

Software Rejuvenation on a Grid Computing Environment for Higher Availability Based on Approximate Inverse Preconditioning

V.P. KOUTRAS¹, A.N. PLATIS¹ and G.A. GRAVVANIS²

¹ *Department of Financial and Management Engineering, Business School, University of the Aegean, 31, Fostini street, GR 82100 Chios, Greece; Email: v.koutras@fme.aegean.gr, platis@aegean.gr*

² *Department of Electrical and Computer Engineering, School of Engineering, Democritus University of Thrace, 12, Vas. Sofias street, GR 67100 Xanthi, Greece; Email: ggravvan@ee.duth.gr*

Abstract— Grid computing is an innovative technology for using geographically distributed resources in order to solve large-scale problems providing heterogeneous resources. In this paper, a Grid computing environment with star topology is studied and in order to provide higher levels of Grid availability, a preventive software maintenance technique, called software rejuvenation, is proposed to be performed on certain Grid components. The aim of our work is to determine optimal rejuvenation policies that can provide higher levels of Grid asymptotic availability. Moreover, some indicators providing the level of contribution of each Grid component to system's availability are defined and computed. Continuous Time Markov Chains (CTMC) are used in order to model the behavior of the studied Grid computing environment and explicit generalized approximate inverse preconditioning methods are used for solving efficiently sparse linear systems, in order to derive the asymptotic availability

Index Terms—Grid Computing, Software Rejuvenation, Markov Chains, Approximate Inverse Matrix Algorithms, Preconditioning

I. INTRODUCTION

GRID computing is an innovative technology for complex systems with large scale resource sharing, wide area communication and multi institutional collaboration [21],[20],[5],[9],[10],[2]. In other words, Grid can be defined as a very large-scale generalised distributed network with geographically distributed machines [8]. Although the idea of what is a Grid computing environment remains the same, many different definitions can be met in the literature, leading even to surveys which state these definitions [24]. A satisfactory and quite simple definition of a grid computing environment is that it can be considered as a newly developed technology for using geographically distributed resources in order to solve large-scale problems. Due to their size and complexity, even supercomputers cannot sometimes solve them effectively. Thus Grid computing can be considered as a proper environment for solving these problems. The main advantage of the Grid is that it enables sharing, selection and aggregation of a wide variety of resources, including supercomputers, data resources, storage systems that are geographically distributed [1]. The sharing method studied in this paper consists of direct access to resources including programs databases, computers or other resources [4].

In order for the Grid to efficiently provide service, it needs to use a set of resources to complete certain tasks. Hence, there is a need of a system that can manage Grid service by matchmaking the service requests with service offers and control the assessing and the use of resources and furthermore define who has the right permissions to share, what to share and under which conditions sharing occurs [21], [4]. This role in the Grid is assigned to the Resource

Management System (RMS) [19], which can be defined as the brain of the Grid.

Despite the motivations of using Grid computing such as exploiting underutilized resource, parallel CPU capacity and other benefits, Grid does not guaranty stableness of resources due to their nature, diverse failures and error conditions that may appear on a Grid environment [26]. A lot of research effort has been paid on the direction of network reliability [22], [3] and especially in Grid dependability such as in [26], where a reliability analysis for Grid computing is defined as the probability of successful run of a given task. Dai and Levitin in [4] solve the problem of optimally allocating services on the Grid to maximize the Grid service reliability and in [21] they use a fast numerical algorithm in order to evaluate the service reliability and performance indices in a Grid with star topology.

In this paper, a Grid system with star topology is studied, which consists of an RMS, various distributed resources contained in the Root Nodes (RN) and additionally communication links between RMS and the RNs. As a dependability measure, the Grid availability is examined. In order to study the Grid asymptotic availability not only the availability of the system nodes, including RMS, has to be studied but furthermore the links' availability has to be determined.

In order to provide higher levels of Grid availability, software rejuvenation is proposed to be performed on the RMS and furthermore on the RNs. Software rejuvenation, which firstly introduced by Huang [17], is a software preventive maintenance technique that can counteract phenomena of resource exhaustion on a computer system. Hence, rejuvenation can counteract resource degradation due to resource sharing of the Grid nodes, or even exhaustion that can sometimes lead to software failures. When software rejuvenation is performed, it stops occasionally the running software, cleans its internal state and restarts it. Garbage collection, flushing operating system kernel tables, reinitializing internal data structures etc, might be involved in cleaning the internal state of the system [17].

