Proceedings of the Korea Nuclear Society Spring Meeting
Cheju, Korea, May 1996

A Dependability Estimation of Microprocessor-based Software under Memory Faults using Stochastic Activity Network (SAN)

Jong Gyun Choi and Poong Hyun Seong
Korea Advanced Institute of Science and Technology

Abstract

In this work, the software behavior under memory faults in operation phase is modeled and simulated using the stochastic activity network, generalized stochastic Petri nets. This networks permit the representation of concurrency , timeliness, fault tolerance, and degradable performance of system and provide a means for determining the stochastic behavior of a complex system. We estimate the reliability of an application software in the digitized system in nuclear power plants and show the sensitivity of the software reliability to the major physical parameters which affect the software failure in normal operation phase. We found that the effects of the hardware faults on the software failure should be considered for predicting the software dependability accurately in operation phase.

1. Introduction

Computer system is the mainstay of systems, such as airline reservation system and nuclear power plant which are required to be highly reliable. This requirements of building highly reliable systems requires a high quality software, recovery mechanisms which prevent any software and hardware errors from causing the system failure and the capability for non-disruptive maintenance. Designing the reliable software and accurate predicting method of the software reliability is, therefore, one of the most important issues. Recent experiments and studies[1-3] have shown that the hardware faults can affect the software dependability and

vice versa. Previous models for dependability prediction of software has mainly concentrated on the development phase, focusing on the reliability growth of single component systems and cannot explain these effects of interactions between the software and hardware. In this work, the software behavior under memory faults in operation phase is modeled and simulated using the stochastic activity network, generalized stochastic Petri nets[4]. This networks permit the representation of concurrency , timeliness, fault tolerance, and degradable performance of system and provide a means for determining the stochastic behavior of a complex system. We estimate the reliability of an application software in the digitized system(ILS software) in nuclear power plants and show the sensitivity of the software reliability to the major physical parameters which affect the software failure in normal operation phase. We found that the effects of the hardware faults on the software failure should be considered for predicting the software dependability accurately in operation phase.

2. System Description and Assumptions

An application of the microprocessors in nuclear power plants is for the nuclear safety logic. The microprocessors are applied to the nuclear safety logic in Younggwang nuclear power plant Unit 3 and 4 in Korea. The microprocessors which are of high reliability and of wide experience in the market are used. The memory has no error correcting circuitry but is supported by the memory test routine to detect memory error periodically. When the memory fault are detected, the faulted memory chip is replaced by the new memory chip which contains no faults. The interarrival time between any two successive memory faults is assumed to be exponentially distributed with parameter λ. The fault location in memory is assumed to be random and the memory is initially fault-free. The software stored in read only memory is executed when it is required to actuate some specific components in nuclear power plants. It is assumed that the software is executed periodically and the failure of the software occurs when the code stored in the memory location having fault is executed. The interesting measure for estimating the dependability of software is mean time to failure(MTTF) and the measure for evaluating efficiency of memory check routine is normalized fault removal number. The parameter for system modeling is shown in Table 1.

Memory	λ	fault arrival rate
	M	memory size (byte)
	T_c	memory checking period
	α	memory repair rate
Software	S	software size(byte)
	T_e	software execution period

Table 1. SAN model parameters

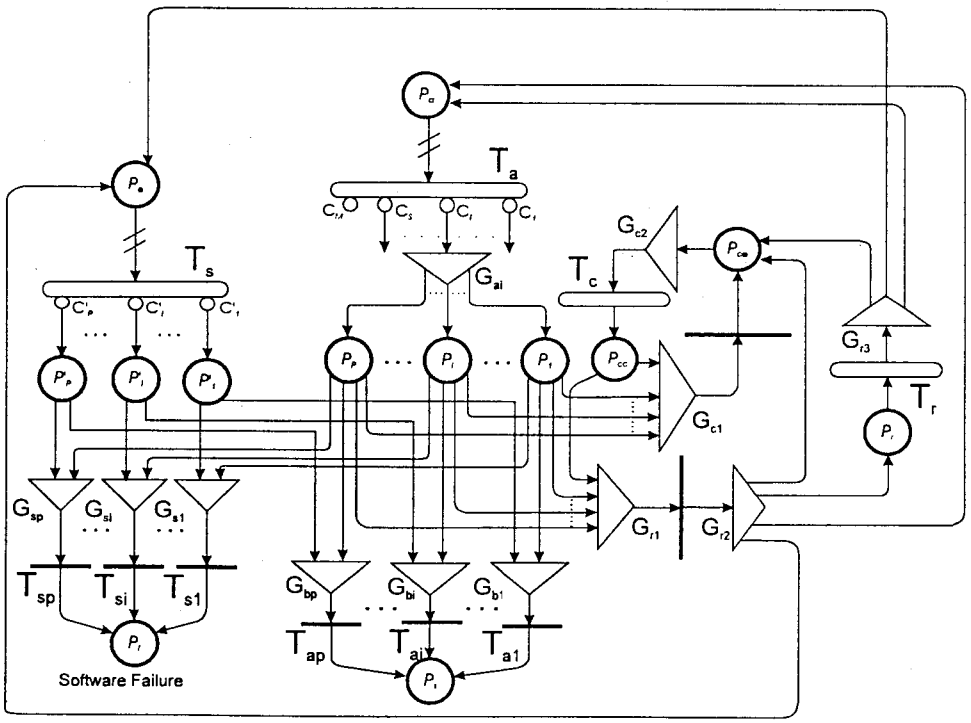


Figure 1 SAN model for representing software behavior under hardware faults

3. Stochastic Activity Network Model Representation

The fault occurrence process is modeled stochastic activity networks(SAN's). A SAN is a generalized type of stochastic Petri net composed of activities, places, and input/output gates. Activities are of two types, timed and instantaneous. Elongated ovals represent timed

