. The shapes for Internet and videoconference traffic at T_s = 1 sec shown in Fig. 6(a) and Fig. 7(a) are used as the reference value in the comparisons of the different sampling disciplines.

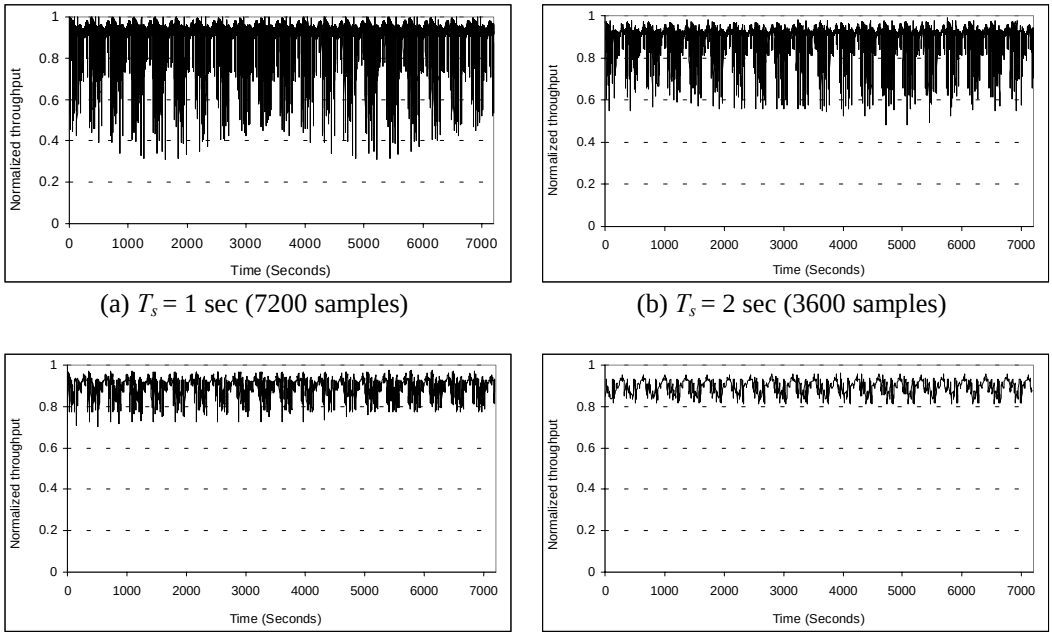
Both figures demonstrate a "smoothing" effect at larger sampling intervals. This effect results from the fact that the samples actually measure a cumulative SNMP byte-count rather than a throughput. Therefore, at higher values of T_s, the measured byte-count is averaged over a longer interval of time. The result is that large throughput spikes are averaged with periods of low activity. This effect both reduces the magnitude of the throughput spikes as well as filters much of the variation from the actual traffic pattern.



(a) T_s = 1 sec (7200 samples) (b) T_s = 2 sec (3600 samples)

(c) T_s = 5 sec (1440 samples) (d) T_s = 10 sec (720 samples)

Fig. 6. Measured traffic patterns of Internet data using systematic sampling.



(a) T_s = 1 sec (7200 samples) (b) T_s = 2 sec (3600 samples)

(c) T_s = 5 sec (1440 samples) (d) T_s = 10 sec (720 samples)

Fig. 7. Measured traffic patterns of videoconference data using systematic sampling.

The measurements conducted at T_s = 10 sec for the Internet data in Fig. 6 indicate a maximum throughput of only about 55% of the peak value at the reference. Fig. 7 illustrates a similar situation with videoconference traffic. For example, in the trace sampled at T_s = 5, the highly fluctuating behavior found at the reference is completely lost. In fact, the variation observed in Fig. 7(c) is only approximately 25%, whereas the reference indicates a variation of 50-70% of the maximum peak throughput.

Since high-performance management services require precise throughput estimations, an inappropriate selection of the sampling time would lead to inaccurate observations. The following section will show how adaptive sampling can be used to maintain accurate measurements for Internet and videoconference traffic.

