FastCARS: Fast, Correlation-Aware Sampling for Network Data Mining

Jia-Yu Pan, Srinivasan Seshan, Christos Faloutsos

***Abstract*—Technology trends are making it more and more difficult to observe and record the large amount of data generated by high speed links. Traffic sampling techniques provide a simple alternative that reduces the volume of data collected. Unfortunately, existing sampling techniques largely hide any temporal relationship in the recorded data.**

Our proposed method, "FastCARS" naturally captures statistics for packets that are 1, 2 or more steps away. It has the following desirable properties: (a) it provides accurate measurements of full trace's statistics, (b) it is simple and can be easily implemented, (c) it captures correlations between successive packets, as well as packets that are further apart, and (d) it is scalable and flexible such that it can be easily adjusted to take into account prior knowledge about characteristics of particular traces.

We also propose several new tools for network data mining that use the information provided by *FastCARS* . The experimental results on multiple, real-world datasets (233Mb in total), show that the proposed *FastCARS* sampling method and these new data mining tools are effective. With these tools, we show that the independence assumption of packet arrival is not correct, and packet train may not be the only cause of dependence among arrivals.

***Index Terms*—Traffic analysis, sampling**

I. Introduction

The ability to monitor and characterize network traffic has proven to be critical to the design and operation of today's networks. However, as links have gotten faster and faster, it has become more difficult to observe key traffic characteristics or record packet data in real-time. Already, most network monitors rely on sampling techniques [1] [2] to provide measurement of high speed links. The ability of these sampling techniques to support different observations will prove crucial to the development of tomorrow's networks.

Today's sampling techniques are targeted towards enabling tasks such as usage-based billing, capacity planning and network research. These techniques can typically answer questions about the traffic such as: *What is the distribution of packet sizes on this link? What destinations are popular? How long are typical connections? or What applications contribute the most traffic?* However, a significant weakness of existing schemes is that they do not answer questions about the temporal correlation of the traffic. For example, some interesting traffic characteristics include: How is the arrival time of packet n related to packet $(n+1)$ or $(n+2)$? Is there correlation among arrivals? In the past, analysis of such packet content characteristics [3] and arrival correlation [4], using full (unsampled) packet traces, have led to the discovery of important phenomena such as "packet trains", a set of sequential packets that have the same source/destination IP addresses and port numbers, and self-similar traffic pattern. Clearly, such traffic characteristics are critical to the design of routers, routing algorithms and caching techniques. It is necessary that these kind of analysis can be done on sampled data, rather than on full trace.

We would like a sampling method that is **Informative** and **Efficient**. It should provide sufficient information for accurate estimates of both average and temporally correlated statistics. It should also be simple and require low computation when implemented on routers. To achieve these objectives, we propose a a new method, Fast, Correlation-Aware Sampling (*FastCARS*), to do sampling and data mining on router traffic. We show that our method can support the traditional uses of network sampling (provide interarrival time distribution), as well as statistics about the traffic at different correlation scales which can be used for further data mining on the sampled traffic. In particular, we use *FastCARS* to detect the presence of packet trains and to explore the independence of interarrival time. We show that packet interarrival times are not independent and packet train may not be the only cause of dependence among arrivals.

This paper is organized as follows. We summarize previous work in Section II. Section III describes our technique. Section IV presents results from using our sampling. Finally, we present our conclusions in Section V.

II. Related Work

Network sampling has played an important role in network measurements for the past decade. In order to describe the desirable properties of a sampling techniques, we define the term *step*.

Definition 1: We call the separation between samples **steps**. A n-step histogram is a histogram of measurements obtained from pairs of sampled packets that are n-steps away (separated by $(n-1)$ packets). Histograms of different steps provide aggregated statistics which reveal short and long term correlations. Given a histogram of samples from a distribution f, f can be represent approximately by the sample frequency from the histogram.

Statistics of samples n steps apart are prefixed by the term **n-step**. For example, interarrival time between back-to-back packets is an instance of **1-step interarrival time**. We will show in the following sections that histograms of multiple steps give us information about the traffic characteristics, and reveal temporal correlation between packets. For example, the 1-step histogram can be used to estimate packet interarrival time distribution, and other histograms can be used for other data mining tasks. For example, we will show that the 2-step histogram can be used to explore the independence of interarrival times. One important property for a sampling technique is that it be

correlation-aware, that is, it should provide statistics for n-step histograms for arbitrary values of n.

