ISSN 0974 - 9373

Vol. 14 No.4 (2010) **Journal of International Academy of Physical Sciences** pp. 425 - 437

Availability Analysis of Software-Hardware System with Common Cause Shock Failure, Spare and Switching Failure*

M. Jain
Department of Mathematics, Indian Institute of Technology
Roorkee-247667 (India)
Email: madhufma@iitr.ernet.in

S. C. Agrawal and Preeti
School of Basic and Applied Sciences
Shobhit University, Meerut-250001 (India)
Email: scagrawal7@rediffmail.com; preeticl37@gmail.com

(Received February 20, 2010)

Abstrat: In this paper we investigate the performance indices of a computer system consisting of software and hardware along with a spare hardware. As soon as the hardware fails, it is replaced by a spare hardware. The failures of the system can be classified in to four categories: hardware failure, software failure, software-hardware interaction failure and common cause shock failure. The state of the system in which some parts of hardware fail but hardware still work, is called degradation state. In such condition the software does not use those failed components of hardware then the system may still work but in very slowly. The life time and the repair time of the unit are assumed to be exponentially distributed. The switching concept also considered in the proposed paper. To solve the simultaneous differential-difference equations governing the model, SOR method is employed. Various system characteristics such as reliability, availability, mean time to failure etc. are established. The numerical result also compared by using Adaptive Neuro-Fuzzy Inference System (ANFIS). A sensitivity analysis is also performed to explore the effects of various parameters on system performance.

Keywords: Software-hardware system, Switching failure, Common cause shock failure, Spare, SOR technique, Neuro-fuzzy approach.

1. Introduction

Computer systems have become part and partial of our modern lives as can be realized in our day-to-day activities. The smooth running of the computer systems depends upon the reliability of both of software and

*Presented at CONIAPS XI, University of Allahabad, Feb. 20-22, 2010.

hardware. The computer systems may fail as its components such as software or hardware or interaction between hardware and software fails. In many operating systems, the software failures are generally related with hardware therefore due to highly involvement of software and hardware it is difficult to find the difference between software and hardware failures. The reasons of failure of hardware are wear-out, electrical stress, or designing error while software is usually failed due to latent faults or because of interaction of software and hardware. There may be partial hardware failure in which the computer system works slowly, however in case of total failure of hardware, the system stops functioning.

In the field of *software and hardware* reliability; a lot of work has been done[1, 2]. Generally it is assumed that the hardware and software subsystems are independent of each other but some times there seems an interaction between hardware and software in such computer systems, Reliability modeling of hardware and software interaction, and its applications has been done by Teng et al.[3]. Levitin[4] explained reliability and performance analysis of hardware-software systems with fault-tolerant software components. Sharma and Trivedi[5] quantified software performance, reliability and security through an architecture-based approach. Using soft computing techniques the software reliability has been evaluated by Kiran and Ravi[6]. Vinod et al.[7] explained the integrating safety critical software system in probabilistic safety assessment. Becker et al.[8] evaluated performance prediction of a palladio component model for model-driven.

Whenever any unit fails, an available spare is instantly switched over for replacement of failed one. *Switching* of a spare can be done either manually or by an automatic process but sometime there is possibility that replaced spare may not work. Jain et al.[9] described machine repair system with warm standby and switching failure. Transient analysis of M/M/R machining system with mixed standby, switching failures has been studied by Jain et al.[10].

The system may fail due to individually failure of hardware and software unit or due to *common cause shock failure*. Subramanian and Anantharaman[11] did the reliability analysis of a complex standby redundant system. Jain[12] analyzed reliability of two unit system with common cause shock failures. An availability analysis for the improvement of active/standby cluster systems using software rejuvenation has been done by Park and Kim[13]. In 2003, Vaurio[14] evaluated the common cause failure probabilities in standby safety system fault tree analysis with testing-scheme and timing dependencies. Vaurio[15] described the uncertainties and quantification of common cause failure rates and probabilities for system analyses. The reliability evaluation of standby safety systems due to independent and common cause failures has been determined by Lu and Lewis[16]. Xing et al.[17] explained the reliability analysis of hierarchical computer based systems subject to common cause failures. Reliability of two non-identical units system with Common Cause Shocks failure and state dependent rates has been discussed by Jain and Mishra[18]. Shen et al.[19] explored exponential asympototic property of a parallel repairable system with common cause failure. Li et al.[20] analyzed a warm standby system with components having proportional hazard rates. A warm standby system subject to common cause failures with time varying failure and repair rates has been analyzed by El-Damcese[21].