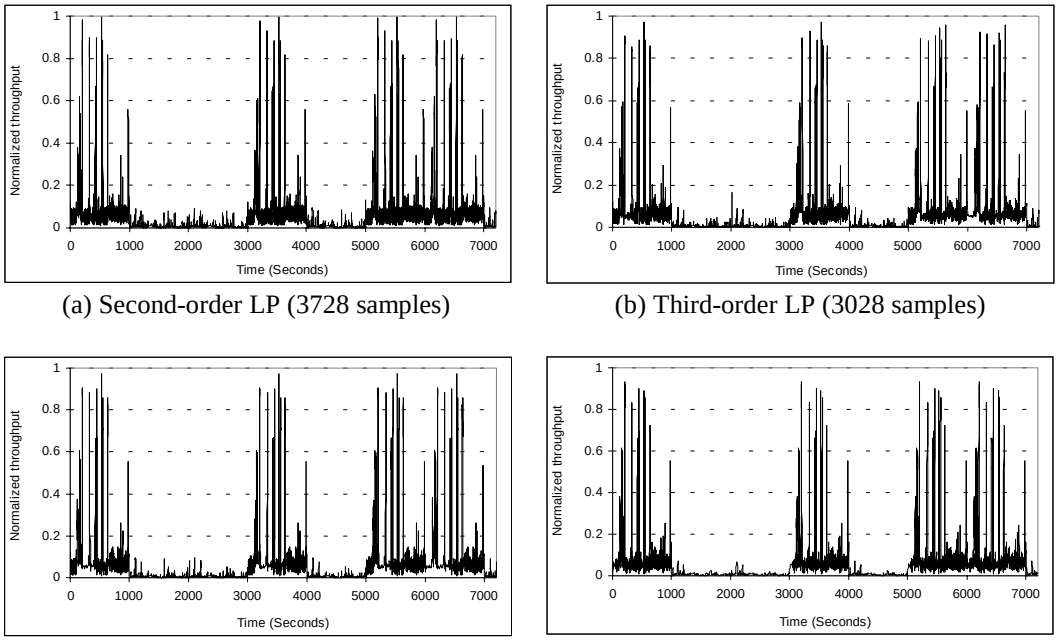
4.2. Adaptive Sampling Measurements

The results of applying the adaptive sampling techniques to the Internet trace using second-, third-, and fourth-order LP samplers and the FLC sampler are shown in Fig. 8. The second-order LP sampler was able to reduce the sample count to about 51% of the reference with almost no discernable change in the peak magnitude or variation of throughput behavior. As the order of the LP sampler is increased, the number of samples are further reduced with

only a small visible loss in peak magnitude and variation. The FLC approach yields the fewest number of samples at 29% of the reference. The FLC shows more variation than the higher-order LP samplers but with marginally lower peaks.

Fig. 9 shows the results of applying the adaptive sampling techniques to the periodic videoconference trace. The LP techniques reduce the number of samples to approximately 65% of the reference but at a slight reduction in signal variation. The FLC achieves shows slightly less variation at 43% of the reference sample count. However, the accuracy of the adaptive sampling techniques on the videoconference data is not nearly as good as it is with the Internet data.

