Development and Performance Analysis of a Fault Tolerant Algorithm for Cluster of Workstations

Syed Misbahuddin, Dr. Eng.
Department of Computer Science and Software Engineering
Hail Community College
King Fahd University of Petroleum and Minerals
PO Box 2440, Hail, Saudi Arabia

Nizar Al-Holou, PhD
Department of Electrical and Computer Engineering
The University of Detroit Mercy
4001 West Mc Nicholas, Detroit MI 48219, USA
alholoun@udmercy.edu

Abstract

A Cluster of Workstations (COW) is network based multi-computer system, which is the most prominent distributed memory system aimed to replace supercomputers. A cluster of workstations can be viewed as a single machine in which one job is divided into n subtasks and delegated to n workstations in the COW architecture. To get the job completed, all subtasks assigned to component workstations must be completed. Therefore, for satisfactory job completion, all workstations must be functional. However, a faulty node can suspend the over all job completion task until. Therefore, a job can not be completed until a faulty node is recovered from fault. This paper presents a fault tolerant architecture for COW, which will allow a normally working workstation to perform the tasks of the faulty workstation in addition to its original assignments. The Markov models are basic tools applied for availability modeling. This paper presents a Markov Availability model for estimating the availability of component workstations as a function of workstation failure rates.

Keywords: COW, Availability modeling, Markov modeling, Distributed computing

I. Introduction

The cluster of workstations (COWs) or network of workstations (NOWs) presents attractive alternatives to expensive supercomputers and parallel computers for high performance computing. This alternative has been possible due to the emergence of powerful workstations and high speed network solutions. In terms of performance, it has been observed that COW can rival or even exceed supercomputers for some applications [1]. Example of such COW systems include: IBM SP2, DEC TruCluster, Hp, Intel/Sandia etc. According to Pfister, over 100,000 computer clusters are in use worldwide [2]. The COWs provide platforms for high speed computing. In addition to handling large computational loads, COWs can accommodate high availability and scalability features. Due to its attractive features, the idea of COW based computing is also moving towards web-based or internet based computing.

In COWs, all networked workstations formulate the illusion of a single machine. In this single system image paradigm, a computational task is broken into *n* subtasks and delegated to *n* workstations. An *ith* computational job is considered as completed when all *n* subtasks of the main job are completed. The computational power of COWs is impeded due to the faulty nature of component workstations. A fault in any of *n* workstations may obstruct the job completion until the faulty workstation is recovered from the fault. A job can not be declared as completed until all subtasks are completed by the participating workstations. The features of fault tolerance in COWs must, therefore, be injected to avoid overall task failures. With ever increasing dependency on COW based distributed systems, the number of applications requiring fault tolerance is also increasing. As a solution for achieving fault tolerance in COW some backup workstations are introduced in the system [8]. In case of a primary workstation's failure, the subtask is reassigned to the backup workstations. In this paper we present a fault tolerant scheme for

Top

COWs, in which no additional backup workstations will be required; rather normally a working workstation node can takeover the subtask assigned to a failed workstation node. This fault tolerant scheme will be managed by monitoring central control workstation (CCW). This central workstation will be responsible for monitoring the performance of all of the workstations in the system. As soon as a failure is identified in a workstation node, the subtask of the failed node will be transferred to another workstation in the system. The assigned node will carry on the assigned task in addition to its originally assigned subtask. We assume that the computational jobs will be submitted to the CCW, which will decompose the job into *n* subtasks and assign them to *n* workstations [8]. The paper is organized as follows. In section II, the features of the proposed fault tolerant cluster of workstations architecture is described, in which all components of the COW machine and central control workstation are connected to a single network backbone. The performance analysis of the proposed fault tolerant algorithm in terms of availability modeling of workstations is presented in section III. Finally, the conclusion is presented in section IV

II Architecture of Fault Tolerant Cluster of Workstations (COW)

A single bus based network provides many advantages over other kind of networks. The main advantage is the easy upgrading of the systems. That is, in a COW size can be easily increased or decreased. In this scheme the N component workstation nodes are connected to one common network. A special workstation called central control workstation (CCW) performs the task distribution assignment. The jobs are submitted to the central control workstation, which decomposes the job into *n* subtasks and assigns them to *n* workstations within the system. The assigned subtasks are performed by the workstations in parallel. After completing the subtask, each workstation reports back to the CCW. Once all the

results of the subtasks are collected from all workstations, the main job is declared as completed by the central control workstation. Obviously, a main job is considered incomplete until the results of all subtasks are not returned back from all assigned workstations. Therefore, the main job will remain suspended until the failed workstation is not recovered. Besides the subtasks assignments to the participating workstation, the central control can also monitor the performance of the workstation nodes. To ensure the availability of an *ith* subtask executed by the *ith* workstation, we propose a fault diagnostic algorithm executed by CCW.

