# CMPT815 Final Examination

Wenguang Wang

April 19, 2000

1.(a) [20 marks] *Develop a generator for exponentially-distributed random variates based on the pseudo-random number generator Random given in Figure 26.2 of Jain's book, and the inverse transformation method discussed in class. Write a program to generate 10,000 exponentially-distributed random variates with a mean of 1.0 using each of 20 randomly generated seed values taken from a separate $U(0,m)$ generator. Compute the mean of the exponentially-distributed random variates generated for each seed, the variance, and the coefficient of variation. How much variation do you observe in these values with different seeds? Using the method described in Section 13.2 of Jain's book, compute the 95% and 99% confidence intervals for the computed mean as an estimate of the true mean. How sensitive is your pseudo-random number generator for exponentially-distributed random variates to choice of seed?*

In this question, a pseudo-random number generator Random given in Figure 26.2 of Jain's book [Jai91] is implemented. A separate $U(0,m)$ generator is developed to generate the 20 seeds required. This generator is defined in Equation 1. Note that the prime number 1009 is selected so as to make the distribution of seeds more evenly. Moreover, it has been verified (not shown in this report) that the 20 random number streams generated by the 20 seeds taken from this generator do not overlap with each other.

$$x_{n+1} = x_n \times 7^5 \mod 2147483637 \mod \mathbf{1009} \tag{1}$$

Table 1 lists the mean, the variance, and the coefficient of variation (i.e. C.O.V.) of the exponentially-distributed random variates generated by the 20 randomly selected seeds, respectively. The 95% and the 99% confidence intervals for these computed means as an estimate of the true mean are presented as well. The $100(1-\alpha)\%$ confidence interval is computed by Equation 2, which is described in Section 13.2 of Jain's book [Jai91]:

$$(\bar{x} - z_{1-\frac{\alpha}{2}} s/\sqrt{n}, \bar{x} + z_{1-\frac{\alpha}{2}} s/\sqrt{n}) \tag{2}$$

where $\bar{x}$ is the sample mean, $s$ is the sample standard deviation, and $n$ is the sample size (which is $10,000$ in this question). For the 95% and the 99% confidence intervals, $z_{1-\frac{\alpha}{2}}$ is 1.960 and 2.576, respectively.

It can be easily seen from Table 1 that both the mean, the standard deviation, and the C.O.V of each of these 20 streams are very close to their true values (the mean, the standard deviation, and the C.O.V should all be 1.0 for exponentially-distributed random variates). Therefore, it can be said that these 20 streams do follow the exponential distribution.

Table 2 shows the mean, the variance, and the C.O.V. for the values of the mean, the variance, the C.O.V., and the 95% and the 99% confidence intervals of the 20 streams generated. It can be seen from the table that both the variance and the C.O.V. of these values are quite small (at a magnitude of $10^{-2}$). This indicates that these values (i.e. the mean, the variance, the C.O.V., and the 95% and the 99% confidence intervals), respectively, are quite close to each other across the 20 streams. Therefore, the exponentially-distributed random variates are insensitive to the choice of seed.

(b) [10 marks] *choose the best of your 20 streams from part (a). Plot the upper and lower values of the 95% and 99% confidence intervals for the mean as a function of the population size, with points for every 500 values generated (i.e. 500 values, 1000 values, 1500 values, 2000 values, etc.). Do the confidence intervals widen or shrink as the population increases? Explain why.*

The first problem is how to choose the best one from the 20 streams. Several metrics could be selected to measure the goodness of a sample stream, which are listed as follows:

