LA-UR- 04 -3552

Approved for public release;
distribution is unlimited.

Title: AN INVESTIGATION OF PACKET REORDERING IN TCP TRACES (EXTENDED ABSTRACT)

Author(s): Gabriel "NMI" Istrate, CCS-5
Anders A. Hansson, CCS-5
Matthew S. Nassr, CCS-5
Christopher L. Barrett, CCS-5
Madhav V. Marathe, CCS-5

Submitted to: ACM Internet Measurement Conference

An Investigation of Packet Reordering in TCP Traces (Extended Abstract)

Gabriel Istrate, Anders Hansson, Matthew Nassr, Christopher Barrett, and Madhav Marathe

Abstract— **Recent research has highlighted the impact of packet reordering on network dynamics. Still, while much work has investigated the statistical properties of inter-packet arrival times of TCP traces, little effort has been devoted to obtaining a model of network traffic that incorporates sequence ID numbers as well.**

With the ultimate goal to develop such a joint model, we present results on the dynamics of packet reordering in a set of publicly available TCP traces recorded at the Network Research Lab at UCLA. We investigate the scaling properties of the number of packet inversions.

We propose a two-state model for the dynamics of sequence IDs based on pivots (defined as packets for which the received packet sequence has no gaps). This concept allows us to partition the trace into time epochs based on the presence or absence of reordering. Thus, we are able to identify and store patterns of reordering in the packet streams. Statistical tests provide a first-order validation of our model.

Finally, we investigate the reordering patterns identified by our model from the standpoint of standard measures of presortedness of integer sequences.

The methodology outlined in this paper enables regeneration of synthetic traces with inversion characteristics that are statistically similar to those of the original data. It is part of RESTORED, a network inference and analysis tool under development at Los Alamos National Laboratory.

Index Terms—**Packet reordering, pivots, traffic modelling.**

I. Introduction

THE traffic flow dynamics in packet-switched communication networks is still subject of intense research. In particular, relatively little work has dealt with *packet reordering*, although this phenomenon has severe effects on many protocols, such as the Transmission Control Protocol (TCP). Instead, the focus has been on characterizing packet arrival times, and in particular, it has been demonstrated that the complexity and richness of the arrival times is well matched by the multiscale analysis and modelling frameworks of self-similarity, long-range dependence, fractals, multifractals, and infinitely divisible cascades (see e.g. [1] for a survey).

Bennett, Partridge and Shectman [2] pointed out the importance of packet reordering to network dynamics. Reordering has quantifiable effects [3] on several metrics for quality of service, e.g. throughput. Our ultimate goal is to develop a model of network traffic that is compatible with the previous results on the dynamics of inter-packet arrival times, but also takes reordering into account. That is, our model would allow to characterize (and regenerate) network traces with respect to both arrival times and sequence IDs. With this goal in mind, in this extended abstract we present some preliminary results on the scaling and modelling of packet reorder in Internet data.

II. Experimental Setup

The packet traces we use in this study were collected during August 2001 at the border router of the Computer Science Department, University of California, Los Angeles (UCLA). The set was obtained by the UCLA Network Research Lab and modified for public use by the UCLA Laboratory for Advanced Systems Research. In particular, we have used a trace of size 2.15 Gb available at http://lever.cs.ucla.edu/ddos/traces/public/trace7/tcp/. Similar traces are available on that website, and we are currently investigating the robustness of our results with respect to all available data, as well as synthetic traces, such as those generated by the network simulators *ns* and *QUALNET*.

When we parsed this trace into different connections we found that most of them are very short. In fact, out of a total of 245,718 connections, 60% contain only one or two packets, 88% contain 10 packets or less, and 98% contain 100 packets or less. However, since we are interested in highlighting nontrivial network dynamics, we have chosen to study those connections with at least 1000 packets; there are 1839 of them. The longest connection contains as many as 277,895 packets. That is, we observe the by-now well-known phenomenon of *elephants and mice* [4], and attempt to investigate and model significant characteristics of elephants.

III. Experimental Requirements for an Analytical Model of Reordering in Large Traces

Understanding the impact of reordering in network traffic is a difficult issue: Bennett *et al.* [2] have argued that reordering is a pervasive phenomenon. On the other hand Jaiswal *et al.* [5] reported measurements on TCP connections within the Sprint backbone that show that a rather small percentage of connections experience inversions, and the amount of reordering was substantially smaller than those reported in [2] and [6].