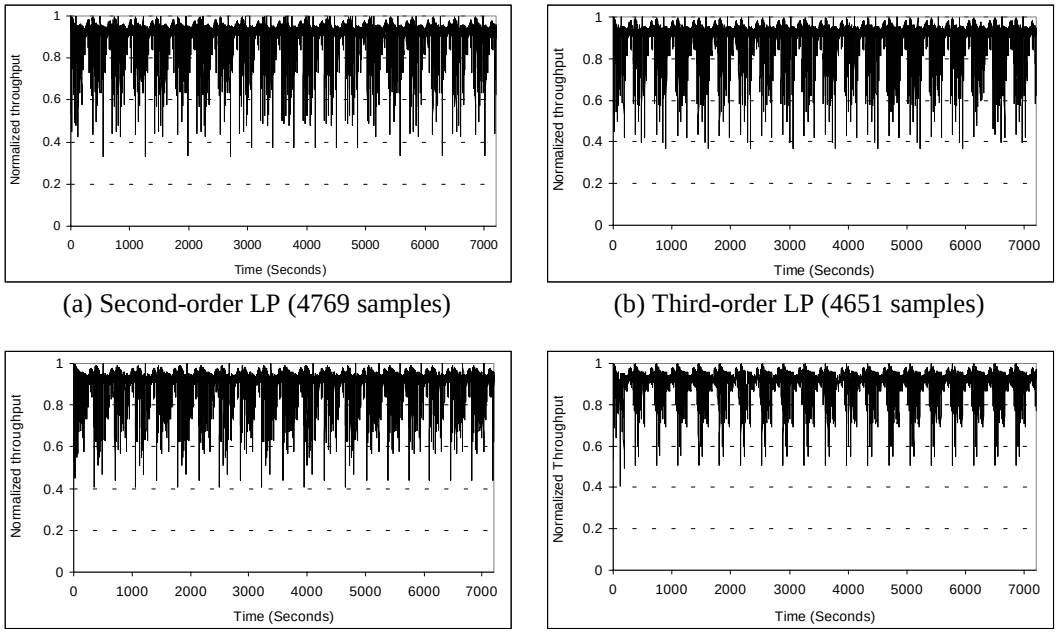
Although a qualitative comparison of the adaptive sampling techniques shows the ability of the adaptive sampling techniques to reduce sample count while retaining a certain degree of accuracy, an accurate picture of the performance is difficult to ascertain. For example, a simple systematic sampler with T_s = 2 sec is also able to reduce the number of samples by 50% while still retaining much of the characteristics in the reference signal. Therefore, it is necessary to quantify the accuracy of the adaptive and systematic sampling techniques.



(a) Second-order LP (3728 samples) (b) Third-order LP (3028 samples)

(c) Fourth-order LP (2641 samples) (d) FLC (2095 samples)

Fig. 8. Measured traffic patterns of Internet data using adaptive sampling.



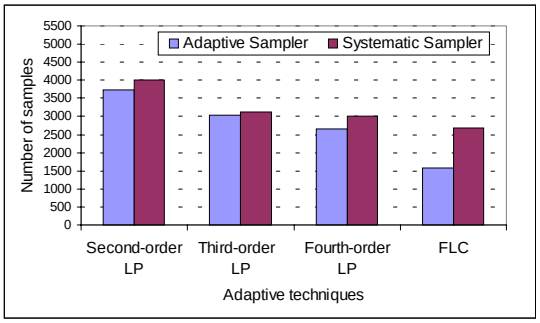
(a) Second-order LP (4769 samples) (b) Third-order LP (4651 samples)

(c) Fourth-order LP (4621 samples) (d) FLC (3095 samples)

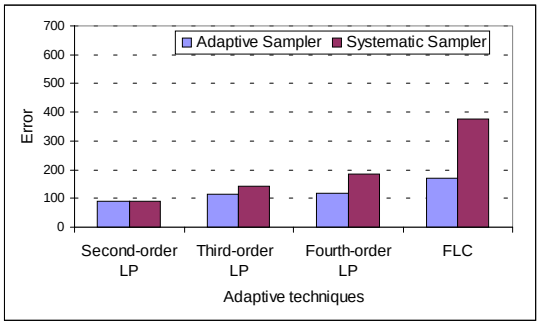
Fig. 9. Measured traffic patterns of videoconference data using adaptive sampling.

