July 1988

UILU-ENG-88-2238
CSG-90

COORDINATED SCIENCE LABORATORY
College of Engineering

MEASUREMENT-BASED ANALYSIS OF MULTIPLE ERRORS AND NEAR-COINCIDENT FAULT DISCOVERY IN A SHARED MEMORY MULTIPROCESSOR

Samir G. Mitra
Ravishankar K. Iyer
Mark Sloan

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS: None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): UILU-ENG-88-2238 (CSG-90)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Coordinated Science Lab University of Illinois — 6b. OFFICE SYMBOL *(If applicable)*: N/A	7a. NAME OF MONITORING ORGANIZATION: NASA
6c. ADDRESS *(City, State, and ZIP Code)*: 1101 W. Springfield Avenue Urbana, IL 61801	7b. ADDRESS *(City, State, and ZIP Code)*: NASA Langley Research Center Hampton, VA 23665
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: NASA — 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: NASA-NAG-1-613
8c. ADDRESS *(City, State, and ZIP Code)*: see 7b.	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
"Measurement-Based Analysis of Multiple Errors and Near-Coincident Fault Discovery in a Shared Memory Multiprocessor"

12. PERSONAL AUTHOR(S)
Mitra, S.G., Iyer, R.K., and Sloan, Mark

13a. TYPE OF REPORT: Technical	13b. TIME COVERED FROM ________ TO ________	14. DATE OF REPORT *(Year, Month, Day)*: February 1988	15. PAGE COUNT: 37

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
parallel processing, multiple errors, simulation probability distributions, near-coincident faults

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*
This paper presents a methodology to study multiple error presence and near coincident fault discovery in the memory of a shared memory multiprocessor. The delay between the generation of an error due to a fault and its detection (error latency) can cause multiple errors and near-coincident fault discovery in a system. The latter effect is widely known to be catastrophic to the continued operation of a system but very little research has been done to understand its behavioral characteristics. The methodology is illustrated on the Alliant FX/8, the basic cluster component of the Cedar Supercomputer at the Center for Supercomputing Research and Development at the University of Illinois. Experimental results are provided under real concurrent workload conditions over a five-day period. This study finds that for a conservative error occurence rate of one error a day, there is a 25% chance that latent errors cause a multiple error condition. Thus, one out of four errors may manifest itself as a multiple error. At the same error occurrence rate, it was found that 8% of the error manifestations are near-coincident in nature for (over)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE *(Include Area Code)* — 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. abstract (continued)

a time-window size of 50 microseconds (approximately 250 instruction cycles on the Alliant FX/8). It was seen that the probability of multiple errors tends to saturate after a threshold error occurrence rate of approximately one error a day. A high degree of similarity between behaviors of multiple errors and near-coincident fault discovery suggested strong correlation between existences of multiple errors and their discovery in near-coincidence.

MEASUREMENT-BASED ANALYSIS OF MULTIPLE ERRORS AND NEAR-COINCIDENT FAULT DISCOVERY IN A SHARED MEMORY MULTIPROCESSOR

Samir G. Mitra
Ravishankar K. Iyer
Mark Sloan

February, 1988

Computer Systems Group
Coordinated Science Laboratory
University of Illinois
1101 W. Springfield Ave.
Urbana, IL 61801

ABSTRACT

This paper presents a methodology to study multiple error presence and near-coincident fault discovery in the memory of a shared memory multiprocessor. The delay between the generation of an error due to a fault and its detection (error latency) can cause multiple errors and near-coincident fault discovery in a system. The latter effect is widely known to be catastrophic to the continued operation of a system but very little research has been done to understand its behavioral characteristics. The methodology is illustrated on the Alliant FX/8, the basic cluster component of the Cedar Supercomputer at the Center for Supercomputing Research and Development at the University of Illinois. Experimental results are provided under real concurrent workload conditions over a five-day period. This study finds that for a conservative error occurrence rate of one error a day, there is a 25% chance that latent errors cause a multiple error condition. Thus one out of four errors may manifest itself as a multiple error. At the same error occurrence rate, it was found that 8% of the error manifestations are near-coincident in nature for a time-window size of 50 microseconds (approximately 250 instruction cycles on the Alliant FX/8). It was seen that the probability of multiple errors tends to saturate after a threshold error occurrence rate of approximately one error a day. A high degree of similarity between behaviors of multiple errors and near-coincident fault discovery suggested strong correlation between existences of multiple errors and their discovery in near-coincidence.

ACKNOWLEDGMENTS

Acknowledgments are due to Ed Davidson, Tracy Tilton, Mark Washburn, Madhu Sharma, Al Malony, Pen Yew, Mike Haney and the other members at the Center of Supercomputing Research and Development for their support and many useful discussions during the course of this work. This work was supported by the National Aeronautics and Space Administration under NASA grant NAG-1-613.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SECTION 1

INTRODUCTION

The reliability and availability issues have become key aspects in a wide range of computer system design methodologies. A prerequisite in designing for reliability is to understand the effect of faults and their manifestation mechanisms. Behavior of faults in a computer system is not easy to realize or understand. This is even more so in a multiprocessing environment, where the number of ways in which error conditions may manifest themselves is usually incomprehensible. Analytical modeling techniques suffer from constraining assumptions and developmental complexity. Alternatives are measurements and experiments on production multiprocessor systems. These aid the model building process and provide valuable insight for designing new systems.

This paper studies the fault discovery process in a shared memory multiprocessor system. There is usually a delay between the generation of a error (caused by a fault) and its discovery by a detection mechanism. This time is commonly referred to as error latency. These latent errors (sometimes referred to as "lurking errors") can be major threats to the reliability of the system. This is so because if there exists more than one latent error there is a possibility that they can be discovered simultaneously behaving as though multiple faults have occurred. Most recovery mechanisms however are not designed to handle multiple faults. Latent errors have been observed to be discovered close in time to each other, thus, stressing the error recovery mechanism. Such situations are referred to as near-coincident fault discovery and are known to be catastrophic in real systems [1,2].

The purpose of this experimental study is to quantify the characteristics of multiple errors and near-coincident fault discovery in a shared memory multiprocessor system under a real concurrent workload[1]. A multiprocessing system presents new complications from the workload point of view, since a number of processes can be active at the same time. This casts a new perspective on the study of latent error behavior as the probability of error discovery is potentially high.

The experiment employs actual hardware measurements from an Alliant FX/8 system to simulate error occurrence in the system and to investigate multiple error occurrence and near-coincident fault discoveries. The Alliant FX/8 is a key component in the "Cedar" parallel supercomputer project at the Center for Supercomputing Research and Development in the University of Illinois at Urbana-Champaign [3]. The measured Alliant FX/8 runs the current version of "Xylem," the Cedar operating system. Specifically, the methodology is applied to the Alliant memory subsystem. The fault model used in this study assumes that a permanent error has already occurred[2]. The physical mechanism causing the faults can be varied and do not affect the results.

The results are unique in that they provide new insight into the behavior of multiple errors and near-coincident fault discovery in a complex parallel processing environment. At a conservative error occurrence rate of approximately one error a day, there is a 25% chance that latent errors cause a multiple error condition in the system. Thus one out of four errors may manifest itself as a multiple error. Further it was found that 8% of the error manifestations are near-coincident in nature at the same error occurrence rate for a time window size of 50 microseconds. It was also found that the

[1]When two or more processors are active the system is said to be in concurrent operation.

[2] An error is that part of the system state which is liable to lead to failure. The cause - in its phenomenological sense - of an error is a fault.

probability of multiple errors tends to saturate after a threshold error occurrence rate of approximately one error a day. Although it was seen that the near-coincident fault discovery behavior has a high correlation with multiple error behavior, the saturation effect on the probability of near-coincident fault discovery becomes apparent at a higher error occurrence rate.

1.1 Related Research

There is little or no research cited in the literature which experimentally investigates the occurrence of multiple or near-coincident fault discovery. Fault injection studies in the FTMP (Fault Tolerant Multiprocessor) showed that the most likely threat to system failure in the short run was arrival of two failures so close to each other that system reconfiguration was not possible [1,2]. These experiments related to specific programs and involved pin-level fault injection. An analytical model for near-coincident faults in NMR systems with different voting schemes is presented in [4]. The general validity of such a model however is not established.