The Fault Diagnostic Algorithm

To obtain high availability and fault tolerance in COW, we propose that the central control workstation performs a supervisory algorithm in addition to the subtask allocation role. During this action, CCW perform a check pointing algorithm in which it sends periodic diagnostic or health check messages to component workstations on a periodic basis. If a *jth* workstation does not respond within a given time window, the failed workstation's assigned subtasks are reassigned to another workstation in the system. The assigned workstation will continue its original assignment on a timesharing basis. For this purpose, we assume that each workstation is running a multi-processing operating system with sufficient computational power. With this scheme, each workstation can handle *n* subtasks (its original job assignment plus the jobs of *n-1* subtasks.). Therefore, according to the proposed fault tolerant scheme, all the subtasks assigned to n workstations can still be considered completed if at least one workstation in COW is functional. We assume all workstations in COW have equal probability of survivals. If *p* be the survival probability of any workstation and therefore are *n* workstations in the system then the probability that at least one workstation is functional at a given time will be $1-(1-p)^n$. Therefore, a job submitted to the central control workstation will be completed as long as at least one workstation in the system is in an operational

Top

state. In other words, the probability of main task completion with n workstations is also $1-(1-p)^n$.

To implement the fault diagnostic algorithm, the CCW uses a variable called Workstation index (*WINDX*), which points to an *ith* surviving workstation at a given instance of time. The CCW sends a diagnostic message periodically to all workstations in the system indicated by WINDX. If an ith workstation is not faulty, it will respond back to the CCW's diagnostic message by sending an acknowledgment message to it. However, if the CCW does not receive the expected acknowledgment message within a predefined interval of time then, it will mark that workstation as "faulty." The CCW updates the list of surviving workstations and assigns the subtask of the faulty workstation to another workstation, which may be performing some frivolous subtasks in the system. The assigned workstation will continue performing its original tasks in addition to this new assignment on a time sharing basis. Figure 1 summarizes the fault tolerant algorithm.

II. Availability modeling

The performance issue of cluster of workstation based system has drawn the interests of several researchers [2-7]. Availability is an important metric that is commonly used along with other metrics to evaluate a fault tolerant computer system. In this section, the availability of the proposed fault tolerant COW will be modeled and evaluated. Misbahuddin and Al-Holou developed availability for high performance multi-computer systems with high availability solutions [10]. COW is special multi-computer system in which all computing nodes are homogenous; therefore the same availability model presented in [10] can be extended to COW architecture with the fault tolerant scheme.