We categorize the more promising past techniques into four categories: event-driven sampling, random sampling, configured run-length sampling and back-to-back sampling. Each of the existing sampling methods has its merits. However, none of them successfully satisfies all the requirements outlined in section I. The following definitions describe these techniques.

Definition 2: The deterministic (systematic) **event-driven sampling method** with sampling period p samples packets numbered 0, p, $2p$ and so on, periodically. It is denoted as **Event(**p**)**. This is the method implemented in Cisco's router. Note that Event(p) is a special case of *FastCARS*, and it is equivalent to *FastCARS*(p).

Definition 3: The **random sampling method** is a variant of the event-driven sampling method where its sampling interval is not fixed but a random variable following a specific distribution.

Definition 4: The **configured run-length sampling method** with sampling period p and run length q is denoted as **ConfRL(**p**,**q**)**. It samples a sequence of q events in every sampling cycle. cycle. That is, if the sampling starts on packet 0, then for the $(k+1)$-th sampling cycle ($k \geq 0$), sampling method ConfRL(p,q) will sample packets numbered kp, $(kp+1)$, ..., $(kp+q-1)$.

Definition 5: The **back-to-back sampling method** with sampling period p samples packets numbered $0, 1, p$, $(p+1)$, $2p$, $(2p+1)$ and so on. It is denoted as **back-to-back(**p**)**. Note that back-to-back sampling method is a special case of configured run-length sampling method. That is, back-to-back(p) is equivalent to ConfRL(p,2). In general, a back-to-back sampling with sampling period p and step s, denoted as **back-to-back(**p**,** s**)**, which samples packets numbered 0, s, p, $(p+s)$, $2p$, $(2p+s)$ and so on.

These different techniques have been evaluated in past work and are used in both past and current products. Claffy et al. [5] evaluated both *event-driven* and *time-driven* sampling methods. In this study, the measured statistics were packet interarrival times and packet sizes. To measure interarrival time, the sampling techniques actually performed back-to-back sampling. This study concluded that the event-driven sampling methods perform better than the time-driven methods, and that the performance differences from different selection patterns were small. Today's routers incorporate sampling techniques similar to those described in [5]. Cisco's *NetFlow* monitoring system provides support for sampling 1 out of N packets in deterministic fashion [1]. Juniper's routers provide some additional flexibility. They allow administrators to apply packet filters before the sampling is done and to request that a configured run length of packets be collected with each sampling event [2]. The ability to collect a set of packets with each sample enables the evaluation of temporal correlations between transmissions. However, this ability comes at the cost of recording significantly more data.

Event-driven sampling method has great difficulty in measuring traffic characteristics such as packet interarrival time. The problem is that the sampling only gives information about the interarrival time between samples, rather than that between back-to-back packet pairs. In [6], interarrival times of the packets between two adjacent packet samples were assumed be the same, and were estimated by dividing the sampled interarrival time by the number of gaps in between (**naive averaging estimation**). The estimated distribution biases toward the overall mean of the interarrival time and does not give enough emphasis at the extreme values as we show in Figure 3.

Back-to-back sampling can provide a good estimate of interarrival times. However, it only gives us information about back-to-back packets (packets 1-step away) and does not give information about packets separated more steps away which are important if sequential packets are correlated.

Random sampling and configured run-length sampling could provide n-step histograms. However, their overhead can be high, and for random sampling, the size of the collected histograms are not predictable. In contrast, our proposed technique, *FastCARS* , is *correlation-aware*, computational efficient and predictable.

III. Proposed Method

Our main goal is to provide a sampling method that provides accurate statistical estimation, and is also simple, scalable, efficient, predictable and capable of capturing temporal correlation. To achieve these objectives, we propose a fast, correlation-aware sampling method (*FastCARS*).