4.3. Quantitative Analysis of Adaptive Samplers

It is difficult to quantitatively compare the performance of the adaptive sampling techniques to an equivalent systematic approach when the measurements differ in both the number of samples and the squared-error relative to the reference. In general, traces with more samples show a lower error. Therefore, two forms of experiments were conducted to compare the adaptive and systematic sampling methods, first with a fixed number of samples and then with a fixed error. In both cases, a single run of the adaptive sampling techniques was used. As demonstrated in the previous section, each technique yielded a different number of samples with a different relative error when compared to the reference. In the fixed sample-count comparison, a systematic sampling interval was selected for each adaptive sampling result such that it would contain the same number of samples. The relative error from each adaptive sampler was then calculated and compared to its equivalent systematic sampler. For the constant-error comparison, a chart of sample count versus relative error was made for a range of systematic sample intervals. Using this chart, the minimum number of systematically distributed samples necessary to achieve a certain relative error was determined.



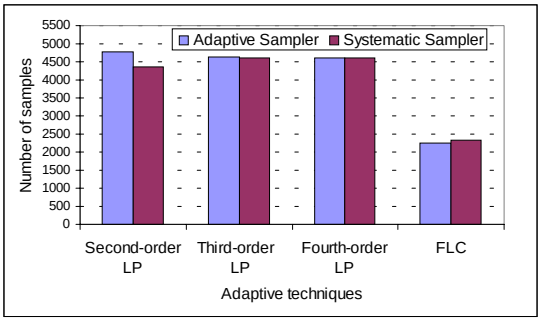
(a) Sample count with constant error



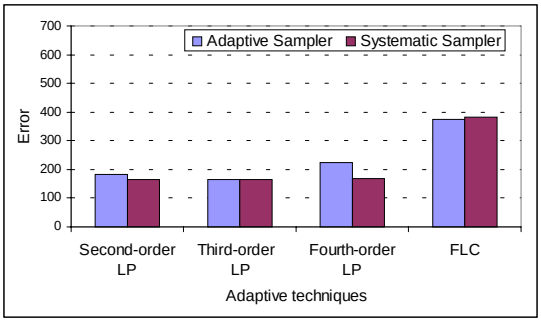
(b) Error with constant sample count

Fig. 10. Sample count and error variation with Internet traffic.

Fig. 10 shows the results of the experiments conducted with Internet traffic. In Fig. 10(a), the amount of sampling error is held constant and then the sample count is measured for each of the adaptive samplers and compared to a systematic sampler with the same number of samples. In Fig. 10(b), the sample count is held constant and the error is measured for each case. As the results indicate, each of the four adaptive samplers outperforms its corresponding systematic sampler. The greatest improvement is seen with the FLC sampler. The FLC sampler achieves nearly twice the performance of its equivalent systematic sampler, which can be interpreted either as decreasing the sample count by a factor of two for a given level of accuracy or as increasing the accuracy by a factor of two for a given number of samples. In contrast, the second-order LP sampler is only marginally better in performance than its systematic counterpart. As previously hypothesized, traffic of a bursty, aperiodic nature is found to be well suited for the capabilities of adaptive sampling, since these samplers are able to dynamically adjust their sampling rate with periods of increased and decreased activity on the network.



(a) Sample count with constant error



(b) Error with constant sample count

Fig. 11. Sample count and error variation with videoconference traffic.

Fig. 11 shows the results of the same experiments with the videoconference traffic. As before, Fig. 11(a) displays the measurements of sample count for a fixed level of accuracy and Fig. 11(b) displays the measurements of accuracy for a fixed number of samples. The results in these experiments with videoconference traffic are markedly different from those with the Internet traffic. The LP samplers show slightly higher error or slightly larger sample counts than their systematic counterparts. The FLC sampler achieves slightly lower sample counts and slightly smaller error rates. Given the nature of the traffic being sampled, these results are also promising. When dealing with traffic of a periodic nature, as is the case with the videoconference, systematic sampling is effective provided that the appropriate sampling interval can be identified and employed. Thus, the primary goal of using an adaptive sampler for periodic traffic is to match the performance of the best systematic sampling rate. This performance is achieved and even surpassed in results from the experiments with videoconference data by using the FLC sampler, and the LP samplers show only a marginal degradation in performance. An advantage of adaptive sampling is that this optimal rate can be dynamically adjusted if the frequency or type of periodic traffic changes.