Other related research includes experiments conducted to measure fault/error latency. Experiments to measure fault latency via pin-level fault injections in FTMP are discussed in [5]. In that study, the researchers measured latency times for faults in different system components and obtained a standard distribution fit to their measured fault latency distribution. CPU fault latency for the digital microprocessor in FTMP was studied in [6,7] via gate-level simulation. A set of specific programs was used to exercise the CPU to reveal faults injected into the simulation.

The above approaches and results are, however, not applicable in general to multiuser systems. More recently, latent fault behavior in the memory of a VAX 11/780 was studied in [8]. The memory system was instrumented for measurements, and fault/error latencies were calculated by simulated fault injection in the memory. The effect of workload on fault/error latencies was investigated in [9].

Although the above studies investigate the subject of latency quite systematically, the question of multiple errors or near-coincident fault discovery is not addressed. Given that several past measurements indicate that these problems are usually catastrophic to the system, points toward a great need for an investigation given by this thesis.

Section 2 describes the experimental methodology used to calculate the multiple error and near-coincident fault discovery probabilities. Section 3 presents results and discusses the multiple error and near-coincident fault discovery behavior seen. Section 4 is the concluding section, which highlights important results of the paper.

SECTION 2

EXPERIMENTAL METHODOLOGY

The measured system is an Alliant FX/8, a shared memory multiprocessor. Figure 1 shows the Alliant FX/8 components related to our study. The system runs the current version of "Xylem," the Cedar operating system. From the software point of view, many features of the Cedar supercomputer are running on the Alliant FX/8. Detailed information on the Alliant FX/8 is given in [10, 11]. The workload on the Alliant FX/8 consisted mostly of scientific applications such as circuit simulation, weather modeling, digital animation and fluid dynamics.

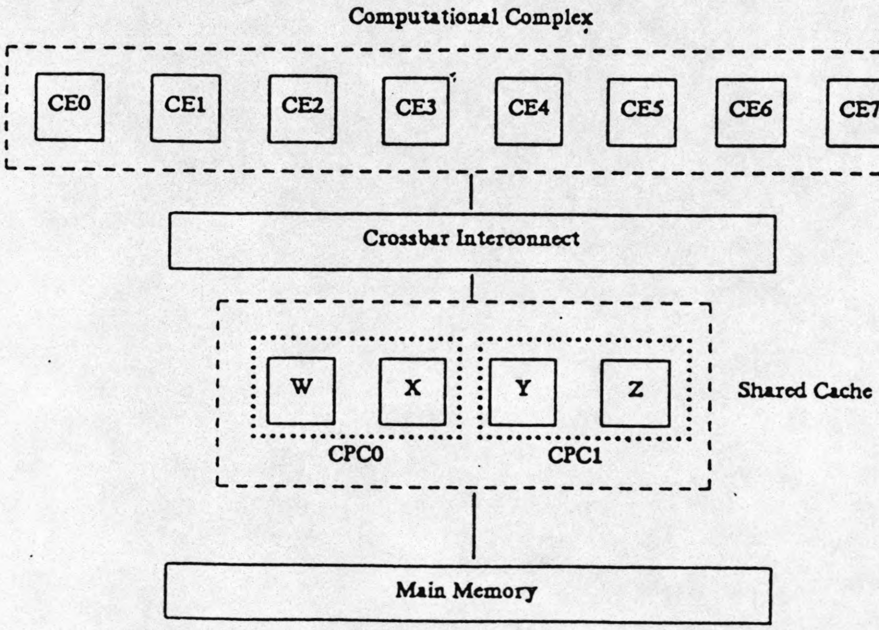

Figure 1. Configuration of Measured Alliant FX/8

We investigate the failure characteristics of the main memory. An important reason for this is that measured field results show the largest number of failures occur in the memory [12]. A large number of CPU errors can also be traced to the memory [13]. As shared memory is a common resource, the possibility of it being the source of failures is significant.

2.1 Hardware Measurements

The Alliant FX/8 backplane was sampled to collect data on memory access operations from the shared cache. A Tek DAS 9200 with a 32K trace buffer was used for this purpose [14]. The hardware probes were attached primarily to the main memory address bus on the backplane of the system. Other probes were used to monitor signals so that appropriate triggering could be performed.

As mentioned in the introduction, the measurements were performed while the system was executing concurrent workload. The measurements were conducted over a five-day period, 8am to 5:30pm daily, Monday to Thursday and 8am to 3:45pm on Friday (primarily due to drop in concurrent operations). Samples were taken approximately every 4 minutes[3], each containing 8K address references (representing 8K machine cycles). The total measurement period was approximately 46 hours.

Table 1 shows the filtered version of the raw data output. Addresses represent memory block start addresses. The memory is accessed in blocks of 32 bytes (transfer size between the shared cache and memory). The fields cntl0 and cntl1 provide

[3] The sampling rate chosen reflects a compromise between an adequate sample size and delay in transferring data to a data logger.

Table 1. Concurrent Workload Memory Address Trace

Line no.	time stamp	address	cntl0	cntl1
1	00033316579	0D3FF8	F	0
2	0003331666B	000230	F	4
3	000333166BC	000232	F	4
4	0003331670D	000234	F	4
5	000333167BB	0D3FF7	F	0
6	000333167CC	1AE0F7	F	8
7	00033316869	0C2FF5	F	0

additional status information about the state of the memory bus.

2.2 Simulation

The memory address trace obtained above was then used as input to a simulation system, which essentially reconstructed the address space into which simulated error injections are performed over the entire measurement period (the simulator is driven by the address trace). An error is discovered when the time of error injection at an address location is less than or equal to the time of arrival of that address in the concurrent workload address trace. By observation of the discovery behavior of these faults , the simulator provided results on the probability of multiple errors and near-coincident faults.

For error injection purposes no distinction was made between specific locations within a block. Since the transfers from main memory occur in blocks of 32 bytes, an error in one location within the block is equivalent to an error in any other location in the block from a discovery point of view. This greatly simplified the simulation and smoothed out discontinuities arising out of the fact that the data were sampled.

Simulated error occurrence (i.e., error injection) were performed assuming an exponential distribution for error occurrence over the entire measurement period. The error injection rates (λ) were varied from 0.009 to 0.058 (x6 error occurrences per hour). Address locations for error injection were randomly chosen. The exponentially distributed intervals between error injections were also randomly chosen .

In order to obtain statistically consistent results, approximately 600 faults were injected at each error injection time. This is equivalent to the simulation being run 600 times for each error injection rate. In each run, a randomly chosen location is injected with an error.

2.3 Measurement of Multiple Errors

Multiple errors occur when two or more errors are yet undiscovered in the system. In order to determine the probability of multiple errors at a given error injection rate, we first construct a latency profile for each injection. The latency profile for an injection is the discovery time profile for all errors injected at that injection time. Once the time to the discovery of each error injected is available, a latency profile can be plotted as in Figure 2.

Consider for simplicity a case in which two error injections are made in the measurement period. Figure 3 shows the three possible latency profile overlaps. Note that at each injection time a number of errors are injected into the memory. The multiple error regions between the two error injections are shown. Errors whose discovery latencies do not lie within the multiple error region are those that do not exist as multiple errors in the system. The multiple error region area versus the total latency

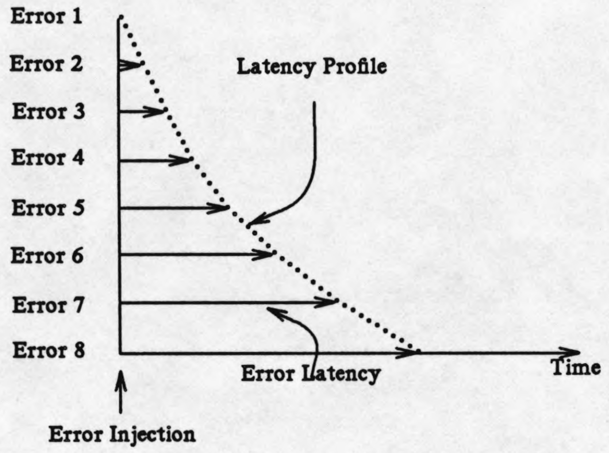

Figure 2. Example of a Latency Profile

profile area of both injections give a rough view of the probability of multiple errors in the system. The probability of multiple errors would be the ratio of number of errors in the multiple error region and the total number of errors injected.