| No. | Mean | Variance | C.O.V. | 95% conf. interval | 99% conf. interval |
|---|---|---|---|---|---|
| 1 | 1.005600 | 0.998390 | 0.993628 | (0.986018, 1.025190) | (0.979863, 1.031340) |
| 2 | 0.984671 | 0.978271 | 1.004470 | (0.965285, 1.004060) | (0.959193, 1.010150) |
| 3 | 1.002760 | 1.035600 | 1.014840 | (0.982817, 1.022710) | (0.976548, 1.028980) |
| 4 | 0.998682 | 0.986950 | 0.994765 | (0.979210, 1.018150) | (0.973091, 1.024270) |
| 5 | 0.997386 | 1.007080 | 1.006160 | (0.977717, 1.017060) | (0.971535, 1.023240) |
| 6 | 0.994653 | 0.976411 | 0.993447 | (0.975286, 1.014020) | (0.969199, 1.020110) |
| 7 | 1.002270 | 0.985633 | 0.990544 | (0.982809, 1.021730) | (0.976693, 1.027840) |
| 8 | 0.994499 | 0.994208 | 1.002610 | (0.974956, 1.014040) | (0.968814, 1.020180) |
| 9 | 1.022180 | 1.050240 | 1.002570 | (1.002100, 1.042270) | (0.995785, 1.048580) |
| 10 | 0.992541 | 0.994035 | 1.004510 | (0.972999, 1.012080) | (0.966858, 1.018220) |
| 11 | 1.006140 | 1.068480 | 1.027370 | (0.985876, 1.026400) | (0.979509, 1.032760) |
| 12 | 1.001020 | 1.025710 | 1.011740 | (0.981170, 1.020870) | (0.974932, 1.027110) |
| 13 | 0.992598 | 0.998085 | 1.006490 | (0.973017, 1.012180) | (0.966863, 1.018330) |
| 14 | 1.007560 | 1.019670 | 1.002220 | (0.987763, 1.027350) | (0.981543, 1.033570) |
| 15 | 1.007880 | 0.970554 | 0.977466 | (0.988570, 1.027190) | (0.982501, 1.033260) |
| 16 | 1.000040 | 1.003610 | 1.001770 | (0.980401, 1.019670) | (0.974230, 1.025840) |
| 17 | 1.014860 | 1.065010 | 1.016880 | (0.994636, 1.035090) | (0.988279, 1.041450) |
| 18 | 1.017160 | 1.065940 | 1.015030 | (0.996919, 1.037390) | (0.990559, 1.043750) |
| 19 | 1.003800 | 1.022170 | 1.007200 | (0.983981, 1.023610) | (0.977753, 1.029840) |
| 20 | 1.008840 | 1.021640 | 1.001910 | (0.989026, 1.028650) | (0.982800, 1.034870) |

Table 1: The Statistics of the 20 streams

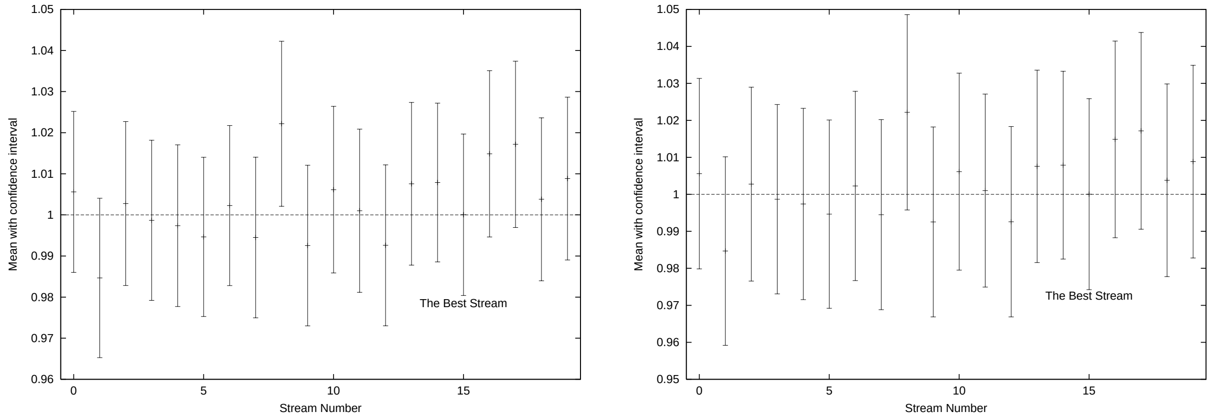
| | Mean | Variance | C.O.V. | 95% Conf. Interval | 99% Conf. Interval |
|---|---|---|---|---|---|
| Mean | 1.002757 | 1.013384 | 1.003781 | 0.019729 | 0.025929 |
| Stdev | 0.009069 | 0.030686 | 0.010852 | 0.000298 | 0.000391 |
| C.O.V. | 0.009044 | 0.030281 | 0.010811 | 0.015087 | 0.015088 |

Table 2: The Comparison of the 20 Streams

- *Metric 1*: the absolute difference between the sample mean and the true mean (which is 1 for exponential distribution). The smaller this metric is, the better a stream fits the exponential distribution.
- *Metric 2*: the absolute difference between the sample standard deviation and the true standard deviation (which is 1 for exponential distribution). The smaller this measure is, the better a stream fits the exponential distribution.
- *Metric 3*: the range of the confidence interval (e.g. the range of the 99% confidence interval). The narrower the range, the better the stream. Because the confidence interval is counted by the Equation 2, its range (i.e. upperbound minus lowerbound) is only related to the sample standard deviation. Therefore, a smaller sample standard deviation always implies a narrower confidence interval (vice versa).