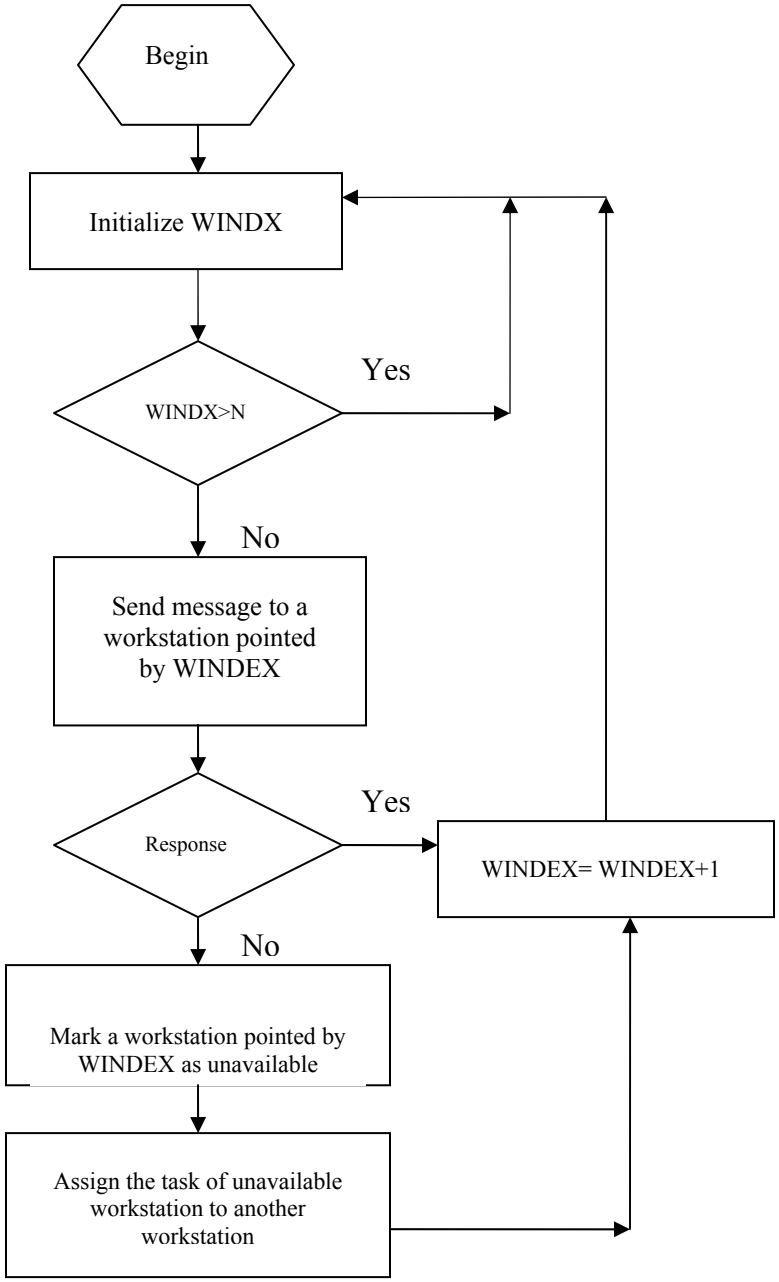

Figure 1 Fault tolerant algorithm

Top

A. Definitions and Assumptions

According to the proposed fault tolerant scheme, any workstation could be in one of two states. When the workstation is performing its original assignment, then it is called a *processing node* (*PN*). A workstation which is performing the tasks of the primary workstation is called the *secondary* processing node (*SPN*). The following assumptions are made in the availability modeling.

- All faults occurring in the system are intermittent or transient
- A processing node is subjected to random faults with a probability of failure λ_1.
- A processing node may recover from the intermittent or transient failure state with a probability λ_2.
- The transition from one state to another state occurs at a constant rate.
- When a *PN* has a transient or intermittent fault, it is said to be in a "faulty state." When the processor recovers from the fault it is said to be in a "recovered state."
- A future state is independent of all past states except the preceding one.

According to the fault diagnostic algorithm in the proposed fault tolerant scheme, each processing node can be in one of the three states defined below:

- S_0: Single operational state. The state in which the processing node is performing its own tasks only.
- S_1: Dual operational state. The state in which a primary processing node executes the tasks of another faulty processing node on a time sharing basis.
- S_2: Faulty state. The state in which the processing node is not available due to a fault

B. Markov Model

The Markov models provide convenient methods for reliability and availability estimation of computer systems [9, 11]. The basic assumption in the Markov model is based upon the notion that the probability of a next state transition of a system depends only on

its current state. In the proposed fault tolerant algorithm, whenever a workstation in the COW system is failed, the central control workstation assigns the tasks of the failed workstation to another workstation in the system. The workstation which has been assigned the tasks of the faulty workstation is said to be in the dual operational state. The workstation's switching to the dual operational state (S_1) depends only on the immediate preceding state (S_0). Therefore, the behavior of the workstation can be represented by a Markov model. Figure 3 shows the Markov model of a workstation in the proposed fault tolerant COW.

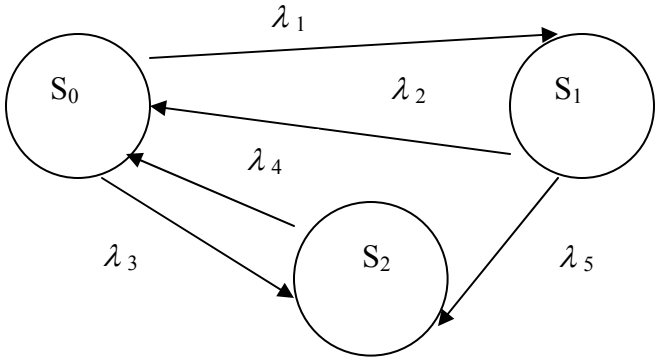

Figure 3: Markov model of a workstation with fault tolerant algorithm

The state transitions shown in Figure 3 are defined as follows:

- λ_1: Dual task rate. The rate at which a workstation is requested to switch its state
  into a dual operational mode.
- λ_2: Dual task recovery rate. The rate at which a faulty workstation recovers from
  faults and therefore, the dual mode workstation transfers from a dual mode to a
  single task mode.
- λ_3: Failure rate. The rate at which a workstation becomes faulty.