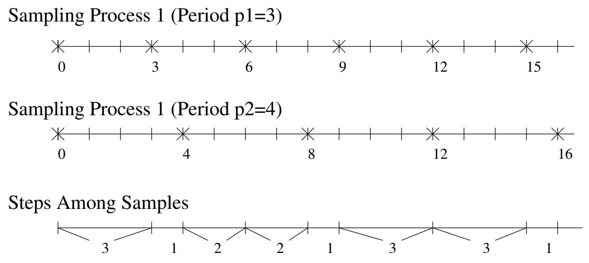
The *FastCARS* method is a combination of multiple deterministic event-driven sampling processes with sampling intervals that are *relative prime numbers*. For every sampled packet, its header information, such as time stamp on arrival, packet size, source/destination addresses, source/destination ports, and protocol, is stored for subsequent processing.

Definition 6: ***FastCARS***$(p_1, p_2, \ldots, p_n)$ sampling method consists of n sampling processes, where $p_1, \ldots, p_n$ are relatively prime numbers. The i-th process has sampling period p_i, which takes one sample every p_i events. **Events** are the objects on which the sampling method is applied. In the case of network traffic, events can be packet arrivals.

Definition 7: *FastCARS*$(p_1, p_2, \ldots, p_n)$ starts all n sampling processes at the same time. We can further generalize *FastCARS* by specifying the starting time of the n processes. We denoted the generalized *FastCARS* by *FastCARS*$(p_1, p_2, \ldots, p_n, s_1, s_2, \ldots, s_n)$, where s_i is the starting time of the i-th process. Note that the configured run-length sampling is a special case of the generalized *FastCARS*. That is, $ConfRL(p, q) =$ *FastCARS*$(p_1, p_2, \ldots, p_n, 0, 1, \ldots, q-1)$, where $p_1 = p_2 = \ldots = p_n = p$.

Figure 1 shows how *FastCARS* works. The shown *FastCARS*(3,4) method has two sampling processes taking samples at periods of 3 and 4 packet arrivals. The figure also shows that the samples collected are either 1-step, 2-step, or 3-step away from each other.

Temporal statistics such as interarrival time depend on the values of two consecutive samples. *FastCARS* collects values for such statistics into n-step histograms. In the case shown in Figure 1, for *FastCARS*(3,4), we will have 1-step, 2-step and 3-step interarrival time histograms. In general, when the sampling periods are chosen to be relative primes and let p_{min} be the minimum among the chosen sampling periods, *FastCARS* guarantees us to have samples of steps ranging from 1 to p_{min}.



Sampling Process 1 (Period p1=3)
0 3 6 9 12 15
Sampling Process 1 (Period p2=4)
0 4 8 12 16
Steps Among Samples
3 1 2 2 1 3 3 1

Fig. 1. How *FastCARS* works *FastCARS*(3,4) is shown here as an example of how *FastCARS* works. *FastCARS*(3,4) has the two sampling processes with sampling intervals 3 and 4. The top two lines indicate the packet numbers sampled by the two sampling processes. The bottom line shows the steps among samples. In this case, we can collect interarrival times of 1, 2, and 3 steps, each of them twice, in every 12 packet arrivals.

This provides the predictability on the sampling result, which random sampling can not give us. *FastCARS* is also tunable in the sense that the sampling intervals can be chosen such that samples of particular steps which are of special interests will occur more often.

FastCARS is a simple generalization of the event-driven sampling which can be efficiently implemented. Event-driven sampling of sampling interval p can be implemented using a counter to keep track of how many packets to be skipped before taking the next sample. The counter is initialized to p. Whenever a packet arrives, the counter is decreased by 1. When the counter reaches 0, the packet arrived is sampled, and the counter is reset to p. *FastCARS* could be implemented similarly with one counter per sampling process.

Figure 2 compares *FastCARS* with other sampling methods. The sampling methods being compared are **event-driven sampling**, **back-to-back sampling**, and **configured run-length sampling**.