Let E_i represent the number of errors injected at error injection number i. Also let $Me_{ij\lambda}$ represent the number of errors of error injection i that exist as multiple errors with error injection j at the error occurrence rate λ (e.g., $Me_{12\lambda}$ represents number of errors in injection 1 that exist as multiple errors with injection 2 at the error occurrence rate λ and $Me_{21\lambda}$ is the number of errors in injection 2 that exist as multiple errors with injection 1 at the error occurrence rate λ). Then between two error injections n and m where $n < m$, the probability of multiple errors $Mp_{nm\lambda}$ for an error occurrence rate of λ is

$$Mp_{nm\lambda} = \frac{Me_{nm\lambda} + Me_{mn\lambda}}{E_n + E_m}$$

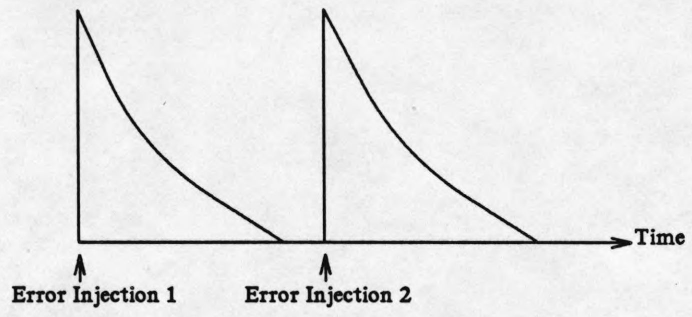

Figure 3(a). No Overlap

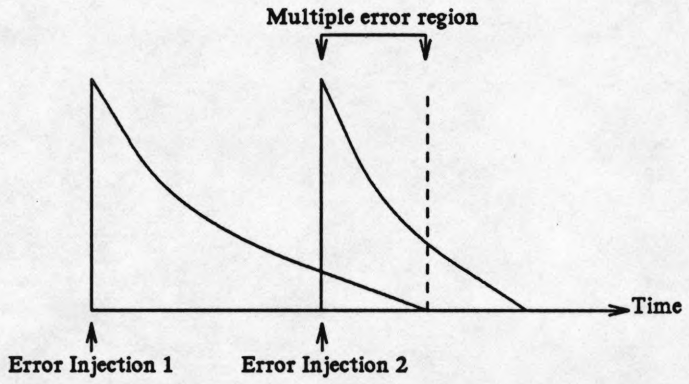

Figure 3(b). Type 1 Overlap

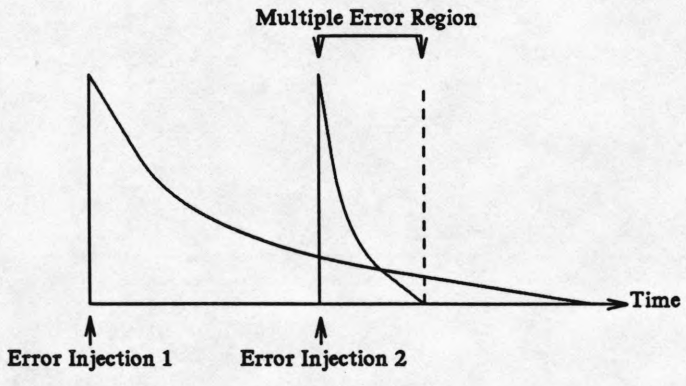

Figure 3(c). Type 2 Overlap

In the complex case of more than two error injections within the measurement period, the multiple error probabilities can be individually calculated with respect to one particular error injection for the given error occurrence rate λ (i.e., $Mp_{12\lambda}$ is a multiple error probability between fault injections 1 and 2, $Mp_{13\lambda}$ is multiple error probability between 1 and 3 etc.). For each $Mp_{nm\lambda}$ the multiple errors may exist in either of the two forms shown in Figures 3(b) and 3(c). But given the definition of multiple errors ,where at least two errors must exist undiscovered in the system, only adjacent error injection probabilities need be considered. Thus only $Mp_{12\lambda}$, $Mp_{23\lambda}$, $Mp_{34\lambda}$ etc. values are used to give an overall multiple error probability (Mp_{λ}) for the error occurrence rate chosen. Thus, if n_{ei} represents the number of error injections achieved at the error occurrence rate λ, the overall probability of multiple errors at error occurrence rate λ is

$$Mp_{\lambda}=\frac{\sum_{i=1}^{i=n_{ei}-1} Mp_{(i)(i+1)\lambda}}{n_{ei}-1}$$

2.4 Measurement of Near-Coincident Faults

In order to measure the probability of near-coincident fault discoveries, we choose an appropriate time window of size T. Next we moved this window over the total measurement time in increments equal to T, each time observing the number of errors (from different error injections) discovered within the time window. The ratio of total number of errors found in that time window to the total number of errors injected gave the probability of near-coincident faults in the system. Note that if errors from the same error injection were discovered within the time window, they do not qualify as a near-coincident fault discovery.

The total measurement period is divided into n time slices t_1 ... t_n, each T long except one (if true integer division is not possible). The number of errors discovered (from different error injections) in each t_k were $N_{k\lambda}$ at an error occurrence rate λ where $1 \leqslant k \leqslant n$. Again n_{ei} represents the number of error injections achieved at error occurrence rate λ. If the total number of errors injected into the system is E, then, the probability of near-coincident faults (NC_λ) for an error occurrence rate of λ is

$$NC_\lambda = \frac{\sum_{i=1}^{i=n} N_{i,\lambda}}{E}$$

$$\text{where } E = \sum_{i=1}^{i=n_{ei}} E_i$$

SECTION 3

RESULTS

This section presents results of multiple error and near-coincident fault behaviors seen on a shared memory multiprocessor (Alliant FX/8) memory subsystem. Recall that errors were injected at exponentially distributed intervals (with an error injection rate λ). The memory address trace was then used for determining multiple error and near-coincident fault discovery probabilities. For purposes of this study, errors were injected in the high usage regions of the memory. The region of injection represented 96% of the address references in the real concurrent workload trace but occupied only an eighth of the memory address space available. Clearly, the behavior of faults in this region is more critical for continued system operation.

On the average, 14% of all injected errors remained undetected during the measurement period (approximately 5 days). The choice of the error injection rate λ for the experiment was chosen to reflect realistic error occurrence rates (see [10]) The range was chosen to be $0.009 \leqslant \lambda \leqslant 0.058$ (x6 error occurrences per hour - approximately 2 to 16 error injections over 5 days). The time-window sizes chosen for analysis in the near-coincident fault discovery calculations represent reasonable error recovery times for a high performance system. The time-window range was varied from 1 microsecond to 250 microseconds (approximately 6 to 1500 instructions on the Alliant FX/8 [9]).

Figures 4(a) and 4(b) show the error latency distribution of detected errors at an error occurrence rate of 0.03 (x6 error injections/hr or ei/hr). In Figure 4(a) errors are injected at the start of the measurement period (detected errors = 549) whereas in Figure 4(b) errors are injected near the beginning of the third day of measurement (detected errors = 497). From Figure 4(a), we find the distribution has three distinct peaks. The first peak occurs during the beginning of the first day of measurement. The second and third peaks also occur during the beginning of the second and third days of measurement, respectively. Close to the start of the working day there is a sudden increase of workload on the system. This leads to a high number of error discoveries. At the first peak there are some error discoveries due to the injection of errors at that point. This behavior was also seen by [8] where error latencies were measured on the uniprocessor system (VAX 11/780). Contrary to the variation observed by [8] during midday hours, Figure 4(a) does not show any significant variation during these hours. The conjecture is that during midday hours the workload is more stable. This is primarily due to assignment of large processes (which run longer) on the CE's during midday hours. Figure 4(b) also shows the same behavior for errors injected close to the beginning of the third day of measurement. The difference in the behavior of latent errors on a multiprocessor system versus that for an uniprocessor system is closely related to the difference of workload characteristics for both types of systems.