A simple argument, based on the fact that there exists a constant upper bound on the size of the congestion window in TCP implementations shows that the number of inversions should be at most linear in the stream length. However, studies such as [2] and [5] have not investigated the dependence of the number of inversions on the connection length.

The issue is further complicated by the fact that (see Fig. 10) the number of inversions in a packet stream does *not* scale with the stream length: in our data we have found very large streams (the largest having close to 50,000 packets) containing *no inversions*. On the other hand the *largest number of inversions* of a stream of length at most T could still depend on T. To test this, we first ordered streams by length, and investigated how the "largest" number of inversions of the first n streams in this

This work was partially supported by the LDRD-DR project *Scalable Reconfigurable Computing* and the LDRD-ER project *Advanced Techniques in Discrete Simulation* at Los Alamos National Laboratory.

The authors are with the Basic & Applied Simulation Science Group (CCS-5), Los Alamos National Laboratory, P.O. Box 1663, MS M997, Los Alamos, NM 87545 (e-mail: istrate@lanl.gov; hansson@lanl.gov; mnassr@lanl.gov; barrett@lanl.gov; marathe@lanl.gov).

ordering scales against the length of the longest of them. Plotting the 95'th percentile of the number of inversions against the *logarithm* of the stream length (Fig. 2) makes apparent that for most streams the number of inversions scales at most polylogarithmically with the stream length. This is further explored in Fig. 3, where the ratio between the logarithm of the number of inversions and the double logarithm of the sequence length is plotted (again, for the 95'th percentile). The graph is shown up to stream length 27,000 to showcase the fact that the sequence with the largest number of inversions has length slightly less than this value.

Similar plots (albeit with different constants) can be drawn for the average, median and the 99'th percentile, highlighting the fact that the polylogarithmical scaling is robust for the data we consider. It would also hold, of course, if instead of all traces of "significant" length we would have considered all traces; we believe, however, that our result is slightly more interesting for elephants.

IV. A Coarsened Model of Network Traffic: Pivot Packets

A simplifying assumption we make in this paper is that packets have identical payload (that is, our aim is to model reordering, rather than packet fragmentation). This allows us to bijectively map TCP sequence numbers to a sequence of packet IDs, the smallest being 1. Consider the following example of an initial ID segment of a connection

$$1 \quad 2 \quad 3 \quad 5 \quad 6 \quad 7 \quad 4 \quad 8 \quad 9 \quad 11 \quad 12 \quad 13 \quad 10 \quad 14 \quad \cdots \tag{1}$$

The sequence reflects the packet arrival order, e.g., packet 4 is delayed. Recall that received packets may be buffered before delivery to guarantee that an ordered packet stream is always uploaded to the application layer. Thus, there are two types of packets:[1]

- Packets that can be immediately passed to the application layer,
- Packets that have to be buffered.

Since reordering is directly associated with buffering, the streams can be partitioned into segments of consecutive packets that belong to either one of two phases:

- The *ordered* phase (in which packets arrive in order),
- The *unordered* phase (in which packets arrive out of order).

For example, packets 5, 6, 7 are temporarily buffered, and the buffer is not flushed until packet 4 has been received. Formally, a received packet for which the buffer could be flushed is called a *pivot packet*. In the example, packets 1, 2, 3, 4, 8, 9, 10, and 14 are thus pivots. Note that this definition fits well with a first-order approximation of TCP, in which there are two regimes:

- State 1, the ordered state corresponding to the absence of buffering,
- State 2, the unordered state corresponding to a non-empty buffer.

This two-state description is naturally related to the dynamics of TCP in that the transmission rate is increased in the absence of reordering and decreased when reordering occurs. Furthermore, it provides an operational motivation to the assumption that the time series derived from observations of network traffic (e.g. the inter-packet arrival time) are stationary: after receiving a pivot packet the TCP protocol is "in the same state" (if we ignore differences in quantities such as size of the congestion window, the number of packets in transit, etc.). This shows that, at least in a first-order approximation, we can assume that the characteristics of network traffic, *when coarsened at the level of pivot packets*, are stationary. In contrast, models of network traffic often assume (see e.g. [7]) *second-order stationarity* of these timeseries without any plausible motivation. Moreover, for large enough timescales the stationarity assumption need not hold [8], and it is not entirely clear at what time scales this assumption is warranted.