Software rejuvenation has been studied in high available and mission critical systems. Modelling and performing software rejuvenation is based on two approaches, the measurement based and the model based approach. A model based approach is considered in [17] to describe software degradation and rejuvenation actions that are adopted to counteract it. A Continuous Time Markov Chain is used and the steady-state availability and downtime cost are derived. The idea of periodic rejuvenation into Huang's model using Markov Regenerative Stochastic Petri Net is introduced in [11]. Extending Huang's model by using semi-Markov models is considered in [6] and in [7] a semi-Markov decision process is used to determine the optimal software rejuvenation schedule. In [27] time-based rejuvenations policies are developed to improve the performability measures of a cluster system and furthermore in [18] software rejuvenation is modeled on a two node cluster system for higher availability levels.

Our aim is to determine optimal rejuvenation policies for the RMS and the RNs that can provide higher levels of the Grid asymptotic availability. Continuous Time Markov Chains are used to model the behavior of the studied Grid system along with rejuvenation and furthermore the condition of the links among the Grid nodes is modelled. Moreover, some indicators providing the level of the contribution of each node are defined and computed. These indicators take into account the software condition of the node and the corresponding link's condition.

For studying the asymptotic availability the steady-state probabilities of each state have to be computed by solving a sparse linear system. The cost-effectiveness of iterative methods over direct solution methods for such systems is now commonly accepted. Recently, generalized explicit approximate inverse preconditioning has been derived and used for the efficient solution of such sparse linear systems. Hence, in this paper, explicit generalized approximate inverse

preconditioning methods are used for solving efficiently such systems in order to derive the asymptotic availability having computed the steady-state probabilities of the model system.

II. MODEL DESCRIPTION

A. Grid System with Star Topology

The system under consideration consist in the RMS, which is considered as a central node of the Grid computing environment and the distributed RNs that offers their available resources in order to execute the jobs that the RMS sends to the RNs after a user's request. The RMS is connected with the RNs through communication links as it is shown in Figure 1, where the Grid is composed by the RMS and M distributed RNs. When a user needs to complete a certain task, a service request is sent to the RMS who is responsible for not only allocating any available resources, but furthermore to assign the task to resources that can fulfill the quality of service (QoS) requirements that the task specifies [19]. RMS has the ability to divide the received task into subtasks that can be executed in parallel, and assign them to different resources. When a resource or set of different resources complete the assigned jobs, they return the results to the RMS which reassembles them and send the completed job to the requesting user. This process can be approximated with a star topology, where its center is the RMS who is directly connected through communication links with the resources. The RMS is connected with computer systems or servers or even processors, called Root Nodes (RN), that contain the resources [4].

B. Software Rejuvenation and Model Analysis

When a software application runs continuously on a node of the Grid environment or even on the RMS, error conditions are accumulated and the system is failure prone due to resource exhaustion [17]. Resource exhaustion, if not counteracted may lead to a software failure, which is not desirable due to the cost that occurs when an application fails. Instead of taking actions that are reactive i.e. they take place after the crash failure, it is proposed to perform the software preventive maintenance technique called software rejuvenation. Rejuvenation costs, but the cost of rejuvenating an application is importantly less than the corresponding cost due a software failure because rejuvenation is a scheduled action in contrast with a failure [17]. Moreover, it is proposed to perform software rejuvenation at the RNs of the Grid too, in order to counteract phenomena of resource exhaustion on them, and consequently increase the ability of RNs to contribute to the Grid.

In order to make a complete study as far the availability of a Grid computing environment is concerned, the state of the communication links has to be also taken into account. A communication link plays an important role on the job requests and result transferring among the RMS and the RNs and hence it has to be at an operational state.

As far as the state of an application running on a node of the Grid is concerned, based on Huang's model [17], when the node starts executing, it can remain in a robust state 0 for a period of time. The robust state is a state where there are enough resources to be consumed and there is not any resource exhaustion. After this time interval, due to resource degradation, the system enters a failure probable state P, in which either rejuvenation action can take place in order to prevent a failure, and the system enters state R, or the resource exhaustion has reached a critical level and a failure cannot be avoided leading the system to state F. The system returns at the robust state after being repaired form a failure or after being rejuvenated [17].