Top

- λ_4 : Repair rate. The transition rate of the workstation from a faulty state to the correct state.
- λ_5: Dual task failure rate. This rate is different from λ_3 because in this case the failure rate has occurred while the workstation was in a dual task mode.

The state transition from state S_0 to state S_1 occurs whenever the central control workstation identifies a faulty workstation and requests a workstation to take over the task assignments of the faulty workstation node. If workstation failure rate is λ_1 then we can say that transition from S_0 to S_1 is also at the rate of λ_1. The transition from state S_1 to state S_0 occurs whenever a faulty workstation is recovered from fault. If the fault recovery rate is λ_2, then the transition from S_1 to S_1 is also at the rate of λ_2. The transition from S_0 to S_2 is with rate λ_3. This transition takes place when a workstation faces a failure. Similarly, the transition from S_2 to S_0 occurs with rate λ_4. This transition occurs whenever a workstation is recovered from a transient or intermittent failure. The transition from S_2 to S_1 doses not happens because a faulty workstation cannot go to a dual task state directly.

A transition matrix **S** can be derived from the developed Markov model. The transition matrix is shown as shown below in Figure 4.

$$\mathbf{S}= \begin{array}{c|ccc} & S_0 & S_1 & S_2 \\ \hline S_0 & 1-(\lambda_1+\lambda_3) & \lambda_1 & \lambda_3 \\ S_1 & \lambda_2 & 1-(\lambda_2+\lambda_5) & \lambda_5 \\ S_2 & \lambda_4 & 0 & 1-\lambda_4 \end{array}$$

Figure 4 Transition matrix obtained from Markov model.

Top

The state probabilities of states S_0, S_1 and S_2 denoted by $P(S_0)$, $P(S_1)$ and $P(S_2)$ are the unique non-negative solutions of the following formulae:

$$P(S_j) = \sum_i P(S_i)\lambda_{ij} \quad (1)$$

$$\sum_j P(S_j) = 1 \quad (2)$$

Where λ_{ij} is the transition rate from the *ith* state to the *jth* state and $P(S_j)$ is the limiting probability of the *jth* state. The probability that a processing workstation is not faulty (State S_0) can be derived from equations (1) and (2) and matrix **S** which is:

$$P(S_0) = \frac{\lambda_4(\lambda_2 + \lambda_5)}{(\lambda_2 + \lambda_5)(\lambda_3 + \lambda_4) + \lambda_1(\lambda_5 + \lambda_4)} \quad (3)$$

Similarly, the probability of a workstation in the dual operational mode (S_1) is calculated as follows:

$$P(S_1) = \frac{\lambda_1\lambda_4}{(\lambda_2 + \lambda_5)(\lambda_4 + \lambda_1 + \lambda_3) + \lambda_1(\lambda_4 - \lambda_2)} \quad (4)$$

The availability of a workstation is the probability that a workstation is available to continue its own tasks. This means that a workstation is available if it is either in state S_0 or S_1. An expression of availability can be represented by:

$$A = P(S_0) \cup P(S_1) \quad (5)$$

A workstation working in state S_0 will switch to state S_1 only if another workstation in the system switches to state S_2. Therefore, we can say that the transition rate from S_0 to S_1 is equal to the transition rate from S_0 to S_2. Similarly, a workstation in state S_1 switches back to S_0 whenever a faulty workstation recovers from a

Top

fault. That is, the transition rate from S_1 to S_0 is equal to the transition rate from S_2 to S_0. Furthermore, λ_5, the failure rate of a workstation in S_1, will be equal to λ_3 and λ_1. This observation can be summarized in the following way:

$$\lambda_1 = \lambda_3 = \lambda_5 = \lambda \quad (6)$$

$$\lambda_2 = \lambda_4 = \mu \quad (7)$$

Using equations (6) and (7), in (3) and (4), we get:

$$P(S_0) = \frac{\mu}{2\lambda + \mu} \quad (8)$$

$$P(S_1) = \frac{\mu\lambda}{(\mu + \lambda)(\mu + 2\lambda)} \quad (9)$$

$$A = P(S_0) \cup P(S_1) = \frac{\mu}{\mu + \lambda} \quad (10)$$

Equation 10 is also the availability metric of a system with repair. This means that the developed Markov model gives us the standard availability model with a repair facility. In other words, in the proposed fault tolerant algorithm, the assignment of a faulty workstation's tasks to a normal working workstation is tantamount to the "repair" process. This result validates our developed model.