Another advantage of the decomposition in states delimited by pivot packets is that it localizes the reordering process: packets that create an inversion are located in a segment corresponding to the same state. This allows us to attempt to relate (as we do in a subsequent section) the number of inversions in a given sequence to the number of state changes. Furthermore, if we make the assumption that the time spent in a given state (and the number of packets generated in it) are timeseries displaying no long-range dependence, we can then arrive at a very simple model of packet ID dynamics (reminiscent of the On-Off source model [7]). The model is depicted in Fig. 1.

To generate a packet stream we first choose the initial state of the network. For each of the two states there is a distribution D_i, $i \in \{1, 2\}$, over $\mathbf{N} \times \mathbf{R}^+$ such that, when in state i the generator creates p_i packets, spread over a time interval t_i (where (p_i, t_i) is a sample from the distribution D_i). D_1 contains the pair $(0, 0)$ to reflect that one can have consecutive runs of the unordered state.

In the ordered state packet IDs are assigned consecutive values, whereas in the unordered state packet IDs are obtained by first generating an *inversion pattern* (see Section VI), and then reconstructing the IDs based on this pattern. Note that we do *not* yet present a full-fledged generator of network traffic, since we do not specify a way to reconstruct inter-packet arrival times.

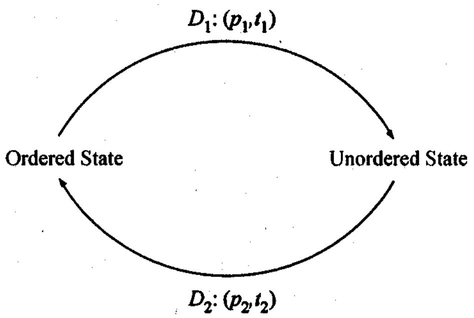

Fig. 1. Approximate model of the ID dynamics.

V. Statistical Validation of the Two-State Approximation

The model presented in the previous section can be statistically validated: Fig. 4 presents a "typical" realization of the

[1] Duplicates of those packets that have already been uploaded to the application layer are discarded, which is consistent with the operation of TCP.

autocorrelation function for the timeseries consisting of the number of packets in each consecutive segment of the ordered phase. Similar plots hold for the duration of the segments of that phase. To obtain a global view of the autocorrelation function for all traces we employ the following measurement: let $T = (t_i)$ be a time series, and let $e_T = (e_{1,T}, \dots e_{k_T,T})$ be the vector consisting of the values of the autocorrelation function of T with lags between 1 and the maximum lag for the timeseries T, say k_T. We define p_T, the *average power of the autocorrelation function of* T by

$$p_T = \frac{1}{k_T} \sum_{i=1}^{k_T} e_i^2. \qquad (2)$$

Fig. 5 and Fig. 6 present the kernel smoothed distribution functions for the average power of the autocorrelation function for the time spent and the number of packets generated in the ordered state. Similar plots are presented in Fig. 7 and Fig. 8 for the unordered state. All these distributions have their mode around value 0.1, consistent with a "typical" value of the autocorrelation exponents e_k of 0.1. One apparent problem with these figures is the existence of long tails of these distribution, which suggest that there exists significant correlation in some of these timeseries. This is, however, the consequence of small size of these: if we plot the value of the power coefficient against the length of the timeseries (as it is done, for the number of packets generated in the ordered state, in Fig. 9) we can easily see the tendency of the power to decrease for timeseries of higher length.

We have separately computed the autocorrelation function for the two components of the distributions associated with both the ordered and the unordered phase. In fact, we would expect significant correlation between the number of packets in that phase and the duration of the phase. Fig. 13 presents an example of linear regression between the number of packets generated in the ordered state and the duration of that state. The situation is, however, more complicated. In particular, the residuals do not always seem to appear from a stationary distribution. We believe that this is due to *clustering* in the packet sequence (that is, packets that arrive almost at the same time), reflecting the congestion window behavior of TCP.

Finally, we have presented results showing independence for the ordered and unordered phase. At the moment of writing the abstract the data is not conclusive enough to predict the same result for the *aggregated* timeseries (that is the number of packets generated in the unordered state is probably correlated with the number of packets generated in the preceeding ordered state, since the two segments are "coupled" by the dynamics of the congestion window). Whether this is true or not is subject to further inquiry.