In this report, first, visual test is used to gain an intuitive idea of which stream is the best. Figure 1(a) presents the mean and the 95% confidence interval of these 20 streams; while Figure 1(b) plots the mean and the 99% confidence intervals. Visually, there are no significant differences among the ranges of the confidence intervals of these 20 streams. Moreover, the No.15 stream can be considered as the best stream because in both graphs, its mean is the closest to the true mean 1. Then, the data listed in Table 1 is checked. It is found that the mean, and the variance of stream 15 are the closest to 1. Therefore, stream 15 is the best stream of these 20 streams.

Figure 2(a) shows the 95% confidence interval of stream 15 as a function of the population size. It can be easily seen from this figure that the mean gets closer to the true mean (i.e. 1.0) as the population size increases. Moreover, in this report, the confidence interval is computed using Equation 2, where $2\times z_{1-\frac{\alpha}{2}} s/\sqrt{n}$ represents the range of the confidence interval. Figure 2(b) plots the value $z_{1-\frac{\alpha}{2}} s/\sqrt{n}$ as a function of the




(a) Mean with 95% Confidence Interval

(b) Mean with 99% Confidence Interval

Figure 1: The Mean and the Confidence Intervals of the 20 Streams

population size. This figure shows that this value decreases as the population size increases, which indicates that the confidence interval becomes narrower as the increase of the population size. Therefore, it can be concluded that the confidence interval shrinks as the population size increases.

Figure 2(c) and Figure 2(d) present the 99% confidence interval versus the population size. These figures also confirm that the confidence interval narrows with the increase of the population size.

In order to explain the reason of this shrinkness, some notation is introduced:

- $\mu$: the population mean (in this report, the mean of the exponential distribution, which is equal 1).
- $\sigma$: the population standard deviation (in this report, the standard deviation of the exponential distribution, which is equal to 1).
- $x_1, x_2, \cdots, x_n$: a sequence of samples generated by a exponentially-distributed random variates generator (which constructs the stream used in this question), where $n = 500, 1000, 1500, \cdots, 10000$.
- $\bar{x}$: the mean of the n samples.
- $s$: the standard deviation of the n samples.

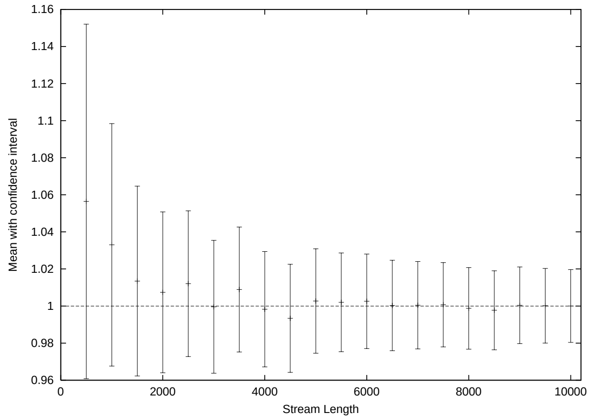
Based on the central limit theorem, if $x_1, x_2, \cdots, x_n$ come from the same population and are independent, the sample mean $\bar{x}$ for large samples is approximately normally distributed with mean $\mu$ and standard deviation $\sigma/\sqrt{n}$:

$$\bar{x} \sim N(\mu, \frac{\sigma}{\sqrt{n}})$$

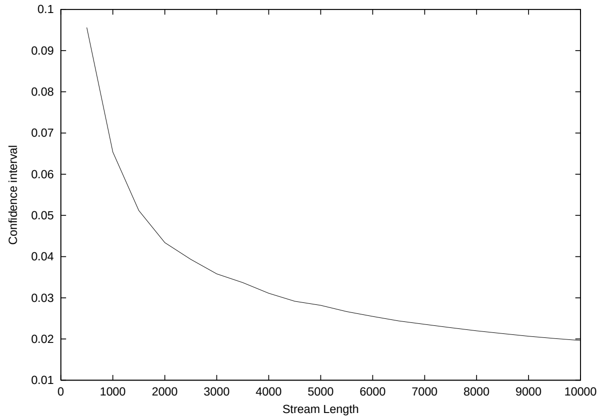
In this report, the sample sizes used can be considered as large samples sizes (all are larger than 500). Therefore, the sample mean follows the above normal distribution. The standard deviation of the sample mean (i.e. $\frac{\sigma}{\sqrt{n}}$) is called the **standard error**. From this expression, it is easy to see that as the sample size $n$ increases, the standard error decreases, which indicates that the sample mean gets closer to the true mean. Moreover, the $100(1-\alpha)\%$ confidence interval for the population mean is given by:

$$(\bar{x} - z_{1-\frac{\alpha}{2}} s/\sqrt{n}, \bar{x} + z_{1-\frac{\alpha}{2}} s/\sqrt{n})$$