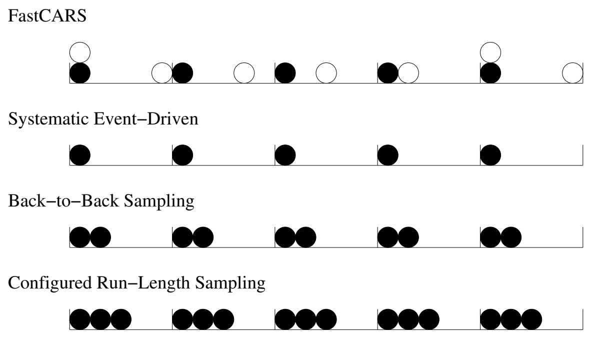


Fig. 2. Comparison of Different Sampling Methods Sampling methods compared here are *FastCARS*(p_1, p_2), event-driven, back-to-back sampling, and the configured run-length sampling, from top to bottom, respectively. Each ball shown (either filled or empty) indicates a sample taken. The *FastCARS* method shown here has two sampling processes with sampling intervals n and $(n-1)$ and their samples are shown as filled balls and empty balls, respectively. The event-driven sampling is used in CISCO routers (sampled NetFlow), and the configured run-length sampling is implemented in Juniper routers. In this figure, the configured run-length is 3.

Configured run-length sampling and random sampling [5] also give us n-step histograms. However, *FastCARS* has several advantages over these other schemes. For example, *FastCARS* is computationally simpler than random sampling. In addition, random sampling does not provide guarantees about the sampling rate for different n-step histograms. In order to collect an n-step histogram, the run-length sampling must be configured to collect n packets at a time. To collect for an arbitrary value of n, run-length sampling must collect *every* packet. In addition, since the run-length scheme collects packets in close succession, it may overload the router for an extended period of time.

IV. Experimental Results

In this section, we present experimental results showing that information collected by *FastCARS* can reveal traffic characteristics accurately. We show that the collected information could be used for typical measurement application, such as interarrival time distribution, as well as novel applications such as inspecting correlation at multiple aggregation levels.

Our experiments are done on the packet header traces obtained from the National Laboratory for Applied Network Research (NLANR) [1]. The results shown in the following sections are mainly from three traces, which we named as **AIX**, **COS** and **IND**. Table I summarizes the details of these traces. The trace collectors are located at aggregation points within HPC networks, the vBNS and Internet2 Abilene. Therefore, the network traffic considered in this paper is traffic in which many independently originated flows are multiplexed.

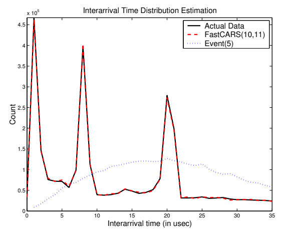
TABLE I
Summary of Traces

Trace	Location	Collected Time (GMT)	Link Speed
AIX	AIX/MAE-West Interconnection	Sunday June 10 2001 15:55:50	OC12c PoS
COS	Colorado State University	Monday August 20 2001 00:47:57	OC-3
IND	QuestPOP at IUPUI (Indianapolis)	Tuesday August 21 2001 22:47:04	OC12c ATM

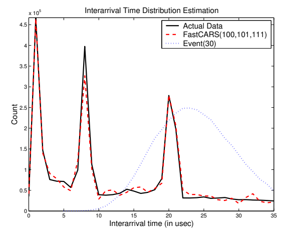
A. Accuracy of FastCARS: Interarrival Time Distribution

In this section, we show that *FastCARS* can give accurate estimation of interarrival time distribution.

Figure 3 compares our estimation with the actual interarrival time distribution collected from the full trace (AIX), and also with the results from the event-driven sampling method. Results on traces COS and IND are similar and not shown here.



(a) FastCARS(10,11) - Event(5)



(b) FastCARS(100,101,111) - Event(30)

Fig. 3. Estimating Interarrival Time Distribution Experiments are done using AIX trace. *FastCARS* gives better estimation that event-driven sampling does. (a) *FastCARS*(10,11) VS Event(5) sampling, (b)*FastCARS*(100,101,111) VS Event(30) sampling.

[1] http://moat.nlanr.net/Traces/Traces/

We used the 1-step histogram collected by *FastCARS* to estimate the interarrival time distribution. We investigate the effects of different number of processes and different sampling intervals on *FastCARS*. The choice of (10,11) and (100,101,111) as sampling intervals in our experiments is for the convenience of analysis. As mentioned in section III, relatively prime sampling intervals guarantee us the 1-step histogram. The estimation of interarrival time using samples from event-driven sampling is done by the naive averaging estimation (section II). It can be seen that the estimations from *FastCARS* samples are very closed to the actual distribution, and as expected, those from event-driven sampling bias toward the distribution mean.