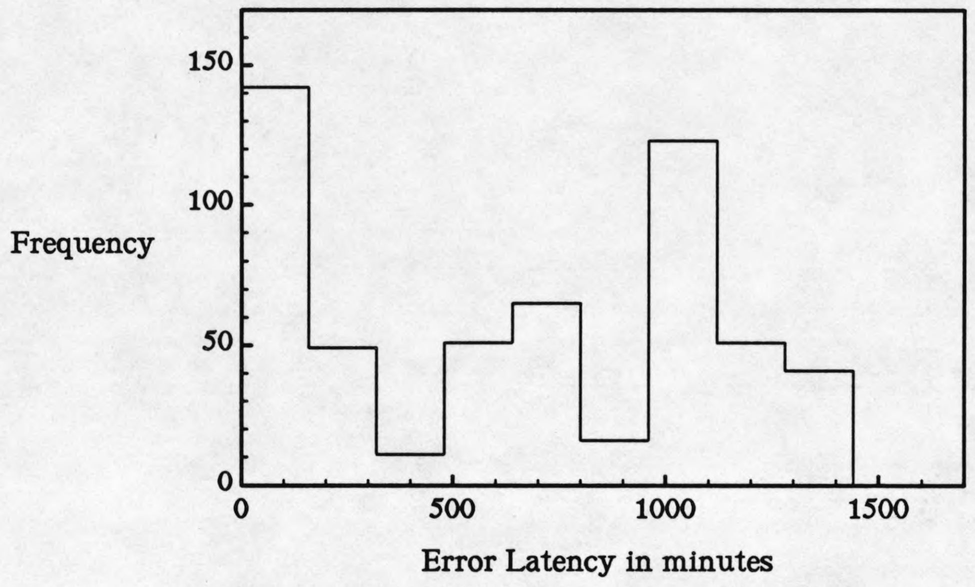

Figure 4(a). Error Latency Distribution at Measurement Time= 0.0 hr

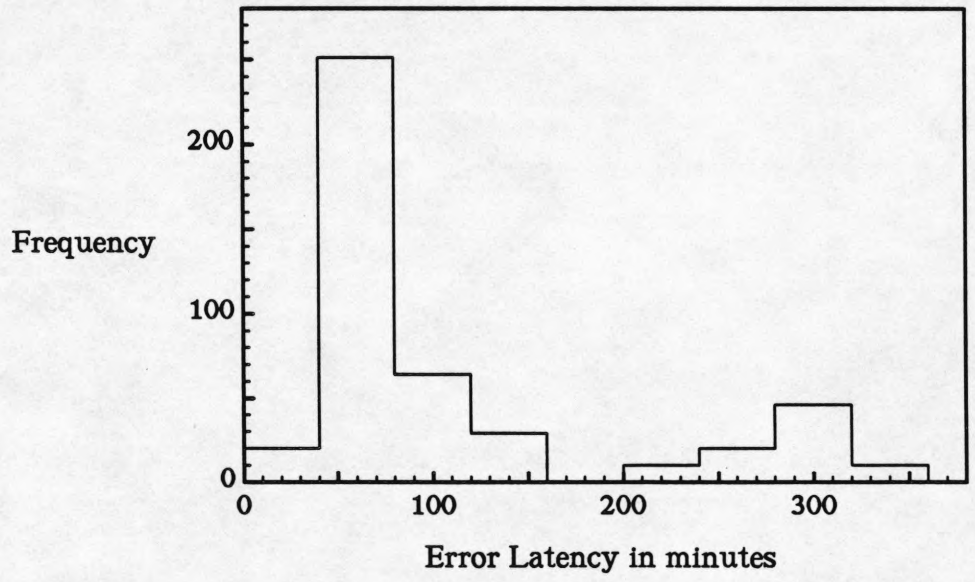

Figure 4(b). Error Latency Distribution at Measurement Time= 20.03 hr

3.1 Multiple Errors

Figure 5 shows the variation in the probability of multiple errors during the measurement period for an error occurrence rate of approximately two errors a day (0.043 x6 ei/hr). Figure 6 shows the probability of multiple errors being present in the system at different error occurrence rates. We find that the probability of multiple errors increases from a low of 0.04 at an error occurrence rate of approximately one error every two days to a high of 0.50 which is more or less a saturation probability. The oscillatory behavior of the graph is primarily due to statistical variations.

Figure 6 shows, at a conservative error occurrence rate of approximately one error a day (λ=.022), there exists a 25% chance that latent errors will cause a multiple error condition. This suggests that one out of four errors may manifest itself as a multiple error ($Mp_{\lambda} > 0.25$ for one error a day). At higher error injection rates, the multiple error probability is high enough to be potentially hazardous. Examining the plot in Figure 6 , we find at low error injection rates that plot has a higher slope than at high error injection rates. As expected the error occurrence rate (or the number of error injections) does have an impact on the multiple error probability, but this effect subsides as the error occurrence rate increases. The threshold is not high, about four errors in a 5-day period (i.e., at less than one error a day). The reason for this is that at higher error occurrence rates seemingly more latent faults tend to be discovered or "swept away", thereby resulting in a tapering effect on the plot.

To show in detail how the multiple error probability changes during the course of error injections, a plot of variation in probability of multiple errors for an error occurrence rate of 0.031 (x6 ei/hour) is shown in Figure 7. There are nine error injections