VI. A Semantic Description of Inversion Patterns

Let us now discuss some of the structural properties of the inversion patterns. Our ultimate goal would be to build a grammar (or a code book) of frequent, well-structured inversion patterns. Consider the example sequence in (1) and its two unordered phases,

$$5 \quad 6 \quad 7 \quad 4 \quad \text{and} \quad 11 \quad 12 \quad 13 \quad 10. \qquad (3)$$

On the assumption that these events have similar inter-packet arrival times they are semantically identical: three ordered packets precede a fourth packet with lower ID. Therefore, we would like to use an identical representation of the two inversion patterns, and this can be readily achieved by computing the difference between the actual ID and the one we would expect to receive (had all the packets arrived in order). In our example, the expected sequences are

$$4 \quad 5 \quad 6 \quad 7 \quad \text{and} \quad 10 \quad 11 \quad 12 \quad 13 \qquad (4)$$

and we arrive at the same *difference pattern*,

$$1 \quad 1 \quad 1 \quad -3. \qquad (5)$$

Using the newly introduced difference patterns, we scanned the trace and constructed a hash table of patterns and their relative frequencies. The ten most frequent patterns are listed in Table I, in which we have also exemplified each difference pattern with a corresponding compatible inversion pattern.

TABLE I
Most Frequent Inversion Patterns

Prob. (%)	Difference Pattern	Example Pattern
58.24	1 -1	2 1
6.70	1 1 -2	2 3 1
4.82	1 1 1 -3	2 3 4 1
3.89	1 1 1 1 -4	2 3 4 5 1
2.97	1 2 3 -3 -2 -1	2 4 6 1 3 5
2.26	1 1 1 1 1 -5	2 3 4 5 6 1
2.02	1 2 -2 -1	2 4 1 3
1.54	2 -1 -1	3 1 2
1.19	1 1 1 1 1 1 -6	2 3 4 5 6 7 1
0.96	1 1 1 1 1 1 1 -7	2 3 4 5 6 7 8 1

Table I makes it apparent that inversion patterns possess a lot of structure (loosely speaking, completely random sequences have much higher complexity/entropy).

In order to quantify this we chose to compute a standard measure of disorder [9]. This measure is denoted by SUS (standing for Shuffled Up-Sequences) and is defined as the minimum number of ascending subsequences into which we can partition each listed inversion pattern.

As an example, a sequence $A = \langle 6, 5, 8, 7, 10, 9, 12, 11, 4, 3, 2 \rangle$ has $\mathrm{SUS}(A) = \|\{\langle 6, 8, 10, 12 \rangle, \langle 5, 7, 9, 11 \rangle, \langle 4 \rangle, \langle 3 \rangle, \langle 2 \rangle\}\| = 5$ (where $\|S\|$ denotes the cardinality of a set S).

All patterns in Table I have SUS=2. In fact, if we compute the SUS metric for *all* obseved inversion patterns, we find that an overwhelming majority (or 97%) of them are of type SUS=2, which can be seen in Table II.

Note that even patterns with a considerable number of inversions are relatively ordered with respect to the SUS measure. A possible explanantion of this phenomenon could be that link striping is not prevalent in the analyzed data. Therefore packets in the same connection will be sent on a limited number of paths, partitioning the stream in a small number of relatively ordered sequences. It would be really interesting to measure this presortedness metric in networks with rapidly changing paths, such as in ad-hoc networks.

TABLE II
DISORDER OF INVERSION PATTERNS

SUS	Prob. (%)
2	96.970
3	2.830
4	0.154
5	0.020
6	0.005
7	0.003
8	0.004

VII. FURTHER MODELLING THE SCALING OF THE NUMBER OF INVERSIONS

We have seen in Section III that the largest number of inversions depends polylogarithmically on the stream length. This can be partly explained by the state decomposition we outlined in the previous sections: inversions are only possible in the unordered states. However, as we show in Fig. 11 the *total* number of states the connection goes through (presented for the 95'th percentile) depends polylogarithmically on the sequence length.

We would have expected the number of inversions in a sequence to depend linearly on the number of states in the given connection. Fig. 12 only shows, however, a much weaker correlation of the two: in essence the number of inversions is upper bounded by a power of the number of states in the trace.