Adaptive neuro fuzzy inference system (ANFIS) technique which is an emerging soft-computing methodology is used for the evolution of the performance characteristics in many complex systems. A queueing model for the performance prediction of flexible manufacturing systems with the help of neuro-fuzzy technique has been developed by Jain et al.[22]. Jain and Upadhyaya[23] has been studied degraded machining system by using the neuro-fuzzy systems.

In the present investigation we study a software-hardware system with spares. The concept of switching failure is also taken in to consideration. The rest paper is organized into various sections. The model description along with requisite assumptions and methods are given in section 2. The equations governing the model are constructed in section 3. In section 3, we obtain the steady state probabilities of the system by using SOR method. Section 4 contains the sensitivity analysis to explore the effect of various parameters on performance of the system. Finally the conclusion is drawn in section 5.

2. Model Description

A software-hardware system consisting of one software and one operating hardware along with a spare hardware is considered. The following assumptions are taken into consideration to formulate the mathematical model:

- The system may fail whenever both hardwares and spare hardware fail or software fails. In case of hardware-software interaction failure or due to common cause shock failure, the system may also fail.
- As soon as the hardware component fails, it is replaced by a spare component which is embedded into computer system.
- The life times of hardware, spare hardware and software are assumed to exponential distributed.

- When hardware fails, the spare hardware switches successfully with some probability p_0 whereas switching may fail with probability $\overline{p_0}$.
- The characteristic of replaced spare hardware is same as that of operating hardware.
- The system may also fail in exponential fashion due to common cause shock failure with failure rate λ_S.
- The switching of the spare hardware takes some time i.e., the spare hardware takes β_A switch over time from detected and recovered state while from detected but not recovered state it takes β_B switch over time.
- The operating hardware partially fails and go to degradation state with failure rate λ.
- The software may fail in detecting the hardware degradation with failure rate λ_C.
- The failure of operating hardware can be detected by the software with probability R_1 and may be recovered by repairing with the probability R_2.
- The failure of operating hardware may not be detected by the software with probability Q_1 and may not be recovered by repairing with the probability Q_2.
- The failure of spare hardware can be detected by the software with probability p_1 and may be recovered by repairing with the probability p_2.
- The failure of spare hardware may not be detected by the software with probability q_1 and may not be recovered by repairing with the probability q_2.

Let P_j be the steady state probability that the system being in j^{th} $(j = 0, A, B, C, S_A, S_{A'}, S_B, S_{B'}, Aa, Ab, Ac, Ba, Bb, Bc, F)$ state as shown in fig. 1 and j denotes the statue of the system as stated follows:

0	The system is in fully working state
A	Degradation state of the operating unit when the failure is detected and recovered by the software
B	Degradation state of the operating unit when the failure is detected but not recovered by the software
C	Degradation state of operating unit: failure is not detected by the software
S_A	State at which switching of the hardware spare from state A is successfully done
$S_{A'}$	State at which switching of the hardware spare from state A is failed

S_B	State at which switching of the hardware spare from state B is successfully done
$S_{B'}$	State at which switching of the hardware spare from state B is failed
Aa	Degradation state of the spare unit from state A in case when the failure is detected and recovered by software
Ab	Degradation state of the spare unit from state A in case when the failure is detected but not recovered by the software
Ac	Degradation state of the spare unit from state A in case when the failure is not detected by the software
Ba	Degradation state of the spare unit from state B in case when the failure is detected and recovered by the software
Bb	Degradation state of the spare unit from state B in case when the failure is detected but not recovered by the software
Bc	Degradation state of spare unit from state B in case when the failure is not detected by the software
F	Total system failure state