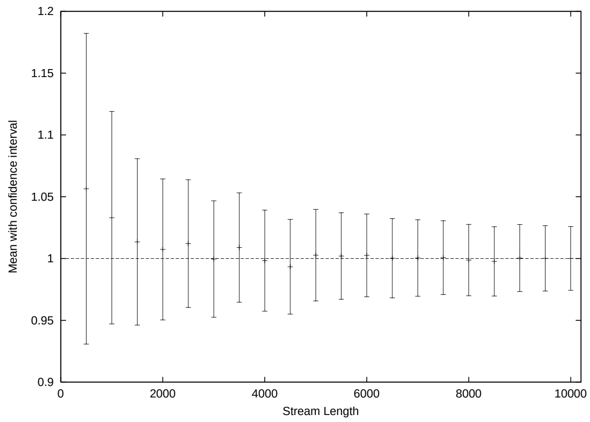
where $2 \times z_{1-\frac{\alpha}{2}} s/\sqrt{n}$ represents the range of the confidence interval. For large sample sizes, the sample standard deviation is relatively stable. Therefore, for a give confidence level (i.e. the same $z_{1-\frac{\alpha}{2}}$ value), as the sample size $n$ increases, the value of $z_{1-\frac{\alpha}{2}} s/\sqrt{n}$ decreases, that is, the range of the confidence interval shrinks.



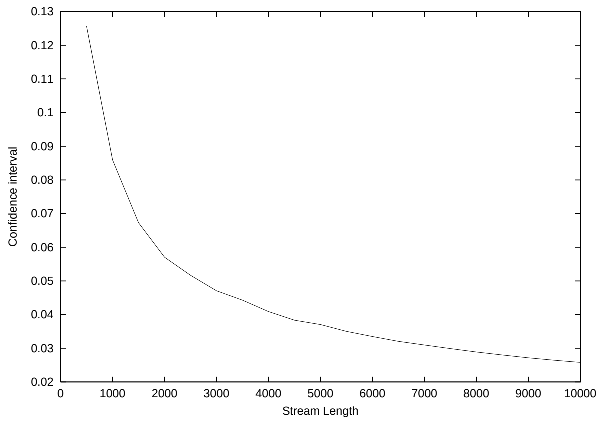

(a) Mean with 95% confidence interval




(b) 95% confidence interval




(c) Mean with 99% confidence interval




(d) 99% confidence interval

Figure 2: The Confidence Intervals with Different Sample Size

(c) [10 marks] *subdivide the stream used in part (b) into 5 sub-streams of 2000 values each. Determine the extent to which these sub-streams have the properties of the full stream. What does this tell you about the dangers of failing to provide separate streams for independent stochastic variables?*

In this question, the stream 15 is subdivided into five sub-streams. The $ith$ sub-stream is generated by taking the $(5j+i)^{th}$ $(where j = 0, 1, \cdots, 1999)$ elements from the original stream. Table 3 lists the means, the variances, the C.O.V., and the 95% and the 99% confidence intervals of the five sub-streams and the original stream, respectively.

Table 3 shows that the mean and the C.O.V. of all five streams are still close to 1. Therefore, these five sub-streams can still represent the exponential distribution. Moreover, both the 95% and the 99% confidence intervals are wider than those of the original full stream, which agrees with the results obtained in question 1(b) that the confidence intervals widen as the population size decrease.

Jain [Jai91, p. 454] stated that subdividing one stream "may result in a strong correlation between the two variables" if each is fed by a sub-stream. To verify this, the correlation coefficient between any two of these five sub-streams are listed in Table 4. For comparison, the correlation coefficient between any two streams of the first five streams generated in question 1(a) (see row 2-6 of Table 1) are listed in Table 5. It can been from the two tables that the correlation coefficients of the five sub-streams are not different