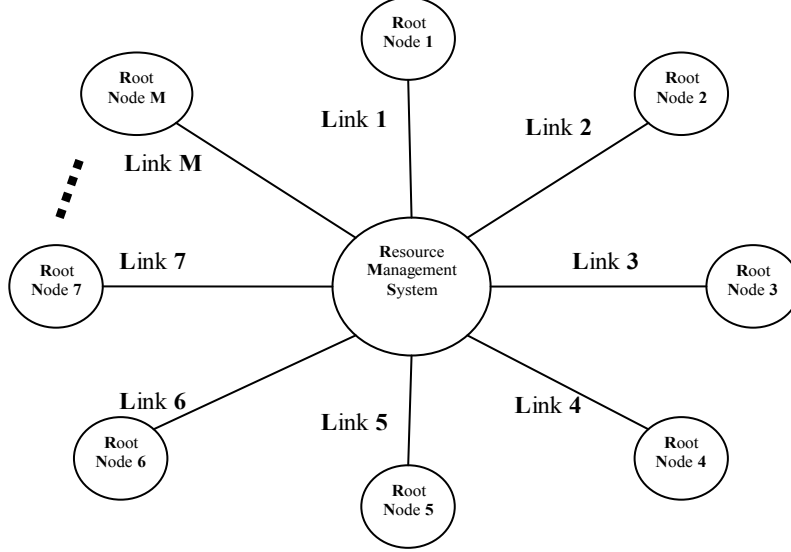


Fig. 1. Grid computing environment with the RMS and M distributed RNs

For the communication links among the Grid nodes, there are only two possible states. Either each link is operational and information can be transferred, or the link has failed and the communication between the connected nodes is not feasible. The number of the existing communication links on a Grid is equal to the number of the distributed RNs, under the assumption that the RMS is connected with each RN as shown in Figure 1.

In Figure 2, a state diagram of a node of the Grid computing environment under consideration is presented. Each state of the transition diagram comprises $M+1$ indicators. In detail, each state is signed with a label $S_j, \ell_1, \ell_2, \ell_3, \dots, \ell_M$ where $j, j \in \{0, P, R, F\}$, denotes the state of the running application on the node and $\ell_1, \ell_2, \ell_3, \dots, \ell_M$ denote the state of the M communication links (M distributed RNs participate in the Grid). Moreover, each one of the M communication links may be operational, $\ell_k = 1$, or failed, $\ell_k = 0$, where $k \in \{1, 2, 3, \dots, M\}$. It is assumed that one failure at a time can occur. For instance, one communication link can fail at a time. Furthermore, a communication link failure and a change on the systems state are assumed not to occur simultaneously. Furthermore, due to the above assumptions, the transitions caused by a change on the state of the running application, occurs between states in which all of the communication links are operational. Finally let us assume that the state structure of all the participating nodes, including RMS is identical to the one of Figure 2, i.e. each node represented by a circle in Figure 1, follows the structure of Figure 2.

The transitions between the diagram's states occur as follows: The running application experiences resource with rate α_i . Furthermore either rejuvenation is triggered with rate r_i or a failure occurs with rate λ_i . The system recovers from the rejuvenation with rate β_i and is repaired after a crash failure with rate μ_i . In the above notations $i \in \{RMS, RN_1, RN_2, RN_3, \dots, RN_M\}$. The communication link failures occur with rates $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_M$ and their repairs occur with rates $\mu_1, \mu_2, \mu_3, \dots, \mu_M$ correspondingly.

III. AVAILABILITY COMPUTATIONS

A. Steady State Analysis

The probability that the system is operational at time t and one or more failures have occur during the interval $(0, t)$ expresses the instantaneous availability of a system denoted by $A(t)$. In

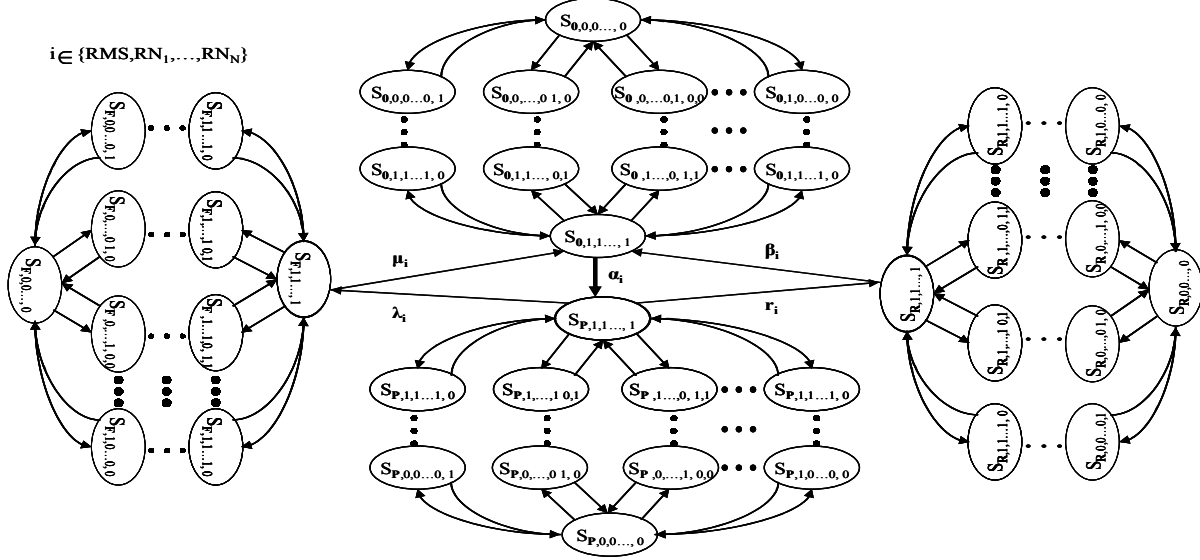


Fig. 2. State transition diagram of one node of the Grid

this paper, the study is focused on the long term behavior of the Grid and hence on the behavior of the asymptotic availability given by equation (1):