Some other notations which are used to formulate the model are given below:

$p_0(\overline{p_0})$	Probability that the spare hardware unit is switched (not switched) on.
$\alpha_1(\alpha_2)$	Failure rate of the spare hardware unit when the system is in $S_B(S_A)$ state.
$\alpha_{1b}(\alpha_{2b})$	Failure rate of the spare hardware unit when the system is in $Bb(Ab)$ state.
$\alpha_{1a}(\alpha_{2a})$	Failure rate of the spare hardware unit when the system is in $Ba(Aa)$ state.
$\alpha_{1c}(\alpha_{2c})$	Failure rate of the spare hardware unit when the system is in $Bc(Ac)$ state.
$\alpha_A(\alpha_B)$	Failure rate of the spare hardware unit when the system is in $S'A(S'B)$ state.
$\mu_A(\mu_B)$	Repair rate of the operating hardware unit from A (B) state.
$\mu_{Ba}(\mu_{Bb})$	Repair rate of the operating hardware unit from $Ba(Bb)$ state.
$\mu_{Aa}(\mu_{Ab})$	Repair rate of the operating hardware unit from $Aa(Ab)$ state.

3. The Governing Equations

The difference equations for different states of the above model are constructed as follows:

(3.1) $$0 = -(\Lambda + \lambda) P_0 + \mu_B P_B + \mu_A P_A,$$

(3.2) $$0=-(\beta_B+\mu_B)P_B+\lambda R_1Q_2P_0,$$

(3.3) $$0=-(\beta_A+\mu_A)P_A+\lambda R_1R_2P_0,$$

(3.4) $$0=-\lambda_C P_C+\lambda Q_1P_0,$$

(3.5) $$0=-\left[\alpha_1\left(p_1q_2+p_1p_2+q_1\right)\right]P_{S_B}+\beta_B p_0P_B+\mu_{Bb}P_{Bb}+\mu_{Ba}P_{Ba},$$

(3.6) $$0=-\left[\alpha_2\left(p_1q_2+p_1p_2+q_1\right)\right]P_{S_A}+\beta_A p_0P_A+\mu_{Ab}P_{Ab}+\mu_{Aa}P_{Aa},$$

(3.7) $$0=-\alpha_B P_{S'_B}+\beta_B\overline{p_o}P_B,$$

(3.8) $$0=-\alpha_A P_{S'_A}+\beta_A\overline{p_o}P_A,$$

(3.9) $$0=-(\alpha_{1b}+\mu_{Bb})P_{Bb}+\alpha_1p_1q_2P_{S_B},$$

(3.10) $$0=-(\alpha_{1a}+\mu_{Ba})P_{Ba}+\alpha_1p_1p_2P_{S_B},$$

(3.11) $$0=-(\alpha_{2b}+\mu_{Ab})P_{Ab}+\alpha_2p_1q_2P_{S_A},$$

(3.12) $$0=-(\alpha_{2a}+\mu_{Aa})P_{Aa}+\alpha_2p_1p_2P_{S_A},$$

(3.13) $$0=-\alpha_{1C}P_{Bc}+\alpha_1q_1P_{S_B},$$

(3.14) $$0=-\alpha_{2C}P_{Ac}+\alpha_2q_1P_{S_A},$$

(3.15) $$0=\alpha_{1b}P_{Bb}+\alpha_{1a}P_{Ba}+\alpha_{1c}P_{Bc}+\alpha_{1B}P_{S'_B}+\alpha_{2b}P_{Ab}+\alpha_{2a}P_{Aa}$$
$$+\alpha_{2c}P_{Ac}+\alpha_{2A}P_{S'_A}+\lambda_C P_C+\Lambda P_O.$$