| Stream | Mean | Variance | C.O.V. | 95% conf. interval | 99% conf. interval |
|---|---|---|---|---|---|
| Sub-stream 1 | 1.039440 | 1.092740 | 1.005680 | (0.993623,1.085250) | (0.979224,1.099650) |
| Sub-stream 2 | 0.969157 | 0.947379 | 1.004310 | (0.926499,1.011820) | (0.913092,1.025220) |
| Sub-stream 3 | 1.006450 | 0.997803 | 0.992498 | (0.962672,1.050230) | (0.948913,1.063990) |
| Sub-stream 4 | 1.000730 | 1.046280 | 1.022130 | (0.955902,1.045560) | (0.941812,1.059650) |
| Sub-stream 5 | 0.984407 | 0.933080 | 0.981261 | (0.942072,1.026740) | (0.928766,1.040050) |
| Full Stream | 1.000040 | 1.003610 | 1.001770 | (0.980401,1.019670) | (0.974230,1.025840) |

Table 3: The Statistics of the Sub-streams and the Original Stream

from those of the long streams on a factor of magnitude. Therefore, we cannot say that there are strong correlations between the sub-streams.

Actually, if the random number generator can generate a real random variate, there would be no correlations between the sub-streams. However, if the random number generator were not good enough, the sub-streams may lose their randomness, and may have strong correlation with each other. When the sub-streams with correlation are used to represent independent variables, these performance results may be affected. For example, in a ATM switch with two inputs, the inter-arrival time of two input streams are two independent variables. If they have positive correlation, a higher cell loss ratio may be observed.

| Sub-stream | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | -0.056271 | | | |
| 2 | 0.008754 | -0.023963 | | |
| 3 | -0.028345 | -0.014190 | 0.040091 | |
| 4 | 0.013567 | 0.008703 | 0.043355 | -0.010393 |

Table 4: The Correlation Coefficient of the Sub-streams

| Full stream | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0.061657 | | | |
| 2 | -0.009004 | 0.046410 | | |
| 3 | -0.005264 | 0.028910 | -0.030673 | |
| 4 | 0.034558 | 0.065392 | 0.069375 | -0.016270 |

Table 5: The Correlation Coefficient of the Five Original Streams

2. [30 marks] *In class, we studied techniques for determining various aspects of simulation run length, in particular the length of a "warm up" period, and the length of the overall simulation. Discuss this issue. Why is this an important thing to do, and how do you do it? Give examples from work you have done to illustrate your points.*

**"Warm up" period**

In most simulations, two states exist: "initial-state" and "steady-state". At the beginning, the system is in the initial-state, where the system states (e.g. throughput, etc.) change with time. As time goes on, the system enters a steady-state, where the system state is relatively stable. Normally, only the **steady-state performance**, that is, the performance after the system has reached the steady-state, is of interest. Therefore, the results of the initial part of the simulation (i.e. initial-state) should be excluded from the final computations. The duration that the system is in the initial-state is also named "warm up" period. If the "warm up" period is included, the results of the simulation may be affected. One way to eliminate this effects of the initial-state is to run the simulation for a very long time. However, the resources may be unnecessarily wasted. One the other hand, if the "warm up" period is selected too large, useful simulation results may be removed. This may affect the correctness and the completeness of the results, and may result in the a waste of resources. Therefore, the proper selection of the "warm up" period is important to the simulation.

For example, in my research project "The Buffer Pool Management in DBMS", the buffer pool, which is a cache caching disk pages of databases, was studied. Figure 3 presents a typical throughput curve of the DBMS. The x-axis is the time (in minutes), and the y-axis shows the throughput of the DBMS. It can be easily seen from the figure that at the beginning, the throughput keeps increasing and reaches a peak value. After that time, the throughput drops rapidly until it reaches the steady state.

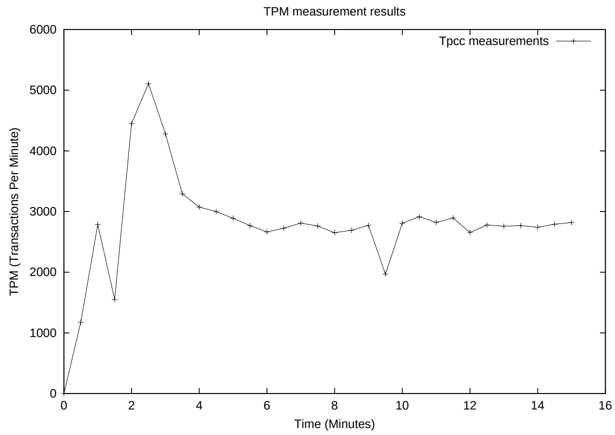



Figure 3: Buffer pool throughput and page distribution

Several methods about removing the initial state are discussed in [Jai91, p. 424–428]. All of these methods are based on a reasonable assumption that the variability during the steady state is less than that during the transient state. The "Truncation" and the "Batch means" methods will be discussed here. For the example of Figure 3, the throughput data are listed in Table 6.