$$A_{\infty} = \lim_{t \rightarrow \infty} A(t) \quad (1)$$

or in terms of steady state probabilities π_i of state I by:

$$A_{\infty} = \sum_{i \in \text{up}} \pi_i \quad (2)$$

B. RMS and RNs Contribution to Availability

In this subsection the availability of the RMS and RNs is examined. The asymptotic availability of the RNs can be computed similarly with the RMS availability, with respect to the corresponding parameters of each RN such as the degradation, failure, repair, rejuvenation and recovery rates.

For studying the asymptotic availability of the RMS, a stochastic process $\{X(t), t \geq 0\}$ is defined, which represents the evolution of an application executed by the RMS in time. Under the assumption that the time elapsed until a transition take place is exponentially distributed, $X(t)$ is a Homogeneous Continuous Time Markov Chain (HCTMC). Let Q_{RMS} denote the generator matrix of the system, where $Q_{\text{RMS}} = [q_{ij}]$, with q_{ij} denoting the transition rate from state i to state j and N the number of states [23], [25].

The RMS is able of contributing to the Grid availability when an application running on it is at an operational state and the RMS can exchange data or information with at least one of the RNs. In contrast, RMS is not able to contribute to the Grid availability when an application executed on it, is at a down state or all the links with the RNs have failed.

Let us define the state space of the RMS as $E_{\text{RMS}} = \{S_j, \lambda_1, \dots, \lambda_M, j \in \{0, P, R, F\}, \lambda_1, \dots, \lambda_M \in \{0, 1\}\}$. Furthermore, the state space E_{RMS} is partitioned into two subsets U_{RMS} and D_{RMS} such that $U_{\text{RMS}} \cup D_{\text{RMS}} = E_{\text{RMS}}$, $U_{\text{RMS}} \neq \emptyset$, $D_{\text{RMS}} \neq \emptyset$ and $U_{\text{RMS}} \cap D_{\text{RMS}} = \emptyset$. Subset U_{RMS} is the subset of the RMS's up states and is defined as $U_{\text{RMS}} = \{S_j, \lambda_1, \dots, \lambda_M, j \in \{0, P\}, \lambda_1, \dots, \lambda_M \in \{0, 1\}\}$ and D_{RMS} is the subset of the RMS's down states and $D_{\text{RMS}} = \{S_j, \lambda_1, \dots, \lambda_M, j \in \{R, F\}, \lambda_1, \dots, \lambda_M \in \{0, 1\}\}$.

For an arbitrary RN, let us also define a stochastic process $\{Y_k(t), t \geq 0, k \in \{RN_1, RN_2, RN_3, \dots, RN_M\}\}$, which represents the evolution of an application been executed at the RN in time. $Y_k(t)$ is also an HCTMC and Q_{RN} denote the generator matrix defined similarly with Q_{RMS} .

RN's contribution to the Grid availability is the probability that an application running on it has not failed or been rejuvenated and simultaneously the communication link between this certain RN and the RMS has not failed. The state space of an RN and its partition is defined as in RMS case.

C. Indicators of Node's Contribution to Grid Availability

For the computation of the asymptotic RMS availability, the steady state distribution has to be determined by solving the linear system of equations:

$$\pi_{RMS} \cdot Q_{RMS} = \mathbf{0}, \sum_{i=1}^N \pi_{RMS,i} \quad (3)$$

where π denotes the steady state probability vector of the RMS and N denotes the total number of the model states depending on the number of the RNs. More specifically, since each of the communication links has two possible states, each application running on the RMS or on a RN, has four possible states and the total number of nodes participating into the Grid is the number of RNs plus the RMS, the total number of states for each node is given by equation (4):

$$N = 4 \cdot 2^M \quad (4)$$

Hence, we can now define an indicator expressing the Grid Contribution Incapacity (GCI) due to the RMS unavailability or to the simultaneous failures of the links. GCI is defined as the probability that an application running on the RMS have failed or being rejuvenated, or simultaneously all the communication links have failed.

$$CGI_{RMS} = \sum \text{sspPr}(\text{application on the RMS} \in \{S_R, S_F\} \cup \text{all the communication links are DOWN}) \quad (5)$$