Denote

$$P=[P_1,P_2,P_3]^T$$

where

$$P_1=\left[P_0,P_B,P_A,P_C,P_{S_B}\right],\quad P_2=\left[P_{S_A},P_{S'_B},P_{S'_A},P_{Bb},P_{Ba}\right]\quad\text{and}$$
$$P_3=\left[P_{Ab},P_{Aa},P_{Bc},P_{Ac},P_F\right].$$

The equations (3.1)-(3.15) can be written in the matrix form as follows:

(3.16) $$AP=P(0),$$

where $P(0)$ is zero column vector of order 15, i.e.

$$P(0)=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]^T$$

and

$$A=\begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}.$$

Here

$$\boldsymbol{A}_1=\begin{bmatrix} \Lambda+\lambda & -\mu_B & -\mu_A & 0 & 0 & 0 & 0 & 0 \\ -\lambda P_1Q_2 & \beta_B+\mu_B & 0 & 0 & 0 & 0 & 0 & 0 \\ -\lambda P_1P_2 & 0 & \beta_A+\mu_A & 0 & 0 & 0 & 0 & 0 \\ -\lambda Q_1 & 0 & 0 & \lambda_C & 0 & 0 & 0 & 0 \\ 0 & -\beta_B p_0 & 0 & 0 & \alpha_1 & 0 & 0 & 0 \\ 0 & 0 & -\beta_A p_0 & 0 & 0 & \alpha_2 & 0 & 0 \\ 0 & -\beta_B \overline{p_0} & 0 & 0 & 0 & 0 & \alpha_B & 0 \\ 0 & 0 & -\beta_A \overline{p_0} & 0 & 0 & 0 & 0 & \alpha_A \end{bmatrix}$$

$$\boldsymbol{A}_2=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\mu_{Bb} & -\mu_{Ba} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_{Ab} & -\mu_{Aa} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{A}_3=\begin{bmatrix} 0 & 0 & 0 & 0 & -\alpha_1 p_1 q_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\alpha_1 p_1 p_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\alpha_2 p_1 q_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\alpha_2 p_1 p_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\alpha_1 q_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\alpha_2 q_1 & 0 & 0 \\ -\Lambda & 0 & 0 & -\lambda_C & 0 & 0 & -\alpha_B & -\alpha_A \end{bmatrix}$$

$$A_4 = \begin{bmatrix} \alpha_{1b}+\mu_{Bb} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_{1a}+\mu_{Ba} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_{2b}+\mu_{Ab} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_{2a}+\mu_{Aa} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_{1c} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_{2c} & 0 \\ -\alpha_{1b} & -\alpha_{1a} & -\alpha_{2b} & -\alpha_{2a} & -\alpha_{1c} & -\alpha_{2c} & 0 \end{bmatrix}$$

The steady state probabilities are obtained by solving equation (3.16) numerically. For this purpose, we apply successive over relaxation (SOR) technique. Using probabilities obtained by SOR method, we can evaluate the steady state availability of the system as follows:

$$A = 1 - P_F \,. \tag{3.17}$$

4. Sensitivity Analysis

In this computational numerical results are illustrated also by using them the effect of various parameters on the system availability is examined. For this purpose Successive over relaxation (SOR) method in software MATLAB is used. The numerical results found are also compared with neuro-fuzzy results by using Adaptive Neuro Fuzzy system in fuzzy toolbox of the MATLAB package. The ANFIS networks are trained for 10 epochs for all the approximations. The membership functions of the input parameters λ and λs which are treated as linguistic variables of the fuzzy system has been described by using the Gaussian function. The number of membership functions and corresponding linguistic values of input parameters are shown in table 1. The shapes of the corresponding membership functions for figs 2(b-c)-3(b-c) wherein we have plotted availability by varying default parameters, are displayed in figs 2(a)-3(a).