| Time | Throughput | Time | Throughput |
|---|---|---|---|
| 0 | 0 | 8 | 2652 |
| 0.5 | 1176 | 8.5 | 2692 |
| 1 | 2788 | 9 | 2772 |
| 1.5 | 1550 | 9.5 | 1968 |
| 2 | 4450 | 10 | 2808 |
| 2.5 | 5110 | 10.5 | 2914 |
| 3 | 4278 | 11 | 2822 |
| 3.5 | 3292 | 11.5 | 2894 |
| 4 | 3074 | 12 | 2656 |
| 4.5 | 3000 | 12.5 | 2778 |
| 5 | 2888 | 13 | 2758 |
| 5.5 | 2768 | 13.5 | 2768 |
| 6 | 2664 | 14 | 2742 |
| 6.5 | 2726 | 14.5 | 2792 |
| 7 | 2810 | 15 | 2818 |
| 7.5 | 2762 | | |

Table 6: Throughput of the DBMS Buffer Pool

In the "Truncation" method, the observations are truncated from the beginning until the first number is neither the minimum nor the maximum of all observations left. After applying this method to the data listed in Table 6, the first four observations (0–1.5) are considered as "warm up" which is not enough. This shows that the "truncation" method is too simple to get the correct "warm up" in some cases. However, combining this method with the visual examination of the simulation results can often generate satisfied results.

Another example comes from the assignment 2 of this class. A client/server two-level file cache was simulated in this assignment. The goal is to compare the read miss-ratio of different cache replacement algorithms on the server cache. Figure 4 shows the change of the miss ratio as a function of time. At the beginning, the cache is empty. As the cache is filling with disk pages, the miss ratio goes down. Visual examination can hardly determine the "steady-state", because the miss ratio is not stable until the end of the simulation.

Before further discussion, some notation is introduced:

- $T$: the total number of traces.
- $M$: the total number of read miss.
- $W$: the length of the "warm up" period.
- $M_w$: the number of read miss during the "warm up" period.

Therefore, the read miss ratio is computed by: $\frac{M-M_w}{T-W}$.

First, whether the "warm up" period is needed is discussed. One opinion is that "warm up" is not needed in this case. The reason is that if a "warm up" period is used, the numbers of read miss in $M_w$ for different cache replacement algorithms are different, and the cache layouts at the time that the "warm up" finishes are also different for different replacement algorithms. Therefore, the "warm up" period has different effects on different algorithms, and the computation of read miss-ratio $\frac{M-M_w}{T-W}$ is not fair.

Another opinion is that "warm up" is still necessary. The read miss ratio when the cache is not full would be bigger than that after the cache is full. Therefore, the read miss ratio before the cache is full cannot represent the properties of the algorithm. If this "warm up" period $H$ is not removed, the final results may be affected by the $M_w$. Although the cache layouts, at the time the "warm up" finishes, are different for different replacement algorithms, this does not affect the fairness of the comparison. The cache layout for different algorithms are always different (except for empty cache). This is caused by the replacement algorithm, and represents the properties of the algorithm.

I prefer the second opinion. The "batch means" method was used to determine the length of "warm up". The variances of the batch means are computed and the result is plotted in Figure 5. After the last peak at 2400, the system enters steady state. Therefore, the "warm up" period is 2400.