Having computed the steady state distribution of the RMS states, equation (5) can be expressed in a mathematical form as:

$$CGI_{RMS} = \pi_{RMS,0,0000\dots0} + \pi_{RMS,P,0000\dots0} + \sum_{i \in \{\lambda_1, \lambda_2, \dots, \lambda_M\}} (\pi_{RMS,R,i} + \pi_{RMS,F,i}) \quad (6)$$

$$\lambda_1, \lambda_2, \dots, \lambda_M \in \{0,1\}$$

The corresponding indicator for the Grid Contribution Incapacity (GCI) due to an arbitrary RN unavailability or to the simultaneous failures of the corresponding link is defined as:

$$CGI_{RN} = \sum \text{sspPr}(\text{application on the RN} \in \{S_R, S_F\} \cup \text{RN's communication link is DOWN}) \quad (7)$$

From the solution of equation (2) and referring to the k^{th} RN, equation (7) is expressed mathematically as:

$$CGI_{RN_k} = \pi_{RN,0,111\dots101\dots1} + \pi_{RN,P,111\dots101\dots1} + \sum_{i \in \{\lambda_1, \lambda_2, \dots, \lambda_M\}} (\pi_{RN,R,i} + \pi_{RN,F,i}) \quad (8)$$

$$\lambda_1, \lambda_2, \dots, \lambda_M \in \{0,1\}$$

where $k \in \{RN_1, RN_2, RN_3, \dots, RN_M\}$ and the k^{th} link, connecting the referring RN to the RMS, in the terms $\pi_{RN,0,111\dots101\dots1}$ and $\pi_{RN,P,111\dots101\dots1}$ has failed.

IV. GENERALIZED APPROXIMATE INVERSE PRECONDITIONING

In order to solve efficiently the system $\pi_j Q_j = 0, j \in \{RMS, RN_1, RN_2, RN_3, \dots, RN_M\}$ with the constraint $\sum_{1 \leq j \leq n} \pi_j(i) = 1$, in order to derive the steady state distribution that is needed for the

The **Explicit Preconditioned Generalized Conjugate Gradient Square (EPGCGS)** method has been presented in [13], [14]. The computational complexity of the **EPGCGS** method is $\approx O[(2\delta l + 2\delta u + 4\lambda_1 + 4\lambda_2 + 11)n \text{ mults} + 8n \text{ adds}]v$ operations, where v denotes the number of iterations required.

According to the proposed computational strategy this class of approximate inverses includes various families of approximate inverses having in mind the desired requirements of accuracy, storage and computational work as can be seen by the following diagrammatic relationship, i.e.,

$$B^{-1} \equiv M \leftarrow \overset{\text{class I}}{\tilde{M}^{\delta l, \delta u}} \leftarrow \overset{\text{class II}}{M^{\delta l, \delta u}} \leftarrow \overset{\text{class III}}{M_1}, \quad (19)$$

where the entries of class I approximate inverse have been retained after the computation of the exact inverse, while the entries of class II approximate inverse have been computed and retained during the computational procedure of the approximate inverse. The class III diagonal inverse was computed based on the inversion of the diagonal entries of the decomposition factor, i.e. $\delta l=1$, resulting in a fast inverse algorithm, cf. [14].

The convergence analysis of the explicit approximate inverse preconditioning has been given in [15] Parallelization issues on the construction of the approximate inverse and the preconditioned conjugate gradient type methods have been addressed in [12], [16].

V. REJUVENATION POLICIES

By performing software rejuvenation on the RMS and the RNs with the appropriate rate, the aim is to minimize the value of the indicators of equations (6) and (8) and hence provide higher levels of Grid availability. But, since software rejuvenation causes a downtime for the system and hence may increase the system unavailability, first of all it has to be checked whether rejuvenation decreases the downtime and consequently unavailability or not. In the case where software rejuvenation is beneficial for a node and consequently increases its availability, it will moreover increase the total Grid computing environment availability. Additionally, system's parameters, representing the system's characteristics, also affect the decision of performing effectively rejuvenation in order to decrease system's unavailability.

The effects of rejuvenation actions on each node are examined in terms of the sign of the derivatives of the incapacity indicators. Each of these indicators is a function of the corresponding rejuvenation rate at the node, and hence its derivative with respect to this rate, can indicate if performing rejuvenation will increase or decrease node's availability.

VI. TOTAL GRID AVAILABILITY

Apart from examining how rejuvenation will benefit a node's availability and consequently the total Grid availability, an overall study of the Total Grid Availability (TGA) is also presented. In order to determine such a study, the behavior of the Grid's nodes have to be combined for the computation of the total availability.

The Grid computing environment studied can be considered as available, firstly when the RMS, is available and there is interaction between the RMS and at least one of the distributed RNs. But in order for the RMS to interact with the RNs, the communication links between them need not to have failed.