Table 1: Linguistic values of the membership functions for λ, λs, α_1 and α_2

Input Variables	Number of membership functions	Figures depicting membership function	Linguistic Values
λ	4	2(a)	Low Moderate Medium High
λs	4	3(a)	Low Moderate Medium High

In order to examine the availability of the system, we set default parameters as P_1=.40, Q_1=.35, β_B=4, β_A=5, α_B=.9, α_A=.9, μ_{Ba}=2, μ_{Ab}=2.5, μ_{Aa}=2.5, p_1=.45, q_1=.35, α_{1b}=.9, α_{1a}=.9, α_{2b}=.8, α_{2a}=.7, α_{1c}=.65 and α_{2c} =.65. In all the figs, the SOR (ANFIS) results are depicted by continuous (discrete) lines. The entire graph shows that the availability of the system increases as switching probability p_0, repair rates μ_a increases.

In figs 2(b)-2(c), we have displayed the behavior for the availability of the system by varying λ and it can be seen that as λ increases availability decreases for the both p_0 and μ_a. Figs 3(b)-3(c), show the effect of λs on the availability of the system and from these graphs it is noticed that there is a decreasing trend in availability on increasing λs for the different values of p_0 and μ_a. Over all it can be conclude that:

- The availability of the system decreases as the failure rate of operating unit, common cause shock failure and the failure rate of standby unit increases.
- The results which have been obtained from ANFIS are easy and more accurate and at par with the analytical results. These observations can play a significant role for system designers for making and improving the systems.