We perform the comparison on the basis of equal amount of samples. Note that *FastCARS* runs more than one sampling processes and therefore takes more samples than a similar event-driven sampling does, if the same sampling interval is used. We tuned the sampling interval of the event-driving sampling method such that equal amount of samples are taken. For example, *FastCARS*(10,11) takes 713435 samples on trace AIX, and it is compared with Event(5), which takes 747407 samples.

B. Testing the Independence Hypothesis of Packet Arrivals

The hypothesis that packet arrivals are independent facilitates tasks such as traffic analysis and modelling. However, is this assumption realistic? A connection usually sends a flow of packets and the transmission times and contents of these packets are not independent. *Do these packets make the independence hypothesis false? Are there other dependences among packets?* In this section, we show how to make use of histograms gathered from *FastCARS*, to explore temporal among packets.

We investigate the independence hypothesis of packet arrivals by the convolution of histograms. We use the 1-step and 2-step histograms collected by *FastCARS* to check the independence hypothesis of packet arrivals. The idea is as follows. 1-step histogram captures the distribution of the actual (1-step) interarrival time. and 2-step histogram captures the distribution of the 2-step interarrival time. If the interarrival times are independent, then a 2-step interarrival time could be viewed as the sum of two 1-step interarrival times. As a result, we would expect that *the distribution of the 2-step interarrival time will be similar to the convolution of the 1-step interarrival time distribution, if the (1-step) packet interarrival time is independent*. In other words, the 2-step histogram should be similar to the convolution of the 1-step histogram, if the independence hypothesis of the packet arrivals holds. In the remainder of this paper, we refer to this checking of independence as **convolution test**.

Let $f_i(.)$ be the probability mass function of the i-step interarrival time distribution (time is discretized into mini-seconds). For 3 consecutive packet arrivals (x_1, x_2, x_3), let T_1 (T_2) be the random variable of the interarrival time between x_1 and x_2 (x_2 and x_3). T_1 and T_2 follows $f_1(.)$. Let D be the random variable of the 2-step interarrival time between x_1 and x_3, and D follows $f_2(.)$.

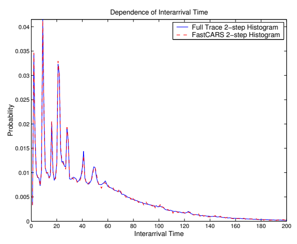
Definition 8: **Convolution Test** If the packet (1-step) interarrival time is independent, then the probability distribution of 2-step interarrival time distribution (D) is the convolution of the 1-step interarrival time distribution (T_1 or T_2). This is due to

$$Pr(D=d) = Pr(T_1+T_2=d)$$
$$= \sum_{t=0}^{d} Pr(T_1=t, T_2=d-t) = \sum_{t=0}^{d} Pr(T_1=t)Pr(T_2=d-t),$$

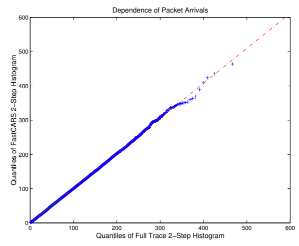
where $Pr(E)$ denotes the probability of an event E.

Figure 4 compares the 2-step histogram and the convolution of the 1-step histogram, both obtained from *FastCARS*(10,11), with the actual 2-step histogram collected from full trace AIX. Results on traces COS and IND are similar and not shown here. The histograms are normalized before doing convolution and comparison. We use the quantile-quantile plot (QQ-plot) of the two histograms as a visualization of the similarity between two histograms.The fit of the QQ-plot to the 45-degree reference lines demonstrates the goodness of fit between the two histograms.

It can be seen that *FastCARS* gives accurate 2-step interarrival time distributions (QQ-plot in Figure 4(a.2) goes along the 45^o line). The big discrepancy shown in Figure 4(b.2) indicates that the actual 2-step histogram is different from the convolution of the 1-step histogram. Therefore, by the convolution test, the interarrival time is not independent.