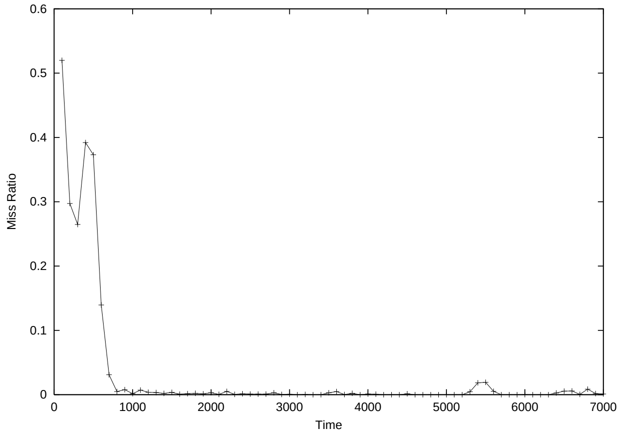



Figure 4: Server Miss Ratio of LRU algorithm

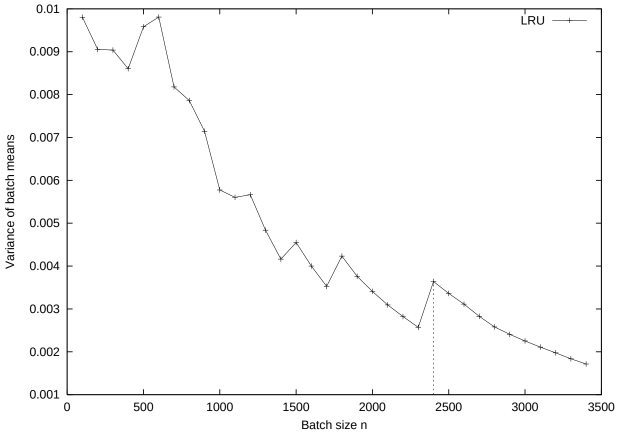



Figure 5: Warmup determination by batch means

**Length of the overall simulation**

The length of the overall simulation should be long enough to keep the variance in the desired level. If the length is not long enough, the simulation results may have too much variance and cannot be used as an estimation of the true values. If the simulation length is longer than necessary, the resources are wasted. Therefore, how to spend minimum resources to achieve minimum variance is important.

Some methods for determining the length of the simulation are discussed in [Jai91]. The "independent replications" method need to run the simulation many times. If the simulation is trace-driven simulation, and only one trace is available, running the simulation repeatedly can only get identical results. Therefore, the “independent replications” cannot be used in this case. The two samples discussed above are in this category.

The “batch means” method does not require replications of simulation run. The long simulation run is divided into batches to reduce the variance. The computation needed is the same as the “batch means” method in the “warm up” period computation. The new step is that the **autocovariance** of the batch means is computed. Different batch sizes are tested and the batch size which makes $\frac{autocovariance}{variance\ of\ the\ batch\ means}$ achieve the minimum value is selected, and the confidence interval for the mean can be minimized.

3. [30 marks] *The attached paper, entitled “TCP Vegas Revisited”, was presented at the* **Infocom 2000** *conference. Provide a detailed commentary on this paper as an example of the application of experimental methodology.*

The outline of this commentary is based on the systematic approaches to performance evaluation discussed in [Jai91, p. 22]

1. **Goals and the Definition of the System**.

   TCP Vegas is a new design for TCP. It includes several innovative techniques. It has been reported that TCP Vegas can achieve an impressive performance gains compared to TCP Reno. Because TCP Vegas is essentially a combination of several different techniques, the objective of this paper [HBG00] is to investigate which of these techniques are responsible for the performance gain.

   The system studied in this paper is shown in Figure 6. The network performance between the two hosts is considered. The network topology and some of other system parameters (such as the segment size, the router queue size, and so on) are all the same as those employed in the original Vegas paper [BOP94]. By doing this, the authors can ensure that their results are comparable to the previous work.


Figure 6: System Boundary

2. **Methodology**

   This paper uses trace-driven simulation to conduct the performance evaluation. The workload used for the simulator is synthetic trace, whose characteristics can be controlled by the parameters including segment size, connection inter-arrival time, etc.

3. **Simulator**

   In this paper, the authors did not develop their own simulator. All their simulation ran on a well-developed network simulator called *x-sim*. The reason of using this simulator is: (a)the experiments in the original TCP Vegas paper were also performed on this simulator, therefore, the author can ensure that their results do not bias the different implementation of TCP Vegas; (b)the implementation of TCP Reno in *x-sim* derived from some production codes (BSD), therefore, the results the authors obtained are realistic.

   In this paper, the authors also made some changes to the *x-sim*. For example, they modified the TCP Vegas source code to separate the various algorithms from each other and to allow each of the algorithms to be selectable individually; they also fixed some problems in the original implementation of the TCP Vegas. In order to maintain the validity of the simulator, the authors always tried to keep the changes to the minimum. Moreover, after each modification, they conducted experiments to ensure that these changes do not impact the other parts of the simulator. Therefore, the simulator used in this paper was well developed, verified and validated.

4. **Metrics**.

   The metrics used in the paper are *average throughput (KB/s) per connection* and *average retransmissions (KB) per connection*. Another commonly used metric for TCP is the fairness, which is discussed in the paper (e.g. section IX.A). However, the authors do not use it as a metric.

5. **Factors and Levels**.