Together these two results explain the polylogarithmic scaling of the number of inversions with sequence length, but leaves open the question why does the typical number of *states* scale logarithmically with sequence length.

VIII. CONCLUSIONS

In this paper we undertook an empirical approach to the problem of modelling packet reorder in real network traces. We first obtained an estimate on the scaling properties of the number of inversions with respect to the trace length. We introduced the concept of *pivot packets*. This allowed us to partition a given connection based on the presence or absence of packet reordering. It also led us to propose a simple two-state model of ID dynamics that was statistically validated at least to a first-order approximation. We measured the presortedness of the resulting inversion patterns, and identified a suitable measure of disorder that showed that the patterns posses a high degree of regularity. Finally, we identified the characteristics of our state-model that can be used to approach the problem of modelling the scaling of the number of inversions observed in real data.

The results and insights presented in this paper have heen incorporated into RESTORED, a network inference and analysis tool currently under development at Los Alamos National Laboratory.

REFERENCES

[1] P. Abry, R. Baraniuk, P. Flandrin, R. Riedi, and D. Veitch, "Multiscale nature of network traffic," *IEEE Signal Processing Magazine*, pp. 28–46, May 2002.

[2] J. Bennett, C. Partridge, and N. Shectman, "Packet reordering is not pathological network behavior," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, December 1999.

[3] M. Laor and L. Gendel, "The effect of packet reordering in a backbone link on application throughput," *IEEE Network*, pp. 28–36, 2002.

[4] *New Directions in Traffic Measurement and Accounting: Focusing on the Elephants, ignoring the Mice*, 2001.

[5] S. Jaiswal, G. Iannacone, C. Diot, J. Kurose, and D. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 backbone," in *Proceedings of INFOCOM*, 2003.

[6] V. Paxson, "End-to-end internet packet dynamics," in *Proc. ACM SIGCOMM '97*, September 1997, pp. 139–152.

[7] K. Park and W. Willinger, *Self-Similar Network Traffic and Performance Evaluation*, Wiley, 2000.

[8] Jin Cao, William S. Cleveland, Dong Lin, and Don X. Sun, "On the nonstationarity of internet traffic," in *Proc. ACM SIGMETRICS*, 2001, pp. 102–112.

[9] V. Estivill-Castro and D. Wood, "A survey of adaptive sorting algorithms," *ACM Computing Surveys*, vol. 24, no. 4, pp. 441–476, 1992.

APPENDIX: FIGURES

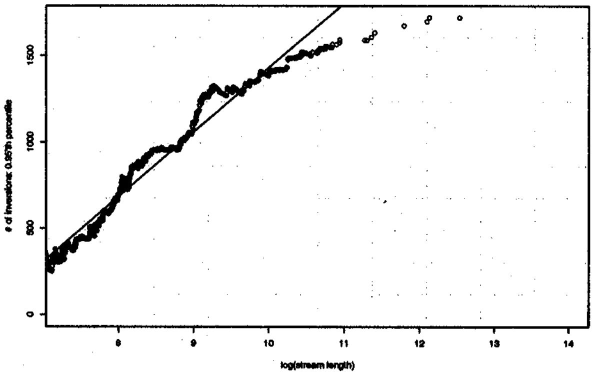

Fig. 2. Scaling of the number of inversions in large files: The 95'th percentile of the number of inversions versus the logarithm of the stream length.

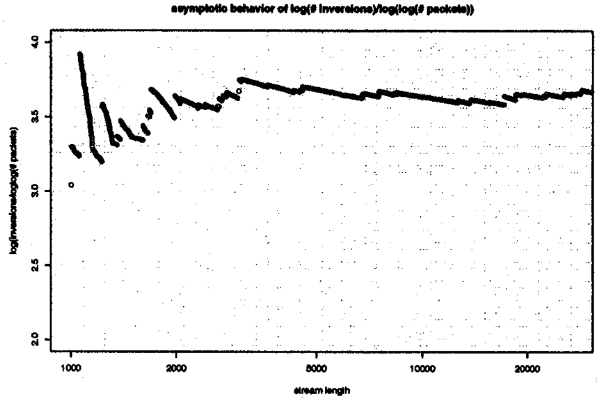

Fig. 3. Evidence for polylogarithmic scaling of the number of inversions in large files.