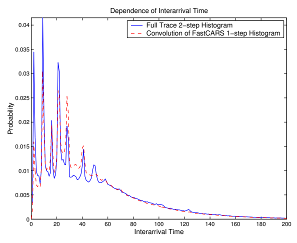


(a.1) Count vs Interarrival time

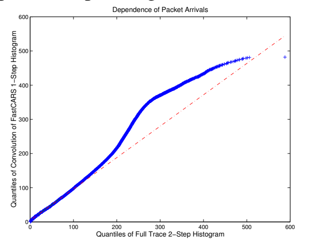


(a.2) Actual vs Sampled Histograms

(a) Actual 2-Step & Sampled 2-Step Histograms



(b.1) Count vs Interarrival time



(b.2) Actual vs Convoluted Histograms

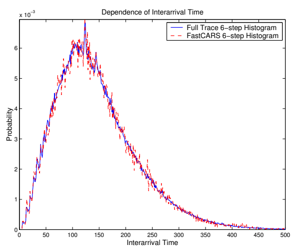
(b) Actual 2-Step & Convoluted Sampled 1-Step Histograms

Fig. 4. Dependence of Interarrival Time Experiments are done on traces AIX. Histograms are collected by *FastCARS*(10,11). Here we compares the actual 2-step interarrival time histogram to (a) the sampled interarrival time histogram, and (b) the convoluted 1-step interarrival time histogram. The discrepancy of the QQ-plot in (b.2), along with the convolution test (definition 8), shows that the (1-step) interarrival time is not independent.

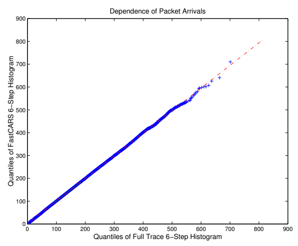
It is expected that as the number of steps increase, the packet arrivals should be less dependent on each other. For example, packets 3 steps away are expected to be more independent of one another, and as a result, the sum of two 3-step interarrival times should be a good estimation of a 6-step interarrival time. This suggests that the convolution of 3-step histogram should be similar to the 6-step histogram.

Figure 5 shows the results on comparing the actual 6-step histogram from full trace (AIX) to (a) sampled 6-step histogram and (b) the convolution of the sampled 3-step histogram. The

6-step interarrival time histogram from *FastCARS* is still a good estimation for 6-step interarrival time distribution (Figure 5(a)). In Figure 5(b.2), the actual 6-step histogram fits well with the convolution of the 3-step histogram, and is clearly better than the fit of Figure 4(b.2). This shows that, as expected, the dependence of packet arrival time diminishes as the separation between packets increases.

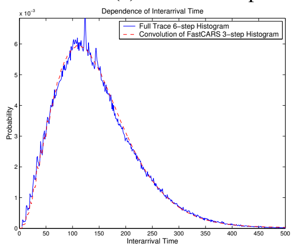


(a.1) Count vs Interarrival time

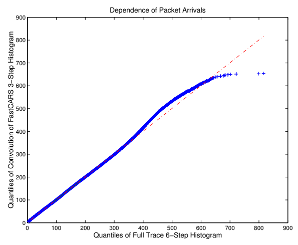


(a.2) Actual vs Sampled Histograms

(a) Actual 6-Step & Sampled 6-Step Histograms



(b.1) Count vs Interarrival time



(b.2) Actual vs Sampled Histograms

(b) Actual 6-Step & Convoluted Sampled 3-Step Histograms

Fig. 5. Dependence of Interarrival Time The trace used in this example is AIX. Here we compares the actual 6-step interarrival time histogram to (a) the sampled the 6-step interarrival time histogram, and (b) the convoluted 3-step interarrival time histogram. The smaller discrepancy of the QQ-plot in (b.2), comparing to (Figure 4(b.2)) shows that the 3-step interarrival time less dependent than 1-step interarrival time.

C. Dependence Assumption and "Packet Train" Phenomenon

A sequence of packets with same source, destination IP addresses and port numbers forms a "packet train". It is known that the packet train phenomenon exists in network traffic [8]. Packets within a packet train are not expected to act independently, and may affects traffic characteristics. For example, it may cause the independence hypothesis of packet arrivals to be incorrect.

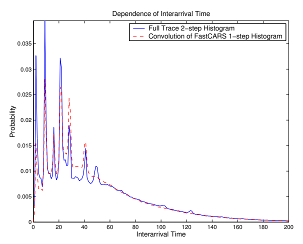
Is it true that packet trains are the main reason that the independence assumption of packet arrivals is false? We test this hypothesis by removing consecutive packets that are in the same flow (packet trains) from the full trace, and then check whether (1-step) interarrival time histogram of the resulting trace set has the independence property. The independence check is done using our *convolution test*.