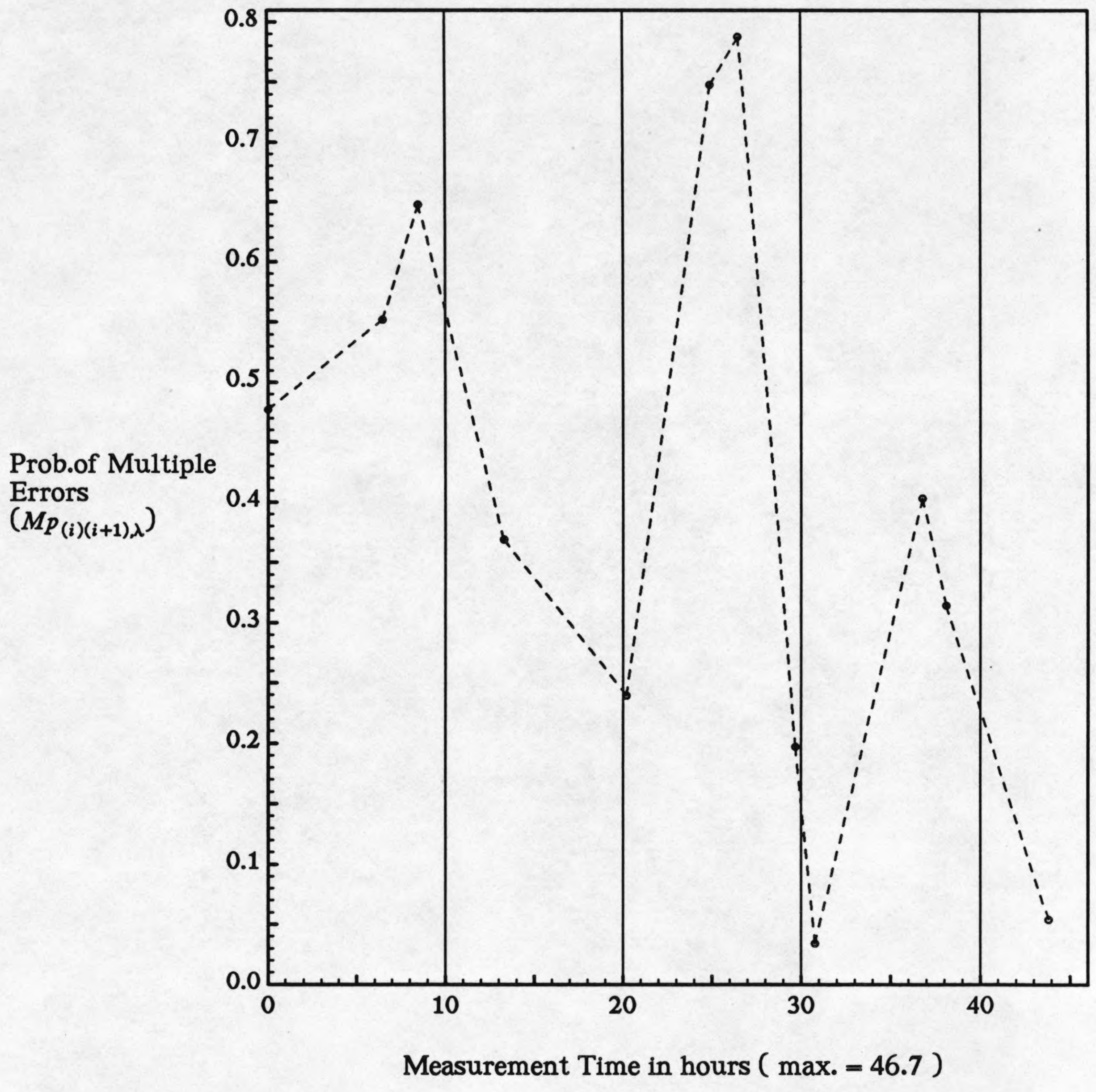

Figure 5. An Example of Multiple Error Presence During the Measurement Period

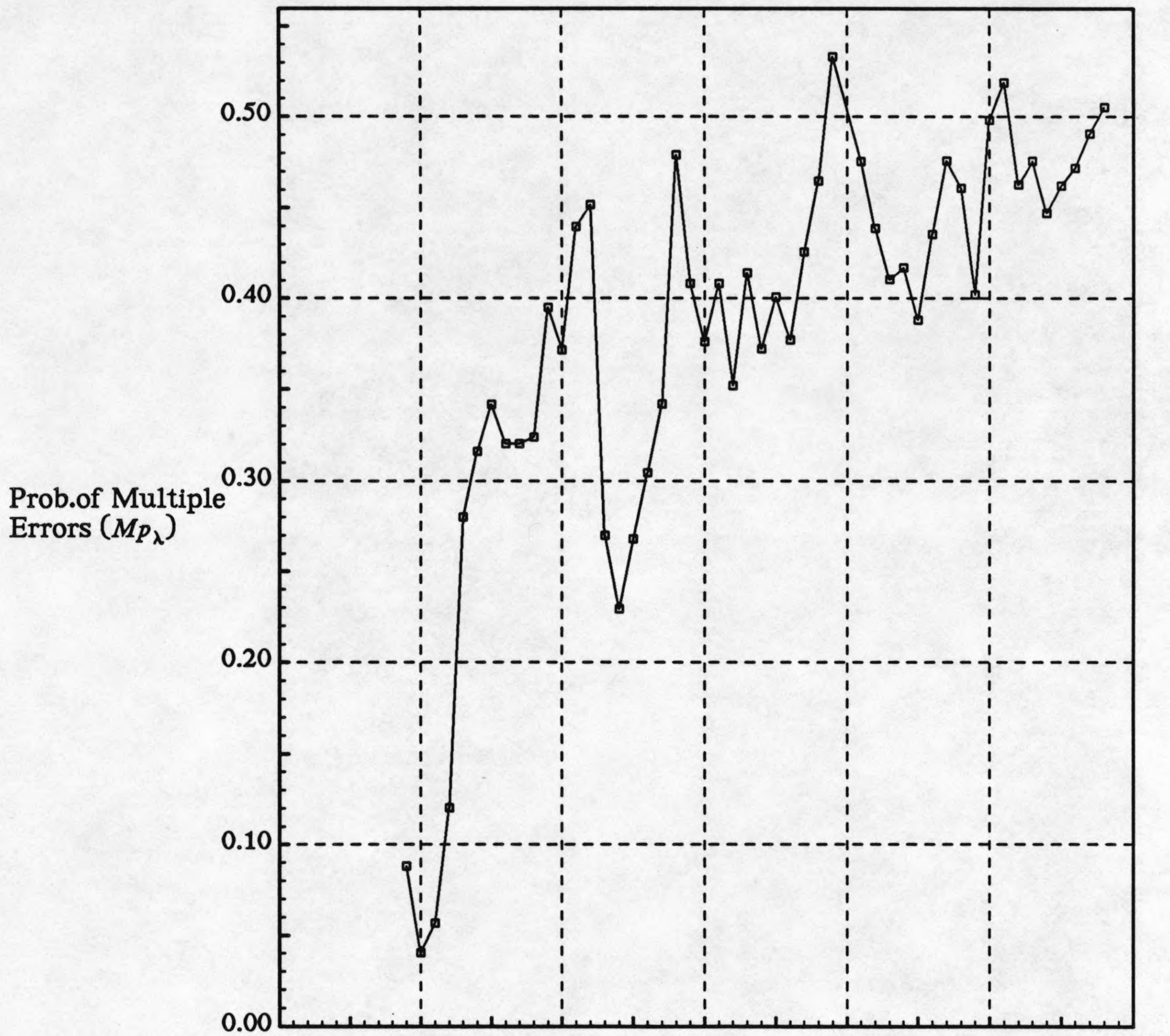

Figure 6. Multiple Error Presence at Different Error Occurrence Rates

in the measurement period for this error occurrence rate. Each dotted line represents one multiple error probability plot with respect to a specific error injection number. L_1 represents multiple error probabilities of error injection 1 (E_1) with error injections E_2, E_3, E_4 and E_5. The $Mp_{12,0.031}$, $Mp_{13,0.031}$, $Mp_{14,0.031}$ and $Mp_{15,0.031}$ values are represented on this line. Similarly L_2 represents multiple error probabilities of error injection two ($Mp_{23,0.031}$, $Mp_{24,0.031}$ and $Mp_{25,0.031}$) and so on. A downward behavior is seen for all the lines. This seems intuitive, say for L_1, the errors of E_1 will tend to be discovered as time progresses, thereby reducing the probability of multiple errors being present in the system when E_5 is introduced.

To explain the peculiarities of the error discovery behavior in the system, the high multiple error probability of 0.9 in L_2 will be explained. We find that most of the errors injected in 2 are discovered during the interval between error injections 3 and 4 (as $Mp_{23,0.031}$=0.9 suggests high values of $Me_{23,0.031}$ and $Me_{32,0.031}$). Of the few that are left some are discovered between error injections 4 and 5 ,very few though as $Mp_{24,0.031}$=0.46 (indicates that most of the contribution is from $Me_{42,0.031}$) The rest are discovered (of those discovered in the measurement period) between error injections 5 and 6. Most multiple error probability distributions over the measurement period for different error occurrence rates show this behavior.