In order to assess the fault tolerance capabilities of the proposed fault tolerant algorithm, it is important to compare the developed availability results with the availability results of a similar system without fault-tolerance capability. In this case, a workstation will be either in a working state (S_0) or in a faulty state (S_2). A faulty workstation can return to a working state after a repair process. A Markov model for a system without fault-tolerant capability is shown in Figure 5.

Top

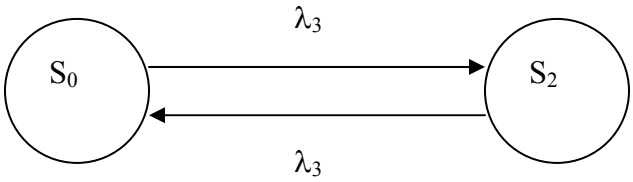

Figure 5: Markov model of a workstation without the fault tolerant algorithm

In this case, the workstation is available only when it is in the state S_0. A simple state transition matrix can be established from Figure 6 as shown below:

	S_0	S_2
S_0	$1-\lambda_3$	λ_3
S_2	λ_4	$1--\lambda_4$

Fig. 6: Transition matrix obtained from Markov model shown in Fig. 5.

The workstation's availability can be derived from the transition matrix as before and the result is:

$$P(S_0) = \frac{\lambda_4}{\lambda_4 + \lambda_3} \qquad (11)$$

If A_{woft} represents the availability of the workstation without the fault tolerant algorithm, then A_{woft} is equal to $P(S_0)$ as shown below:

$$A_{woft} = P(S_0) \qquad (12)$$

Top

C. Analysis of the Results

To evaluate the fault-tolerance capability of the COW architecture with proposed fault tolerant algorithm, a metric called the Availability Improvement Factor (*AIF*) can be used [11]. *AIF* is defined in equation (13), which indicates the degree of availability of the proposed COW architecture compared with the degree of availability of the COW architecture with no fault tolerance capability. If A_{wft} represents the availability of the workstation with fault tolerance capable COW, then an expression for AIF can written as follows:

$$AIF = \frac{A_{wft} - A_{woft}}{A_{woft}} \tag{13}$$

Using typical transition rates, A_{wft} and A_{woft} and *AIF* are calculated and are summarized Table 1.

λ_1	λ_2	λ_3	λ_4	λ_5	A_{wft}	A_{woft}	*AIF*
0.0010	0.0050	0.0010	0.0050	0.0010	0.8333	0.8333	0.0000
0.0011	0.0050	0.0011	0.0050	0.0010	0.8218	0.8197	0.2546
0.0012	0.0050	0.0012	0.0050	0.0010	0.8108	0.8065	0.5405
0.0013	0.0050	0.0013	0.0050	0.0010	0.8004	0.7937	0.8553
0.0014	0.0050	0.0014	0.0050	0.0010	0.7906	0.7812	1.1966
0.0015	0.0050	0.0015	0.0050	0.0010	0.7813	0.7692	1.5625
0.0016	0.0050	0.0016	0.0050	0.0010	0.7724	0.7576	1.9512
0.0017	0.0050	0.0017	0.0050	0.0010	0.7639	0.7463	2.3611
0.0018	0.0050	0.0018	0.0050	0.0010	0.7558	0.7353	2.7907
0.0019	0.0050	0.0019	0.0050	0.0010	0.7481	0.7246	3.2386

Table 1: A_{wft}, A_{woft} and AIF as a function of failure rate

The results shown in Table 1 indicate that as the failure rate increases, the system availability decreases. However, the availability of the system with fault tolerant capability provides higher availability than a COW system with no fault tolerance. Furthermore, it is observed that the AIF increases as the failure rate increases. This means that as the working conditions are

Top

worsened, the availability in the proposed COW system is improved. Figure 7 and 8 show the graphical relationship between availability and *AIF* as a function of failure rate, λ_1.