Figure 6 shows the results of the convolution test on AIX trace, after packet trains are removed. The 1-step and 2-step histograms are collected by *FastCARS*(10,11). Since the discrepancy in the quantile-quantile plot in Figure 6 remains significant, this suggests that packet trains might not be the sole cause of the failure of the independence hypothesis of packet arrival.

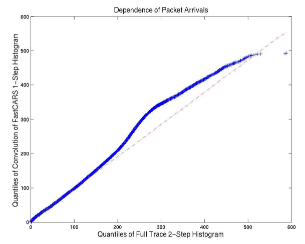
To summarize, we shown how we could use the information provided by *FastCARS* on several data mining applications. In particular, we explored the independence hypothesis of packet arrivals and the packet train phenomenon. We proposed the *convolution test* for testing dependence of packet arrivals. From our experiments, we made several observations:

Observation 1: The assumption of independent arrivals is not correct.

Observation 2: Packet trains is not the sole cause of the failure of the independence hypothesis of packet arrival.



(a) Frequency vs Interarrival time



(b) Actual vs Convoluted Histograms

Actual 2-Step & Convoluted Sampled 1-Step Histograms

Fig. 6. Packet Train and Dependence of Interarrival Time Experimental results on trace AIX (after removing the packet trains within) are shown. The number of packets removed is 155863, out of the total 3737038 packets. Histograms are collected using *FastCARS*(10,11). The discrepancy in (b) remains significant (compared to Figure 4(b.2)). This suggests that packet train may not be the sole cause of the failure of the independence hypothesis of packet arrival.

V. Conclusion

In this paper, we present *FastCARS* , a fast, correlation-aware sampling method for network data mining, which is (1) **accurate** on providing traffic statistics, (2) **simple and scalable** on implementation, and (3) **correlation-aware** in the sense that it easily captures information about n-step histograms and therefore reveals short and long term correlation among packet arrivals.

Using the information obtained from *FastCARS*, we also provide several new tools for network data mining, namely, the **n-step histograms** (section II, definition 1), and the **convolution test** (section IV-B, definition 8).

In addition, *FastCARS* and our tools enable the following observations on real world traffic traces:

1) *FastCARS* preserves traffic characteristics and accurately estimates the interarrival time distribution.
2) The assumption of independent arrivals is not correct.
3) Packet trains are not the sole cause of the failure of the independence hypothesis of packet arrival.

FastCARS can also be used in other areas that demand accurate, efficient, and correlation-aware sampling techniques. For example, *FastCARS* could be used to compare synthetic traces from traffic generators to real-world data. This would insure that the traffic generators created traces that had the appropriate temporal correlations as well as the normally tested long-term aggregate distributions.

References

[1] Cisco Systems Inc., "Netflow services solutions guide," *http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/netflsol/nfwhite.htm*, August 2001.

[2] Juniper Networks Inc., "Junos 5.0 internet software configuration guide: Interfaces, class of service, and firewalls: Configure traffic sampling and forwarding," *http://www.juniper.net/techpubs/software/junos50/swconfig50-interfaces/html/sampling-config.html*, August 2001.

[3] R. Jain and S. Routhier, "Optimizing bulk data transfer performance: A packet train approach," *IEEE Journal on Selected Areas in Communications*, vol. 4, no. 6, pp. 986–995, September 1986.

[4] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, February 1995.

[5] Kimberly C. Claffy, George C. Polyzos, and Hans-Werner Braun, "Application of sampling methodologies to network traffic characterization," *Proceedings of SIGCOMM 93*, 1993.

[6] Kedar Dhandhere, Hyang-Ah Kim, and Jia-Yu Pan, "The application and effect of sampling methods on collecting network traffic statistics," http://www.cs.cmu.edu/∼jypan/writing/network_sampling.ps.gz, May 2001.

[7] John M. Chambers, W. S. Cleveland, B. Kleiner, and P. Tukey, *Graphical Methods for Data Analysis*, Pacific Grove: Wadsworth and Brooks/Cole, 1983.

[8] Cheng Song and Lawrence H. Landweber, "Optimizing bulk data transfer performance: A packet train approach," *Proceedings of SIGCOMM 88*, September 1988.