Thus, in terms of unavailability, the Grid is at a non-operational state (down) when the RMS is unavailable or the RMS is at an operational state, all the links are up and all the RNs are down,

or the RMS is up, k links have failed and the RNs corresponding to the $M-k$ available communication links are down, or finally the RMS is at an operational state and all the communication links have failed.

In terms of the stochastic processes defined in section III, the Grid is down when:

$$\begin{aligned}
& X(t) = S_{R,i} \text{ or } X(t) = S_{F,i}, \quad \text{where } i \in \{0...0...1...\} \text{ OR} \\
& X(t) = S_{j,1K1} \text{ and } (Y_k(t) = S_{R,i} \text{ or } Y_k(t) = S_{F,i}), \quad \text{where } j \in \{0, P\}, k \in \{RN_1, \dots, RN_M\}, i \in \{0...010...\} \text{ OR} \\
& X(t) = S_{j,1...101...1} \text{ (} m^{\text{th}} \text{ element is 0) and } (Y_k(t) = S_{R,i} \text{ or } Y_k(t) = S_{F,i}) \\
& \text{where } j \in \{0, P\}, k \in \{RN_1, \dots, RN_M\} - \{RM_m\}, i \in \{0...010...\}, m \in \{1, \dots, M\} \text{ OR} \\
& X(t) = S_{j,1...101K101...1} \text{ (} m^{\text{th}} \text{ and } n^{\text{th}} \text{ elements are 0) and } (Y_k(t) = S_{R,i} \text{ or } Y_k(t) = S_{F,i}) \tag{20} \\
& \text{where } j \in \{0, P\}, k \in \{RN_1, \dots, RN_M\} - \{RM_m, RM_n\}, i \in \{0...010...0\}, m, n \in \{1, \dots, M\}, m \neq n \text{ OR} \\
& \dots \\
& X(t) = S_{j,0...010...0} \text{ (} m^{\text{th}} \text{ element is 1) and } (Y_k(t) = S_{R,i} \text{ or } Y_k(t) = S_{F,i}) \\
& \text{where } j \in \{0, P\}, k \in \{RM_m\}, i \in \{0...010...0\}, m \in \{1, \dots, M\} \text{ OR} \\
& X(t) = S_{j,0...0}, \quad j \in \{0, P\}
\end{aligned}$$

where states $0, P, R, F$ have been already defined and M denotes the total number of the distributed RNs participating in the Grid. Under the hypothesis that an application is executed independently on each Grid's node and according to expression (20), the asymptotic Total Grid Unavailability (TGU) is given by the following formula:

$$\begin{aligned}
\text{TGU}(r, r_{RN_1}, \dots, r_{RN_M}) = & \sum_{y \in \{R, F\}} \sum_i (\pi_{RMS, y, i}) + \left(\sum_{j \in \{0, P\}} \pi_{RMS, j, 1...1} \right) \cdot \prod_k \left(\sum_{y \in \{R, F\}} \sum_i \pi_{RN_k, y, i} \right) + \\
& \sum_{m_1 \in \{1, \dots, M\}} \left(\sum_{j \in \{0, P\}} \pi_{RMS, j, 1...101...1} \right) \cdot \prod_{k \in \{RN_{m_1}\}} \left(\sum_{y \in \{R, F\}} \sum_i \pi_{RN_k, y, i} \right) + \\
& \sum_{m_1 \in \{1, \dots, M\}} \left(\sum_{j \in \{0, P\}} \pi_{RMS, j, 1...101...101...1} \right) \cdot \prod_{k \in \{RN_{m_1}, RN_{m_2}\}} \left(\sum_{y \in \{R, F\}} \sum_i \pi_{RN_k, y, i} \right) + \\
& \sum_{m_2 \in \{1, \dots, M\}} \\
& \dots \\
& + \sum_{j \in \{0, P\}} \pi_{RMS, j, 0...0}
\end{aligned} \tag{21}$$

Since the steady state distribution of the RMS and the distributed nodes is a function of the rejuvenation policy for each of them, expressed by $r, r_{RN_1}, r_{RN_2}, \dots, r_{RN_M}$, the Total Grid Unavailability is also a function of these rates.

By equation (21) the Total Grid Availability with respect to the Grid nodes can be derived, which is also a function of $r, r_{RN_1}, r_{RN_2}, \dots, r_{RN_M}$:

$$\text{TGA}(r, r_{RN_1}, \dots, r_{RN_M}) = 1 - \text{TGU}(r, r_{RN_1}, \dots, r_{RN_M}) \tag{22}$$

Based on equation (22) the effects of triggering rejuvenation at each one of the nodes to the Total Grid Availability can be derived, and hence the optimal rejuvenation policy, consisting in the optimal time to rejuvenate each node, that increases the total Grid computing environment availability can be distinguished. Thus, for a given Grid computing environment with a star

topology and known characteristics of the nodes participating, an optimal rejuvenation policy that provides higher levels of availability can be designed, for the nodes that are benefited by rejuvenation.