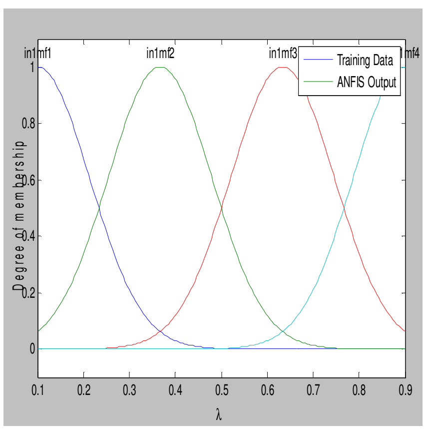

Fig. 2(a) Membership functions for input parameter λ

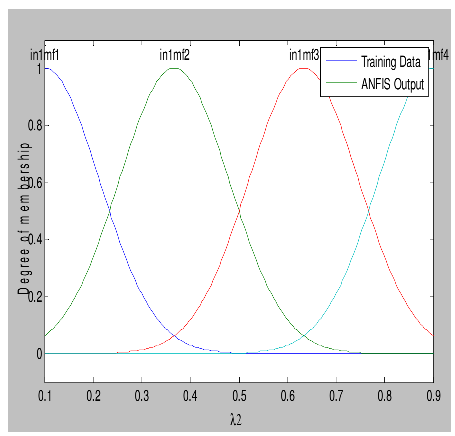

Fig. 3(a) Membership functions for input parameter λs

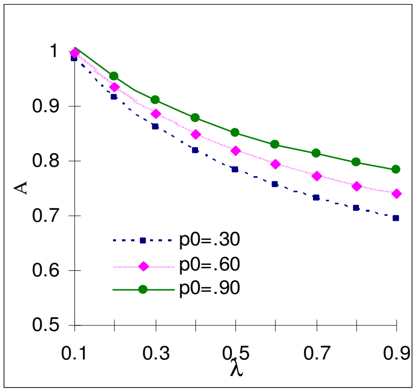

Fig. 2(b) Availability v/s λ *for different values of* p_0

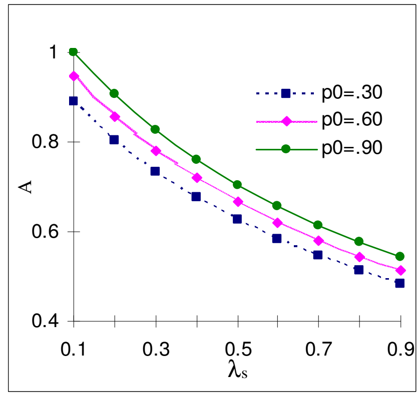

Fig. 3(b) Availability v/s λ_S *for different values of* p_0

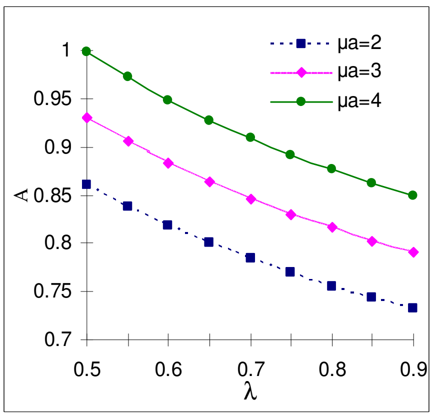

Fig. 2(c) Availability v/s λ *for different values of* μ_a

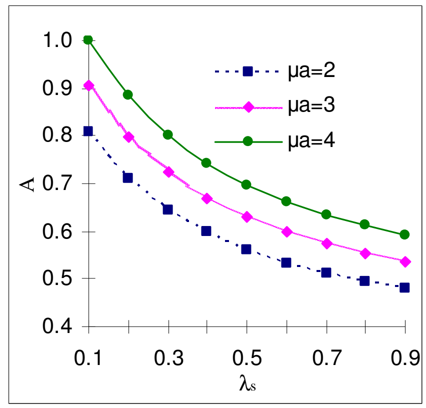

Fig. 3(c) Availability v/s λ_S *for different values of* μ_a

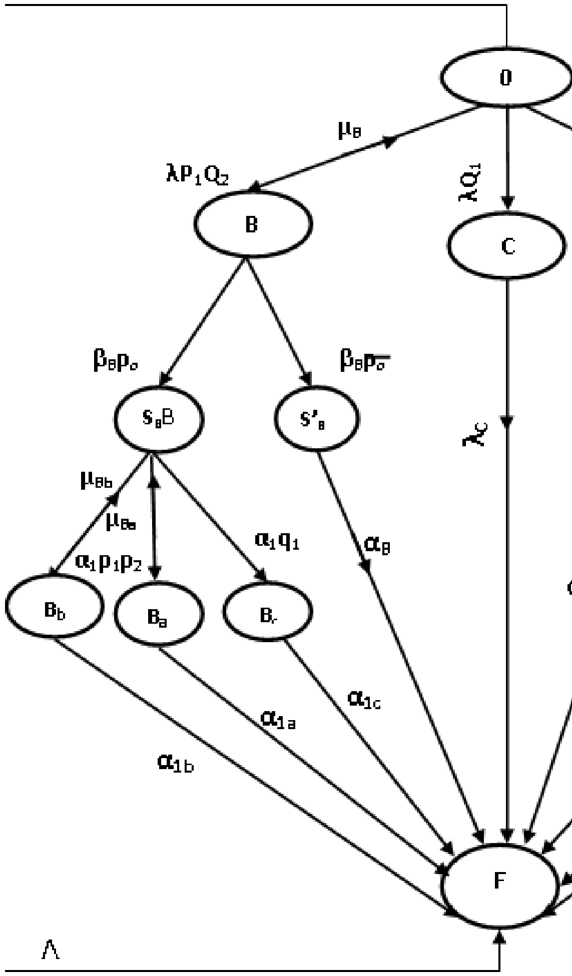

Fig 1: State transient diagram of the model

5. Conclusion

In the present investigation, we have investigated a system with software- hardware interaction, spares, common cause shock failure and switching. Performance measures such as availability have derived for the system. In the present time, many models consider only hardware failure or software failure, but we have considered interaction between the software and hardware. Therefore, our investigation will be helpful for predicting about the reliability of the system with the effect of interaction of software and hardware in spite of predicting about hardware and software separately which is usually done by several researchers. We hope that this investigation is to be used for improvement of the performance of the software and hardware systems.

References

1. S. R. Welke, B.W. Johnson and J. H. Aylor, Reliability modeling of hardware/software systems, *IEEE Trans. Relia*., **44**(3) (1995) 413-418.

2. U. Herzog and J. Rolia, Performance validation tools for software/hardware systems, *Perf, Eval*, **45** (2001) 125-146.