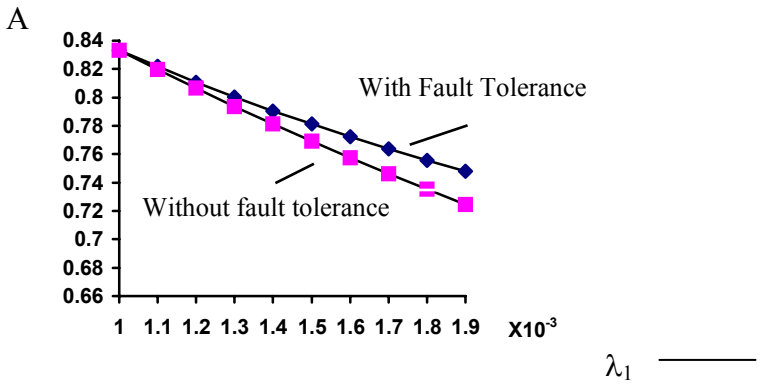

Figure 7 Availability (A) as a function of workstation failure rate

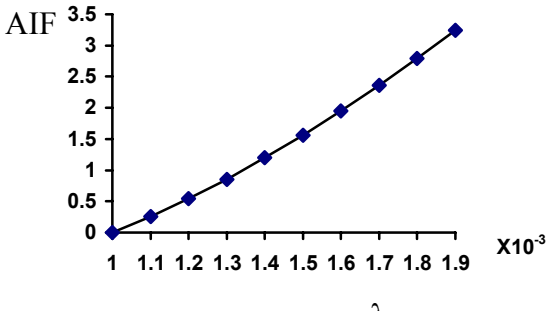

Figure 8: The Availability Improvement Factor as a function of workstation failure rate

III. Conclusion

A Cluster of workstation (COW) is an ensemble of desktop PC's, which are connected to other computers by network technologies. COWs are intended to replace costly supercomputers. COWs can give an illusion of a single computer machine to solve one common problem. However, this single machine image of COW is

Top

impaired when one or more component workstations' fail in the system. This paper presents a fault tolerant algorithm for COW. The algorithm allows a normally working workstation to replace a faulty workstation. In this way, no redundant backup workstations are needed. Markov modeling techniques have been used to evaluate the availability of the COW with and without fault tolerant shows. Numerical results show that the algorithm improves the workstation availability to the assigned tasks by assigning the tasks of the failed workstations to the normally working workstations. The assigned workstations continue their original assignments in addition to the assigned tasks of the failed workstations.

References

[1] Cristiana Amza, et al "Tread Marks: shared memory computing on networks of workstations," IEEE Computers, Feb 1996, pp. 18 – 28.

[2] G. F. Pfister, "Clusters of computers: Characteristics of an invisible architecture," keynote address presented at IEEE Int'l. Parallel processing Symp., Honolulu, April 1996.

[3]. R. P. Martin et al., "Effect of communication latency, overhead and band width in a cluster architecture, " Proc. of 24th Int'l Symp, Computer Architecture, pp. 85-96, June 1997.

[4]. Garasoulis and T. Yang, "A comparison of clustering heuristics for scheduling directed acyclic graphs in multiprocessor, " Journal of Parallel and Distributed computing, Vol. 16, pp. 276-291, 1992.

[5]. A. Mortiz and M.I. Frank, " LoGPC: Modeling network contention in message-pasing programs", IEEE Trans. Parallel abd distributed syst. Vol. 12, pp. 404-4155, 2001.

[6]. J. Kim and D.J Lilja, "Performance-based path determination for inter processor communication in distributed computing system." IEEE trans. Parallel and distributed syst. Vol. 10, pp. 372-384, 1996.

[7]. R. Davoli, L. Giachini, O. Bbaoglu, A. Amorso, and L. Alvisi, "Parallel computing in networks of workstations with

Top

parlex, “ IEEE Trans. Parallel and Distributed Syst. Vol. 7, ppp. 371-384, 1996.
[8]. Sameer Bataineha and Jamal Al-Karaki, “Fault Tolerant computing on cluster of workstations”, ACS/IEEE Int’l conf. on computer systems and applications, Tunis, Tunisia, July 14-18, 2003.
[9]. B.W Johanson, “The design and analysis of fault tolerant digital systems,” Addison Wesley, 1988.
[10]. Syed Misbahuddin, Nizar Al-Holou, “An Availability model of high performance Computing Systems*,”* Proceedings IASTED International conference on simulation and Modeling, Marina Del Ray, CA, USA, May, 2002, pp. 80-84.
[11]. Salim Hariri, “A hierarchical modeling of availability in distributed systems,” Proceedings International conference on distributed systems, May, 1991, pp. 190-197.

Top