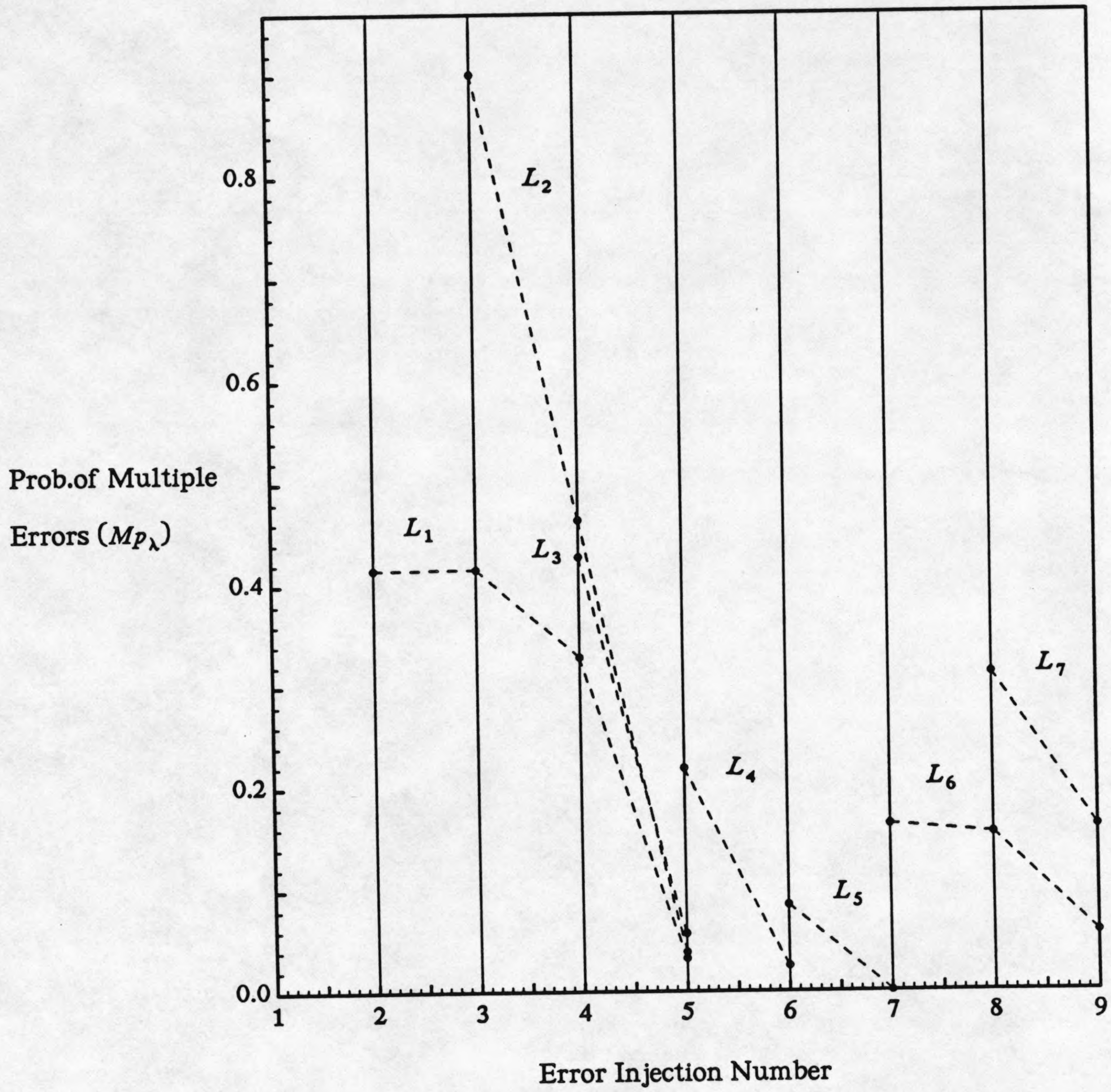

Figure 7. An Example of Multiple Error Presence with Successive Injections

3.2 Near-Coincident Fault Discovery

We will first observe how the near-coincident fault probability changes with varying time-window sizes. Figure 8 shows the variation of probability of near-coincident fault discovery with time-window sizes from 10 to 250 microseconds for three different sets of error injections. As expected, we see a monotonically increasing function of time-window size. But we find that the rate of increase in probability of near-coincident fault discovery slowly decreases for larger time-window sizes. Figure 9 shows a microscopic view (1 to 10 microseconds) of the behavioral change in the near-coincident fault probabilities. It presents itself in a stepwise fashion. This is easily understood by the fact that if we have near-coincident faults in time-window size T, then those same near-coincident faults must exist in time-window size $T+1$.

The variation of probability of near-coincident fault discovery with respect to error injection rate for three time-window sizes is shown in Figure 10. In Figure 10, the range of error occurrence rates is $0.009 \leqslant \lambda \leqslant 0.049$ (x6 error occurrences per hour), about 2 to 14 error injections over the measurement period. From Figure 10, the near-coincident fault probability values range from 0.003 to approximately 0.21 over the 10 to 250 microsecond time-window size.

There exists a high degree of similarity between Figure 6 and Figure 10. As a result, we can see there exists a high correlation between the existence of multiple errors and their discoveries in near-coincidence. From Figure 10, after the initial steep rise the plot starts to taper as in the multiple error probability case. But the saturation effect comes about slowly in Figure 10, becoming more apparent at a higher error occurrence rate than that seen in the probability of multiple error behavior. This can be explained

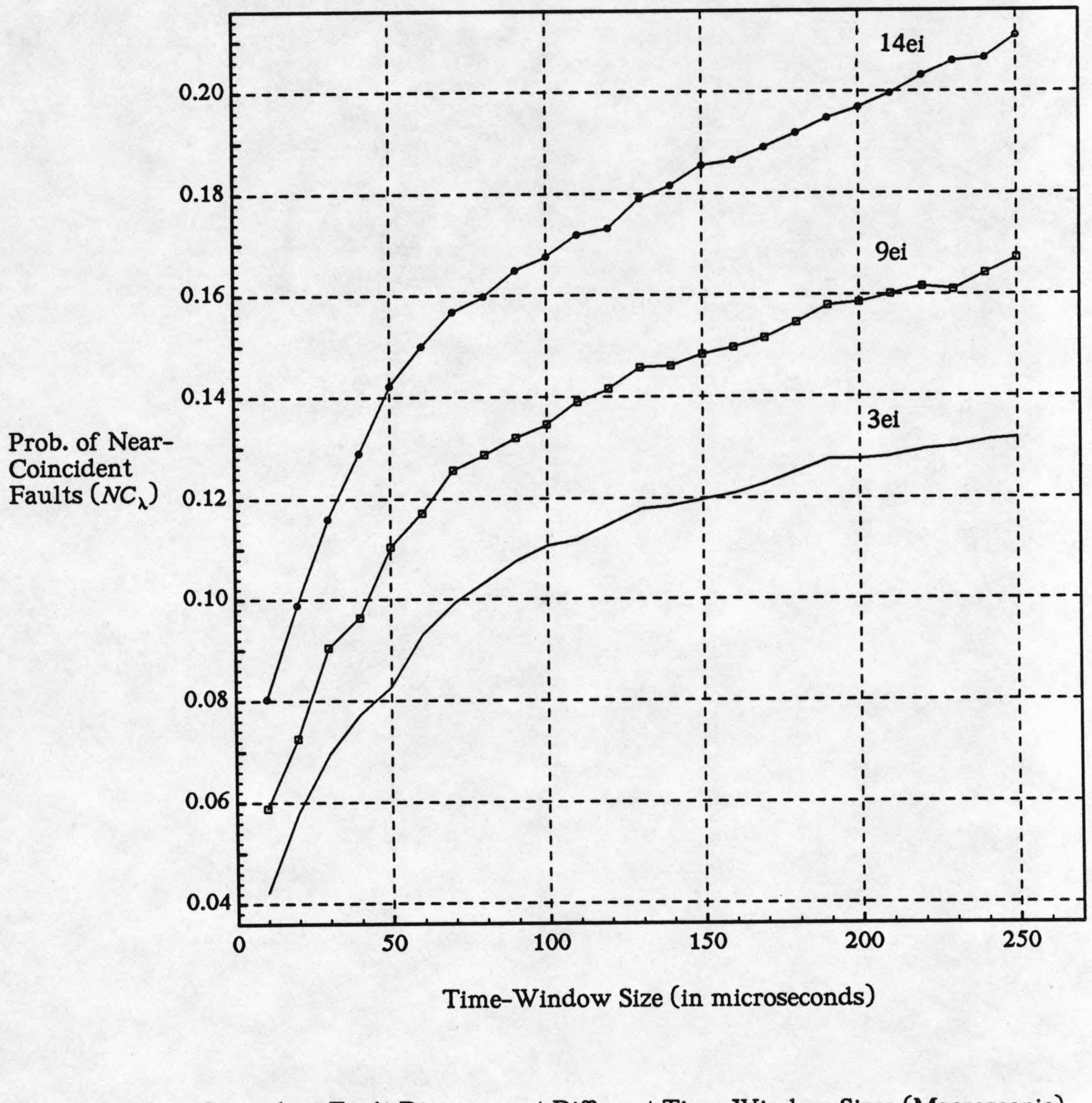

Figure 8. Near-Coincident Fault Discovery at Different Time-Window Sizes (Macroscopic)