3. X. Teng, H. Pham and D. R. Jeske, Reliability modeling of hardware and software interaction, and its applications, *IEEE Trans. Relia*., **55(4)** (2006) 571-577.

4. G. Levitin, Reliability and performance analysis of hardware-software systems with fault-tolerant software components, *Reliab. Eng. Syst. Safety*, **91** (2006) 570-597.

5. V. S. Sharma and K. S. Trivedi, Quantifying software performance, reliability and security: An architecture-based approach, *J. Sys. Soft*., **80** (2007) 493-509.

6. N. Raj Kiran, and V. Ravi, Software reliability prediction by soft computing techniques, *J. Syst. Soft.*, **81(4)** (2008) 576-583.

7. G. V. Vinod, T. V. Santosh, R. K. Saraf and A. K. Ghosh, Integrating safety critical software system in probabilistic safety assessment, *Nuclear Eng. and Design*, **238**, 9 (2008) 2392-2399.

8. S. Becker, H. Koziolek, R. Reussner, The Palladio component model for model-driven performance prediction, *J. Sys. Soft.*, **82** (2009), 3-22.

9. M. Jain, G. C. Sharma and R. Sharma, Machine repair system with warm standby, additional repairmen, discouragement and switching failure, *Jnanabha*, **36** (2006) 157-168.

10. M. Jain, G. C. Sharma and N. Singh, Transient analysis of M/M/R machining system with mixed standby, switching failures, balking, reneging, and additional removable repairmen, *Int. J. Eng., Trans. A: Appl.*, **20(2)** (2007) 169-182.

11. R. Subramanian and V. Anantharaman, Reliability analysis of a complex standby redundant system, *Reliab. Eng. Syst. safety*. **48** (1995) 57-70.

12. M. Jain, Reliability analysis of two unit system with common cause shock failures, *Ind. J. Pure & Appl. Maths*, **29**(12) (1998) 1-8.

13. K. Park and S. Kim, Availability analysis and improvement of Active/Standby cluster systems using software rejuvenation, *J. Syst. Soft*., **61(2)** (2002) 121-128.

14. J. K. Vaurio, Common cause failure probabilities in standby safety system fault tree analysis with testing-scheme and timing dependencies, *Reliab. Eng. Syst. Safety*, **79** (2003) 43-57.

15. J. K. Vaurio, Uncertainties and quantification of common cause failure rates and probabilities for system analyses, *Reliab. Eng. Syst. Safety*, **90** (2005) 186-195.

16. L. Lu and G. Lewis, Reliability evaluation of standby safety systems due to independent and common cause failures, *IEEE Conference on Automation Sci. Eng.*, (2006) 274-279.

17. L. Xing, L. Meshkat and S. K. Donohue, Reliability analysis of hierarchical computer based systems subject to common cause failures, *Reliab. Eng. Syst. Safety*, **92(3)** (2007), 351-359.

18. M. Jain and A. Mishra, System Reliability of two non-identical units system with Common Cause Shocks failure and state dependent rates, *J. Infor. Comp. Sci.*, **10(1)** (2007) 1-12.

19. Z. Shen, X. Hu and W. Fan, Exponential asymptotic property of a parallel repairable system with warm standby under common-cause failure, *J. Math. Anal. Appl.,* **341(1)** (2008) 457-466.

20. X. Li, R. Yan and M. J. Zuo, Evaluating a warm standby system with components having proportional hazard rates, Oper. Res. Letters, **37(1)** (2009) 56-60.

21. M. A. El-Damcese, Analysis of warm standby system subject to common cause failures with time varying failure and repair rates, *Appl. Math. Sci*., **3(18)** (2009) 853-860.

22. M. Jain, S. Maheshwari and K. P. Baghel, Queueing network modelling of flexible manufacturing system using mean value analysis, *App. Math. Model*., **32(5)** (2008) 700-711.

23. M. Jain and S. Upadhyaya, Threshold N-policy for degraded machining system with multiple type spares and multiple vacations, *Qual. Tech. Quan. Manage*., **6(2)** (2009) 185-203.