VII. CASE STUDY AND NUMERICAL RESULTS

In this section a case study is presented. The Grid computing environment is constructed under a star topology and consists of the RMS and three distributed RNs. In the case study, the data used are only for demonstration and not taken from real life systems although that are close to data met in the literature of the Grid dependability.

A. RMS and RNs Contribution to Availability

In a Grid computing environment with structure similar of Figure 1, the state transition diagram of a process describing an application executed on the RMS and the RNs has the structure of Figure 3.

The model parameters measured in hours⁻¹, are shown in Tables 1 and 2. These values correspond to three different data sets. The links' characteristics are assumed to be equal for all the data sets.

According to equations (6) and (8), the Grid Contribution Incapacity (GCI) for each of the four nodes participating into the Grid can be then computed. These quantities are functions of the corresponding rejuvenation rate values, i.e. $GCI_{RMS}(r_{RMS})$, $GCI_{RN1}(r_{RN1})$, $GCI_{RN2}(r_{RN2})$, $GCI_{RN3}(r_{RN3})$. Note that r_{RMS} represents the rejuvenation rate for the RMS, and r_{RN1} , r_{RN2} , r_{RN3} represent the rejuvenation rates for the distributed nodes RN_1 , RN_2 and RN_3 correspondingly. In order to decide whether rejuvenation is beneficial for each node, the values of these indicators have to be compared under the cases of performing or not rejuvenation, for all the data sets.

For data set 1, $dGCI_{RMS} / dr_{RMS} > 0$, which indicates that as the rejuvenation rate increase GCI_{RMS} also increases and hence performing rejuvenation does not contribute to the RMS availability and it is not beneficial. The same result is derived for the rest of the distributed nodes and hence performing software rejuvenation on a Grid computing environment characterized by the rates of data set 1 does not increase the Grid's availability. In such a case, it is proposed not to trigger any rejuvenation actions.

For data set 2, $dGCI_{RMS} / dr_{RMS} < 0$ denoting that increasing r_{RMS} , GCI_{RMS} decreases leading to higher levels of RMS availability and hence to an effective contribution of the RMS to the total Grid availability. In contrast, $dGCI_i / dr_i < 0$, $i \in \{RN_1, RN_2, RN_3\}$, indicating that software rejuvenation does not benefit any of the distributed nodes. Hence, software rejuvenation is recommended only for the RMS in order to achieve higher levels of the Grid availability.

Finally for data set 3, $dGCI_i / dr_i < 0$, $i \in \{RN_1, RN_2, RN_3\}$ and consequently as the rejuvenation rate increases the GCI_j $j \in \{RMS, RN_1, RN_2, RN_3\}$ decreases leading to higher levels of the Grid availability. Hence, rejuvenation has to be performed on all the Grid nodes in order to provide higher availability.

In Figures 4, 5 and 6 the above results for all the data sets are presented in terms of the Grid Contribution Capacity (GCC), where

$$GCC_j(r_j) = 1 - GCI_j(r_j), \quad j \in \{RMS, RN_1, RN_2, RN_3\} \quad (23)$$

TABLE I
NODES' PARAMETERS

Rate	Parameters' values in hours ⁻¹		
	Data Set 1	Data Set 2	Data Set 3
α_{RMS}	0.004	0.00001	0.00001
α_{RN1}	0.005	0.00002	0.00002
α_{RN2}	0.0075	0.00025	0.000025
α_{RN3}	0.006	0.0005	0.00003
λ_{RMS}	0.0006	0.00002	0.00002
λ_{RN1}	0.0001	0.00005	0.00003
λ_{RN2}	0.0015	0.00001	0.00004
λ_{RN3}	0.002	0.00008	0.00004
β_{RMS}	3	1	1
β_{RN1}	2.5	2	2
β_{RN2}	2	1.3	1.5
β_{RN3}	2.5	2	2.5
μ_{RMS}	2	0.25	0.25
μ_{RN1}	2	2	0.3
μ_{RN2}	1	0.4	0.25
μ_{RN3}	1.5	0.5	0.4

TABLE 2
COMMUNICATION LINKS' PARAMETERS

Rate	Parameters' values in hours ⁻¹
λ_1	0.000001
λ_2	0.000005
λ_3	0.000008
μ_1	6
μ_2	3
μ_3	2