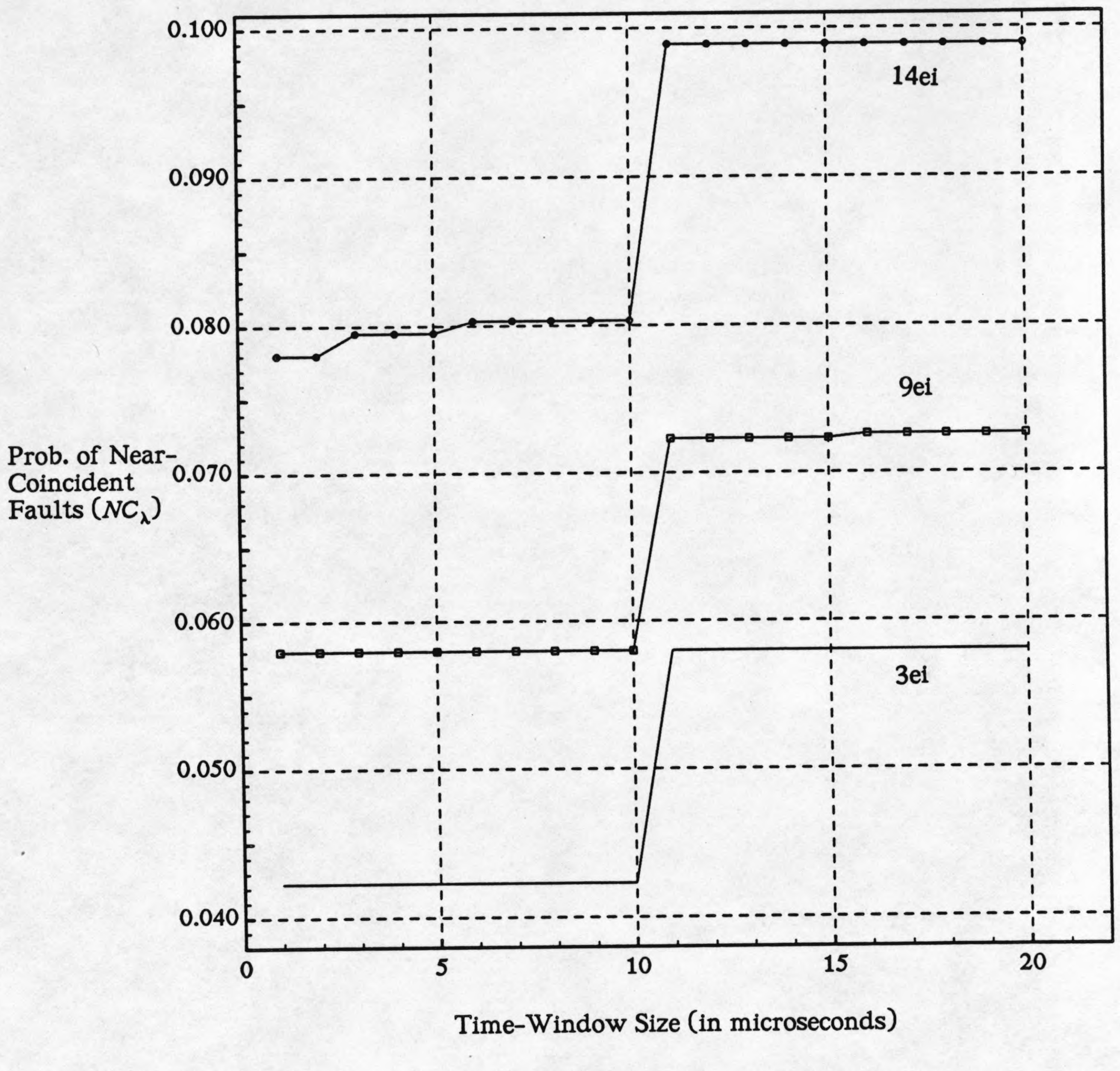

Figure 9. Near-Coincident Fault Discovery at Different Time-Window Sizes (Microscopic)

by understanding the situation in which the rate of number of latent errors being "swept out" increases; the probability of near-coincident fault discovery can increase as a side effect. But after a certain error occurrence rate, the rate of removal of latent errors from the system has more effect on the probability of near-coincident fault discovery. Thus we find that although the behavior of of multiple errors and near-coincident fault discovery has a high correlation, the saturation effect on the probability of near-coincident fault discovery occurs slower than that for multiple errors.

From Figure 10 we find that the percentage increase in probability for near-coincident faults from its lowest value to highest value is more than twice that of multiple errors (Figure 6). This shows that the probability of near-coincident fault discovery is more sensitive to change in error occurrence rate than the probability of multiple errors.

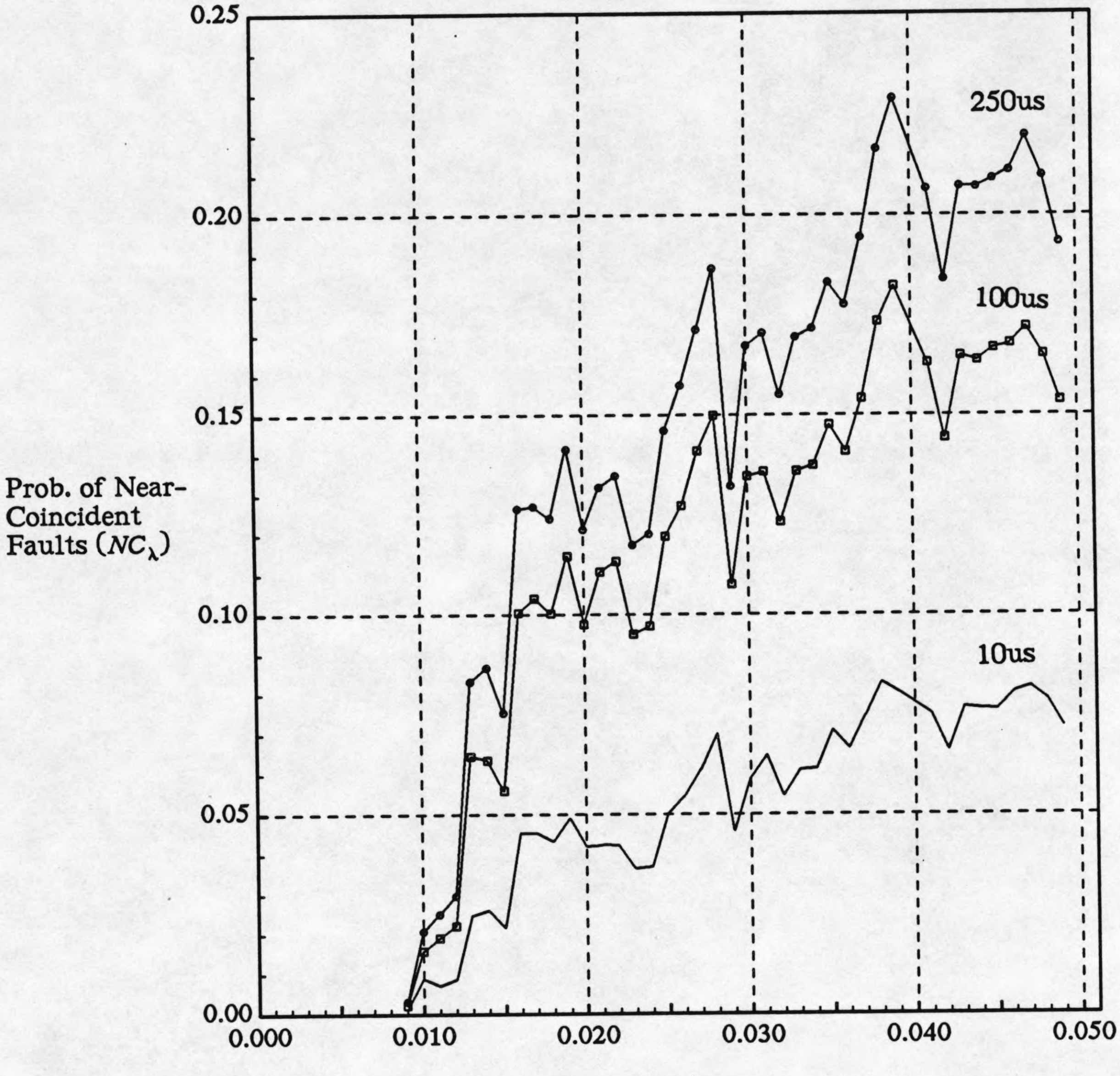

Figure 10. Near-Coincident Fault Discovery at Different Error Occurrence Rates

SECTION 4

CONCLUSIONS

Past studies have shown that multiple errors and near-coincident fault discovery can seriously degrade the reliability of a system even in a highly fault-tolerant environment such as the FTMP. Usually there exists a delay between the generation of an error due to a fault and its detection. Due to different error latencies, many "lurking" errors may be present in the system behaving as though the cause were a multiple fault. Most error recovery mechanisms cannot handle multiple faults or very close discovery of these faults. A sound understanding of multiple errors and near-coincident fault discovery can aid the development of intelligent memory scrubbing techniques to improve system reliability. This paper describes a methodology to study the behavior of multiple errors and near-coincident fault discovery in the memory subsystem of a shared memory multiprocessor.

The methodology is illustrated on a production multiprocessor system, the Alliant FX/8, in an operating system environment for the "Cedar" supercomputer under real concurrent workload conditions. A measurement period of five days was used to monitor the behavior of multiple errors and near-coincident fault discovery in a simulated error occurrence environment. Results are provided for a multiprocessing environment at realistic error occurrence rates.

The results show that for a conservative error occurrence rate of one error per day, there is a 25% chance that latent errors cause a multiple error condition. Thus one out of four errors may manifest itself as a multiple error. It was also found that 8% of the error manifestations are near-coincident in nature at the same error occurrence rate for a time window size of 50 microseconds. It was seen that the probability of multiple errors tends to saturate after a threshold error occurrence rate of approximately one error a day. The saturating effect on the probability develops as the number of latent errors being "swept" out from the system increases at higher error occurrence rates. This can lower the chances of a multiple error condition in the system and is observed after the threshold error occurrence rate. It was also found that there exists a high degree of similarity between the behaviors of multiple errors and near-coincident fault discovery, suggesting a strong correlation between existence of multiple errors and their discoveries in near-coincidence. The saturation effect on probability of near-coincident fault discovery was seen to take effect slower than that for the probability of multiple errors. The reason for this is that as the rate of latent errors being swept out increases as a result of an increase in error occurrence rate, the probability of near-coincident fault discovery increases as a side effect. The effect of time-window size on the probability of near-coincident fault discovery was also studied. It was found that the near-coincident fault probability increases monotonically with larger time-window sizes.