activities and solid bars represent instantaneous activities. Timed activities represent activities of the modeled system whose duration impacts the system's ability to perform. Instantaneous activities represent system activities which, relative to performance variable in question, complete in a negligible amount of time. Places are depicted as circles and each place can hold a nonnegative number of tokens. The distribution of tokens in the places of the network at a given time constitutes the marking of the network at that time. Places(and associated marking) represent the state of the system being modeled. Cases may be associated with both timed and instantaneous activities and are represented by small circles. Cases permit the realization of spatial uncertainty and hence allow complex interactions that can be parameterized to be represented compactly. The two remaining net primitives are input gates and output gates. Each activity may have an associated input gate and output gate. Input gates are specified by an enabling predicate and input function(on the marking of the places). The enabling predicate must be true in order for the activity associated with that gate to be enabled. Upon completion of the associated activity, the input function is executed possibly changing the marking of the network. Output gates are characterized by a single output function (on the marking of the places), which is executed upon completion of the associated activity. The stochastic nature of SAN's is realized by associating an activity time distribution function with each of the timed activities and a probability distribution with each set of cases. Generally, both distributions can depend on the global marking of the network. In Figure 1, timed activity T_s represents the software execution. Timed activity T_a represents the arrival of the memory faults. When activity T_a is completed, the selection of ith case represents an fault occurrence in the ith row of the memory. The case distribution is assumed to be uniform. After the completion of T_a, enabling gates G_{ai}'s add 1 to the marking of places P_k's as follows :

$$MARK[P_k] = MARK[P_k] + 1$$

if ith row of memory is accessed for executing the path k of the software.

Timed activity T_c and T_r represent the memory checking and the repair process, respectively. After completion of activity T_c , activity T_r is enabled when the memory fault is detected by memory checking routine. Instantaneous activities T_{sp}'s indicate the failure of the software and activities T_{ai}'s indicate successful execution of software. When activity T_s is completed, one path of the software is selected and executed. Cases C_i's represent the selection of the

path and are assumed to be distributed uniformly. After the completion of T_s, instantaneous activities T_{si}'s and T_{ai}'s change the system state as follows :

- $MARK[P_f] := MARK[P_f] + 1$

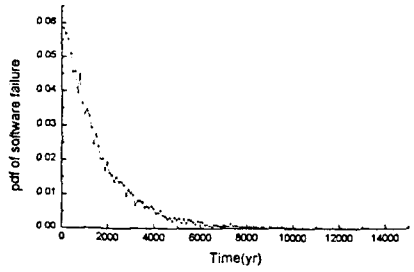
  $if\ MARK[P'_i] = 1\ and\ MARK[P_i] \geq 1,\ for\ i=1, \ldots, p.$

- $MARK[P_s] := MARK[P_s] + 1$

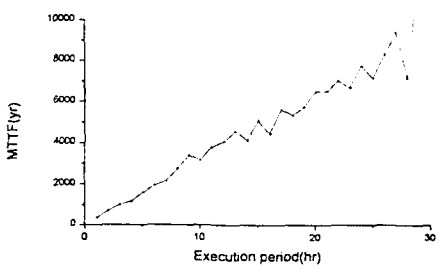
  $if\ MARK[P'_i] = 1\ and\ MARK[P_i] = 0,\ for\ i=1, \ldots, p.$

4. Results and Conclusions

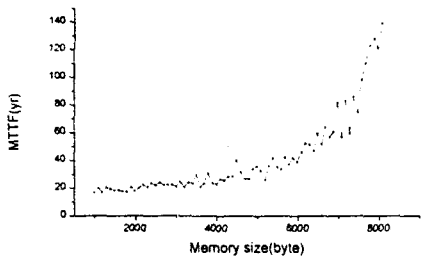
To apply the derived model to the program, we used physical parameters and assumptions as follows : $M = 8192\ bytes$, $T_e = 0.5\ hr$, $T_c = 5\ minutes$, $\lambda = 10^{-4}/hr$, $\alpha = \infty$. Figure 2-(a) shows probability distribution function of software failure obtained by SAN model and Figure 2-(b) shows the mean time to failure of software according to execution period. Figure 2-(c) shows the mean time to failure of software according to memory size. As the memory size becomes larger, the probability that the memory faults affect the software is decreased because the memory fault rate is constant, fault location is random, and the portion of the software in memory is decreased. Therefore, the increase of the memory size improves the reliability. If the time interval between the software executions increases, the number of the program execution in a given time decreases. So, the chance of the software failure is reduced and the reliability increases. In this work, the software behavior under memory faults in operation phase is modeled and simulated using the stochastic activity network, generalized stochastic Petri nets. This modeling method is particularly attractive for medium size programs such as software used in digitized systems of nuclear power plants.



(a)



(b)



(c)

Figure 2. Mean time to failure of software

5. Reference

1. R. K. Iyer and P. Velardi, " Hardware-Related Software Errors : Measurement and Analysis." *IEEE Trans. Software Eng., vol. SE-11, Feb. 1985.*
2. K. K. Goswami and R. K. Iyer, "Simulation of Software Behavior Under Hardware Faults." *Proc. on Fault-Tolerant Computing, pp. 218-227, 1993.*
3. M. Sullivan and R. Chillarege, "Software Defects and their Impact on System Availability -Study of Field Failures in Operating System", *Proc. On Fault-Tolerant Computing, pp.2 - 9, 1991*
4. J. F. Myer and A. Movgar, and W. H. Sandra, "Stochastic activity networks: Structure, behavior, and application," in *International Workshop on Timed Petri Nets, Torino, Italy, July 1-3, 1985, pp. 106-115.*