Taken together, the results of the experiments with Internet and videoconference traffic confirm that adaptive sampling can play an important role in decreasing the load on the network and the network manager, increasing the accuracy of the measurements, or both. While these two traffic models do not of course encompass the entire universe of possible traffic patterns, they are reasonably representative of the boundaries of that universe in terms of bursty, aperiodic traffic versus streaming, periodic traffic. Thus, these results illustrate that adaptive sampling provides the potential for better monitoring, control, and management of high-performance networks.

5. RELATED WORK

Several researchers have studied sampling techniques in networks. Claffy et al. studied three non-adaptive sampling methodologies: systematic (i.e. periodic), random, and stratified random sampling [4]. Their results indicated that stratified random sampling has better accuracy when the three methods are used to capture the same number of samples. Cozzani and Giordano made use of conventional sampling methods and studied their effects in quality of service measurements for ATM networks [5]. Drobisz proposed a network capture device for Gigabit Ethernet, adapting CPU utilization based on traffic burst anticipation by increasing CPU usage when a bursty period of traffic was expected [12].

Fuzzy and LP adaptations have been used in network applications for congestion, admission, and flow control. Jacobson and Karels studied flow-control protocols based on filters for the congestion control algorithm for TCP [10]. Kalampoukas et al. applied filter-based techniques for window adaptation in TCP [13]. These methods control protocol parameters based upon network traffic behavior. The adaptive sampling techniques presented herein employ a similar approach, but by contrast the purpose is for more efficiency in sampling of network management data with the goal of reducing sample count and/or increasing accuracy in a dynamic traffic environment.

Fuzzy-logic controllers have also been studied for ATM networks. Bonde and Ghosh [14] and Catania et al. [15] proposed queue managing and congestion control based on fuzzy sets. Cheng and Chang suggested a fuzzy architecture for both congestion and call admission control in ATM networks [11], and discovered that congestion control, using fuzzy logic, performed better than the leaky bucket algorithm inherent to ATM. The FLC-based sampling technique proposed herein uses an adaptation of the methods applied in these studies on control of ATM networks. Whereas these other studies used fuzzy logic to control parameters in the ATM network, our FLC-based adaptive sampling applies fuzzy logic to the monitoring of network traffic in any environment. Givan and Chong applied Markov and Bernoulli processes to model network traffic behavior and to create learning algorithms to predict future traffic measurements in a switch-based application [16]. By contrast, the adaptive techniques presented herein assume minimal knowledge of the traffic model, instead applying linear prediction and fuzzy-logic to adapt to any traffic pattern.

6. CONCLUSIONS

This paper has presented two techniques to adaptively monitor network behavior. One approach is based on using linear prediction to dynamically alter the sample rate based on the accuracy of the predictions, where inaccurate predictions indicate a change in the network's behavior and result in a smaller sampling interval. The second approach models the cognitive process of a human network manager by using fuzzy logic. When certain pre-determined conditions are met, such as an increase in network traffic, corresponding actions are taken such as a decrease in sampling interval.

These adaptive techniques are shown to perform well on random, bursty data such as conventional Internet traffic. All approaches are able to reduce the sample count while maintaining the same degree of accuracy as the

best systematic sampling interval. Equivalently, all approaches are able to increase accuracy while maintaining the same sample count. The higher-order LP samplers perform better than the lower-order ones, while the FLC sampler shows the greatest reduction in sample count. The reduced sample count is an important factor for network management in high-performance networks. Accurate measurements are required to find bottlenecks, while at the same time the impact on the network must be minimized to allow applications to take advantage of the low latency and high throughputs such networks provide.

For periodic data such as that found in a videoconference environment, the adaptive sampling approaches perform comparably to the best systematic approach in terms of accuracy and sample count. In particular, systematic sampling marginally outperforms the LP samplers while the FLC approach shows a slight improvement over systematic sampling. However, perhaps more importantly, the adaptive techniques have the ability to adjust to changing traffic loads, and therefore in an environment with dynamic traffic patterns, such as with data communications, multimedia and integrated services, adaptive sampling techniques hold the potential to outperform systematic sampling with its pre-selected and fixed sampling interval.