   The factors in the paper are different algorithms that the are new in TCP Vegas. The authors concluded 10 such algorithms. The levels of these factors are "on" and "off", where "on" means an algorithm is used and "off" means it is turned off. In this paper, each algorithm is considered as an addition to the TCP Reno. Moreover, it is also required that each of them can be independently turned on or off. The authors validated this implementation by turning all algorithms "off" to make sure that it produced the same results as the TCP Reno; and turning all algorithms "on" to check that it achieved identical throughput with the original implementation.

   All these ten factors are the objects for study. Because the $2^k$ factorial design will be used, the number of factors $k$ should be kept small. By $2^{10} - 1$ preliminary tests on all 10 factors, it was found that the influence of most combinations was indeed smaller than 1%. To reduce the complexity and increase the "expressiveness" of our experiments, the 10 factors are clustered into the following three groups: *Slow-start, Congestion avoidance*, and *Congestion recovery.* Therefore, only three factors need to be considered.

6. **Experimental Design**.

   As stated above, the $2^k$ factorial design was applied for the three factors. This methodology was used to test the effects of each factor and the effects of every combination of any number of factors.

   Each combination of the factor levels are tested, and the result for each experiment are recorded. $2^3 - 1 = 7$ experiments are needed for three factors. After solving a $7 \times 7$ system of equations, the seven "effect powers" of each factor and factor combination are available.

7. **Data interpretation and results presentation**.

   From the values of the parameters computed above, it is easy to see the effect of each factor. The results show that TCP Vegas's techniques for slow-start and congestion recovery have the most influence on throughput. The reasons of these observations are also discussed.

8. **Overall comments**.

   This is an excellent paper on TCP Vegas performance evaluation. The authors investigated the effect of every new techniques incorporated into TCP Vegas by the $2^k$ factorial design.

   1. Comparability of the Results

   The authors in this paper intend to investigate the effects of innovative techniques integrated in the TCP Vegas on the performance gains. They try to reconstruct an experiment environment similar to that employed in the original paper. For example, they use the same network topology, the system parameters, and the same simulator (with minor changes). The advantage of this reconstruction is that their results are comparable to the previous results, thereby making the results more persuasive.

   2. The Development and the Validation of the Simulator

   In this paper, the authors perform their simulation on an existed simulator. Actually, when evaluating the performance of a system as complex as the TCP protocol, developing the simulator from the scratch is impractical. Adopting simulators have been developed and validated is a good choice. This not only saves a lot of time and efforts, but also makes plenty previous results available for the verification of the new simulator.

   In most cases, the existed simulators need to be changed, corrected, or extended to satisfy the new requirements of the current experiments. At this time, maintaining the validation of the simulator is a critical issue. This paper does a good job in this aspect. Before making the modifications, the authors analyze the problem and inspect the original simulator thoroughly so that all the changes are carefully planed and controlled. Although they needed to make many changes to the original simulator, they tried to keep these corrections and extensions to a minimum extent to avoid any behavioral differences between the current and the original simulators. Moreover, each time the authors completed some changes, they validated their new version of the simulator against the old version.

   3. The Reduction of Complexity

In preliminary tests, all ten factors are tested. However, most factors only have very small influence on the overall performance. It is impossible to present the $2^{10}$ factorial experiment in a paper. Therefore, these factors are reduced to three categories, and the number of parameters are reduced to a manageable number.

4. Experimental Design

In the preliminary test, the $2^{10}$ factorial experiment are performed. However, this requires 1023 different experiments. Moreover, the $2^7$ experiment is also performed by disabling the last three factors. The $2^{10}$ experiment contains the $2^7$ experiment. Therefore, it should be unnecessary to perform the $2^7$ experiment. Performing the $2^{10}$ experiment may be too expensive. A careful selection and reduction of factors should be performed before performing such an expensive experimental design. For example, the factors can be separated into $k$ groups and a $2^k$ experiment is performed. Based on this result, a more detailed experiment can be performed only on the factor group with significant effect.

5. Experiment Errors

In all data shown in this paper, the experiment error ranges from 12% to 92%, and half of them are more than 40%. The authors did not explain where this error came from, whether they mattered the result, or how to reduce these errors.

# References


[BOP94] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. Tcp vegas: New techniques for congestion detection and avoidance. *ACM Comp. Commun. Review (SIGCOMM'94), Conference Proc.*, pages 234–244, 1994.

[HBG00] U. Hengartner, J. Bolliger, and Th. Gross. TCP Vegas Revisited. In *Proceedings of the IEEE INFOCOM*, Tel-Aviv, Israel, March26–30 2000. IEEE.

[Jai91] Raj Jain. *The art of computer systems performance analysis.* John Wiley, New York, 1991.