To further understand the behavior of multiple errors and near-coincident fault discovery, more experimental research on other systems should be attempted. In the future, the effects of transient errors on multiple error occurrence and near-coincident fault discovery will also be investigated.

REFERENCES

[1] A. L. Hopkins, T. B. Smith, and J. H. Lala, "FTMP - A highly reliable fault-tolerant multiprocessor for aircraft," *Proceedings of the IEEE*, vol. 66, no. 10, pp. 1221-1239, October 1978.

[2] J. H. Lala, "Fault detection, isolation and reconfiguration in FTMP : methods and experimental results," *Proceedings of the IEEE National Aerospace Electronics*, vol. 1, pp. 21.3.1-21.3.9, 1984.

[3] D. Kuck, D. Lawrie, A. Sameh, and D. Gajski, "CEDAR - A large scale multiprocessor," *Proceedings of the 1983 International Conference on Parallel Processing*, pp. 524-529, August 1983.

[4] J. McGough, "Effects of near-coincident faults in multiprocessor systems," *Proceedings of the IEEE/AIAA Fifth Digital Avionics Systems Conference*, pp. 16.6.1-16.6.7, 1983.

[5] K. G. Shin and Y. H. Lee, "Measurement and application of fault latency," *IEEE Transactions on Computers*, vol. C-35, no. 4, pp. 370-375, April 1986.

[6] F. L. Swern, S. J. Bavuso, A. L. Martensen, and P. S. Miner, "The effects of latent faults on highly reliable computer systems," *IEEE Transactions on Computers*, vol. C-36, no. 8, pp. 1000-1005, August 1987.

[7] J. McGough and F. L. Swern, *Measurement of Fault Latency in a Digital Avionic Miniprocessor Part II*. NASA Contractor Report 3651, 1983.

[8] R. Chillarege and R. K. Iyer, "Measurement-based analysis of error latency," *IEEE Transactions on Computers*, vol. C-36, no. 5, pp. 529-537, May 1987.

[9] R. Chillarege and R. K. Iyer, "The effect of system workload on error latency: an experimental study," *ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, pp. 69-77, 1985.

[10] *Alliant FX/Series Product Summary*. Alliant Computer Systems Corporation, Littleton, MA, October 1986.

[11] *Alliant FX/Series Architecture Manual* . Alliant Computer Systems Corporation, Littleton, MA , October 1986.

[12] M. C. Hsueh, R. K. Iyer, and D. J. Rossetti, "Measurement and modeling of computer reliability as affected by system activity," *ACM Transactions on Computer Systems*, vol. 4, no. 3, pp. 214-237, August 1986.

[13] R.K. Iyer and D.J. Rossetti, "A statistical load dependency of CPU errors at SLAC," *Proceedings of the 12th International Symposium on Fault Tolerant Computing*, pp. 363-372, 1982.

[14] *DAS 9200 System & Module A60 User's Guide*. Tektronix, Beaverton OR., 1987.

APPENDIX

THE SIMULATOR

The simulation environment consisted of three separate simulators: ELS - the Error Latency Simulator, MES - the Multiple Error Simulator and NCFS - the Near-Coincident Fault Simulator. Figure 11 shows the manner in which the simulation environment is set.

ELS uses the concurrent workload address trace (RCW) and error injector (EI - set to a particular error injection rate) as inputs to monitor the discovery behavior of faults.

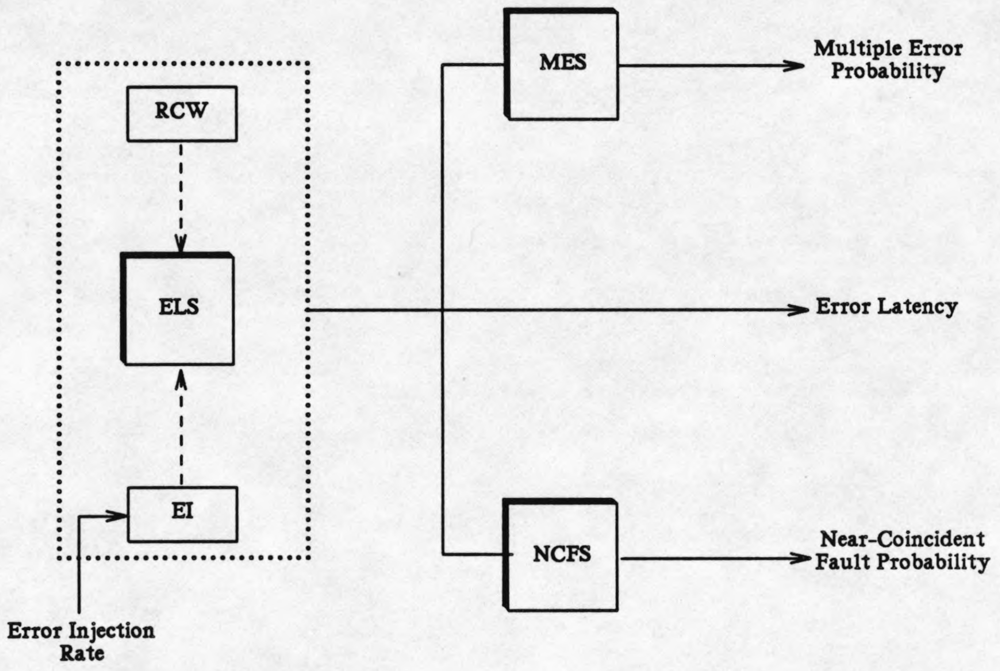

Figure 11. The Simulation Environment

The ELS is a real-time simulator whose clock is incremented by both the time stamp in the address trace and sampling times.

Table 2. Error Latency Simulator Output

Er. Latency	Address	Er.Inject Time	Er.Discover Time	Er.Inject Rate
10905974.0	102F00	1.0	10905975.0	0.030
10925851.0	1D7600	1.0	10925852.0	0.030
110409.0	01FB00	1.0	110410.0	0.030
1117435.0	04D500	1.0	1117436.0	0.030
11618402.0	0478F2	1.0	11618403.0	0.030

Table 2 shows a portion of a typical output from ELS. All times are given in time units of the real clock. We note that all the errors shown in Table 2 originated from the same error injection, but each one has a different error latency.

The multiple error simulator MES uses the output generated by ELS and examines the latency profiles to find multiple error situations. The MES calculates the multiple error probability seen at a given error injection rate.

Table 3 shows a portion of the MES output at the error injection rate = 0.30; this corresponds to 9 error injections over the complete measurement period. Note that the time for the last fault injection is not required as it does not have any multiple error probability. We see for any one source error injection , its corresponding multiple error probability with respect to other error injections decreases. From these data, MES calculates the overall multiple error probability at the given fault injection rate.

NCFS also uses the output from ELS (Table 3) and observes along real time for a specific time window size the probability of near-coincident faults. NCFS uses the discovery times of errors (from different error injections) to calculate number of near-

Table 3. Multiple Error Simulator Output

Er. Inject. Time	Source Inject no.	Dest. Inject no.	Mult. Prob.
1.0	0	1	0.431487
1.0	0	2	0.437318
1.0	0	3	0.369546
1.0	0	4	0.060724
2573348352.0	1	2	0.834617
2573348352.0	1	3	0.422660
2573348352.0	1	4	0.048530
3340261376.0	2	3	0.388702
3340261376.0	2	4	0.051163
5145478656.0	3	4	0.203210
5145478656.0	3	5	0.038282

coincident faults. Table 4 shows a portion of output from NCFS for time window sizes ranging from 1 microsecond to 10 microseconds. From these data the probability of near-coincident faults can be calculated at a given error injection rate.

Table 4. Near-Coincident Fault Simulator Output

Time_window size	No. of NCF	Total Errors Injected
1	158	2697
2	158	2697
3	158	2697
4	158	2697
5	158	2697
6	159	2697
7	159	2697
8	159	2697
9	159	2697
10	159	2697