In general, the evidence suggests that the fuzzy-logic adaptive technique provides more flexibility and better performance than the LP methods. The main disadvantages of the FLC are the selection of the boundaries of the membership functions and the computational overhead required to implement it. Future work is needed to study methods to optimally tune the parameters of the adaptive samplers. One possible area of concentration is the development of an autonomous adaptive manager using fuzzy logic that dynamically adapts the parameters for its membership functions while the monitoring process is underway. Furthermore, the FLC could maintain a history of past samples as in the LP approach, at the cost of even more storage and computational overhead, but could potentially yield better results. In addition, other approaches for the design of adaptive samplers and management systems are worthy of investigation, such as samplers based on neural-network or neuro-fuzzy controllers.

ACKNOWLEDGEMENTS

This research was sponsored in part by the National Security Agency. A portion of this work was made possible by a Fulbright Foreign Graduate Student Fellowship from the Institute of International Education (IIE).

REFERENCES

1. S. Kay, *Modern Spectral Estimation – Theory and Application,* Prentice Hall, USA, 1988.

2. B. Widrow and S. Stears, *Adaptive signal processing*, Prentice Hall, USA, 1985.

3. E. Cox, *The Fuzzy Systems Handbook*, Second Edition, AP Professional, USA, 1999.

4. K. Claffy, H. Braun, and G. Polyzos, Application of Sampling Methodologies to Network Traffic Characterization, *Proceedings of ACM SIGCOMM '93*, San Francisco, California, pp. 194-203, September 1993.

5. I. Cozzani and S. Giordano, Traffic Sampling Methods for End-to-end QoS Evaluation in Large Heterogeneous Networks, *Computer Networks and ISDN Systems*, Vol. 30, No. 16-18, pp. 1697-1706, 1998.

6. J. Case, M. Fedor, M. Schoffstall, and J. Davin, *A Simple Network Management Protocol,* RFC1157, SNMP Research, May 1990.

7. M. Rose, *The Simple Book: An Introduction to Internet Management*, Second Edition, Prentice Hall, 1994.

8. S. Feit, *SNMP – A Guide to Network Management*, McGraw Hill, USA, 1995.

9. W. Stallings, *SNMP, SNMPv2, and CMIP. The Practical Guide to Network Management Standards*, Addison-Wesley, Reading, Massachusetts, USA, 1993.

10. V. Jacobson and M. Karels, Congestion Avoidance and Control, *Proceedings of the ACM SIGCOMM '88*, Stanford, California, pp. 273-288, August 1988.

11. R. Cheng and C. Chang, Design of a Fuzzy Traffic Controller for ATM Networks, *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, pp. 460-469, June 1996.

12. J. Drobisz, Adaptive Sampling Methods for High Speed Networks to Determine Traffic Statistics including the Hurst Parameter, *Master's Thesis*, University of South Florida, 1998.

13. L. Kalampoukas, A. Varma, and K. Ramakrishman, Explicit Window Adaptation: A Method to Enhance TCP performance, *Technical Report,* University of California Santa Cruz, UCSC-CRL-97-21, August 1991.

14. A. Bonde and S. Ghosh, A Comparative Study of Fuzzy versus 'Fixed' Threshold for Robust Queue Management in Cell-Switching Networks, *IEEE/ACM Transactions on Networking*, Vol. 2, No. 4, pp. 337-344, August 1994.

15. V. Catania, G. Ficili, S. Palazzo, and D. Panno, A Comparative Analysis of Fuzzy versus Conventional Policing Mechanisms for ATM networks, *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, pp. 449-459, June 1996.

16. R. Givan and E. Chong, Intelligent and Adaptive Management of Multi-class Networks, *Technical Report,* Purdue University, ARPA No. G371, April 1999.