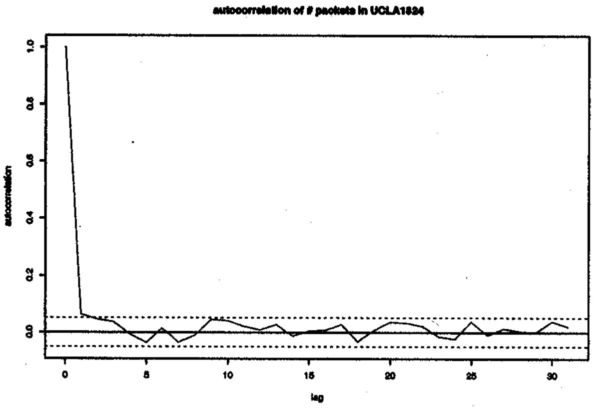

Fig. 4. Sample autocorrelation of the number of packets in State 1.

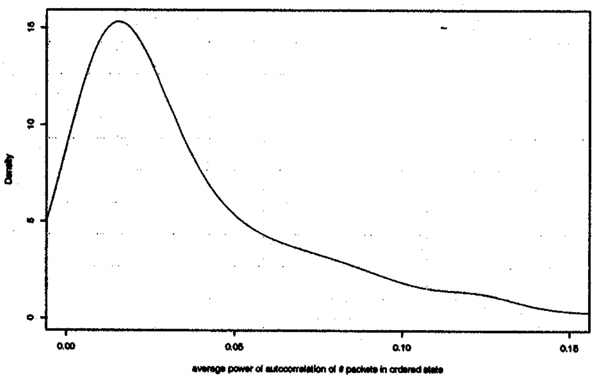

Fig. 5. Distribution: power of autocorrelation of the number of packets generated in the ordered state.

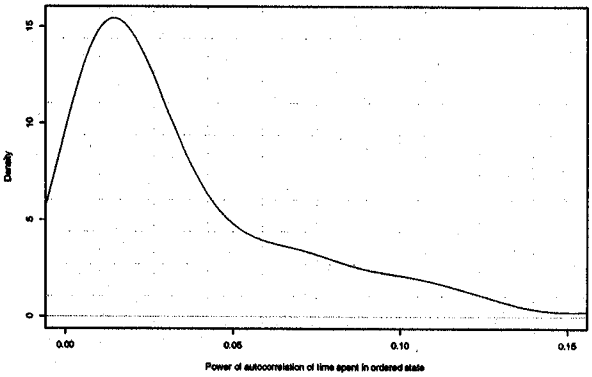

Fig. 6. Distribution: power of autocorrelation of the time spent in the ordered state.

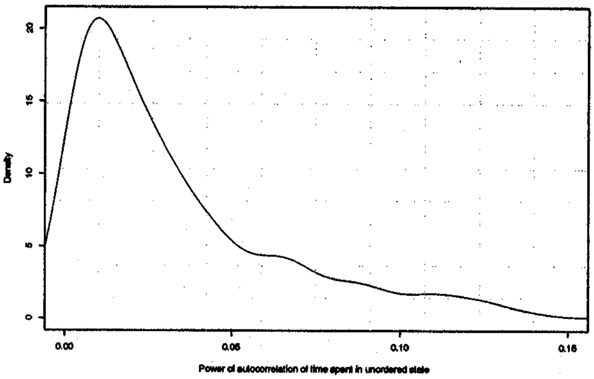

Fig. 7. Distribution: power of autocorrelation of time spent in the unordered state.

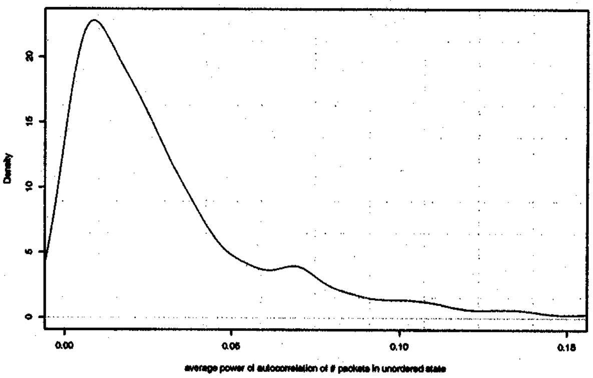

Fig. 8. Distribution: autocorrelation of the number of packets generated in unordered state.

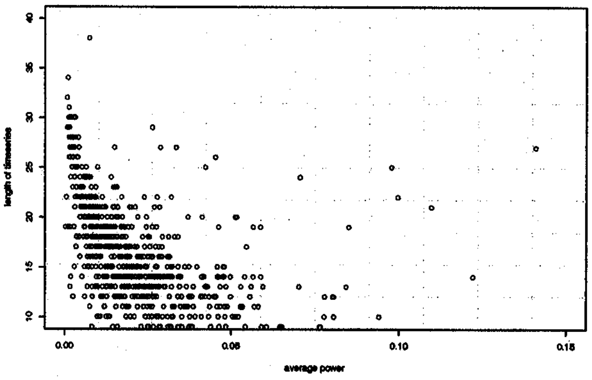

Fig. 9. Dependence of the power of the autocorrelation on the length of the timeseries.

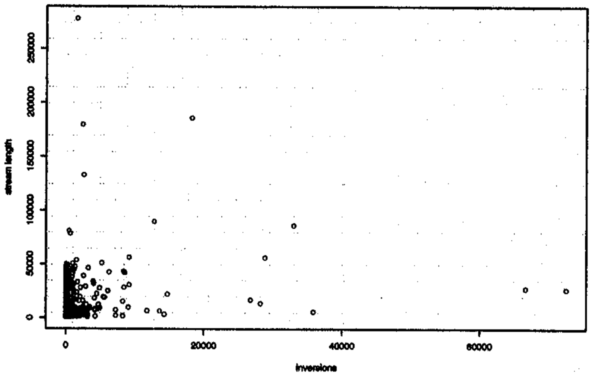

Fig. 10. Dependence of the number of inversions on the stream length

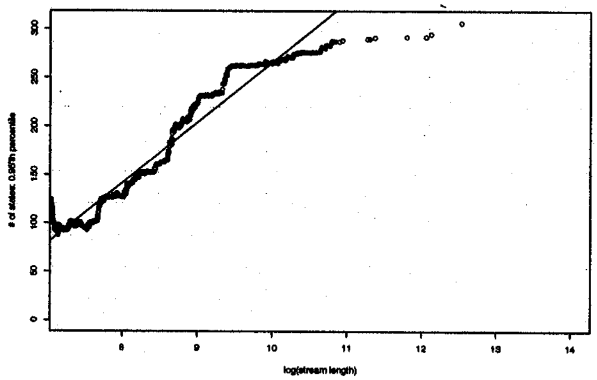

Fig. 11. Dependence of the number of states on the stream length.

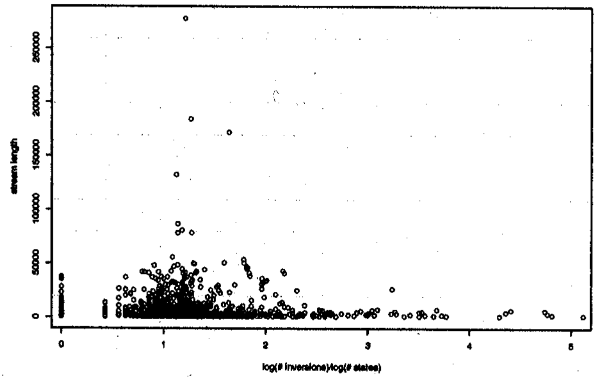

Fig. 12. Dependence of the number of inversions on the number of states.

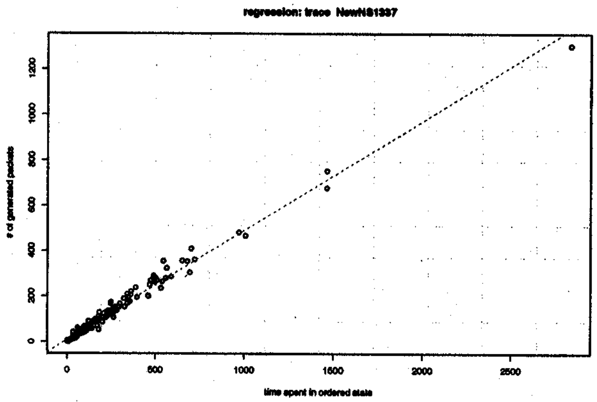

Fig. 13. Sample regression: time spent vs. the number of packets generated in the ordered state.