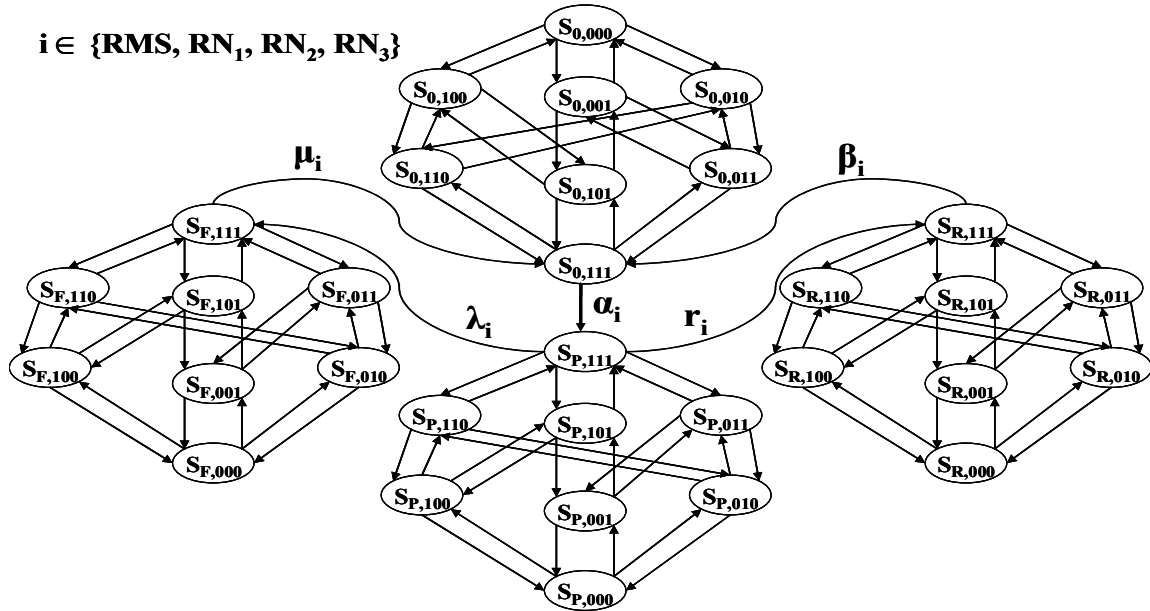


Fig. 3. Case Study: Transition state diagram of a Grid node

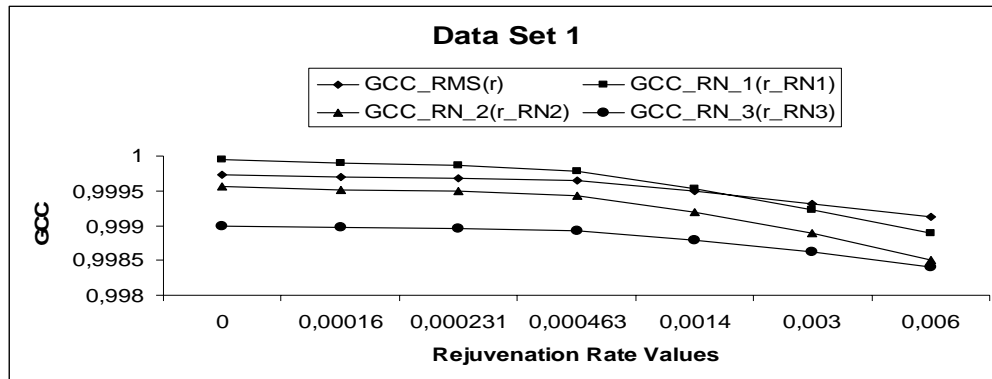


Fig. 4. Contribution to Grid Availability of all the nodes for Data Set 1

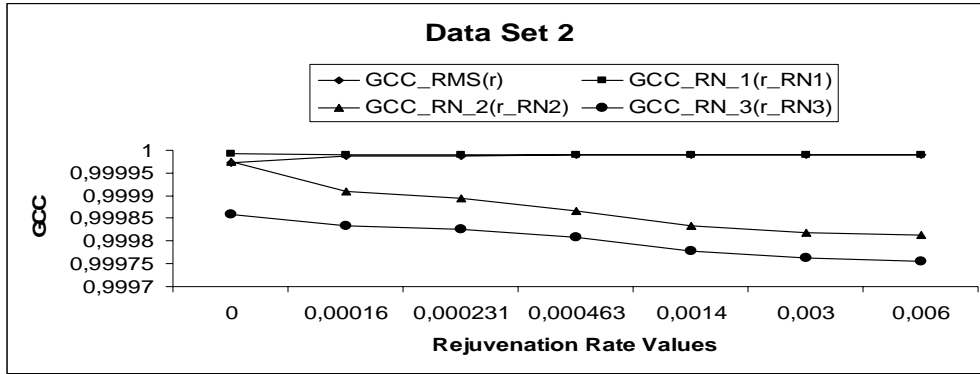


Fig. 5. Contribution to Grid Availability of all the nodes for Data Set 2

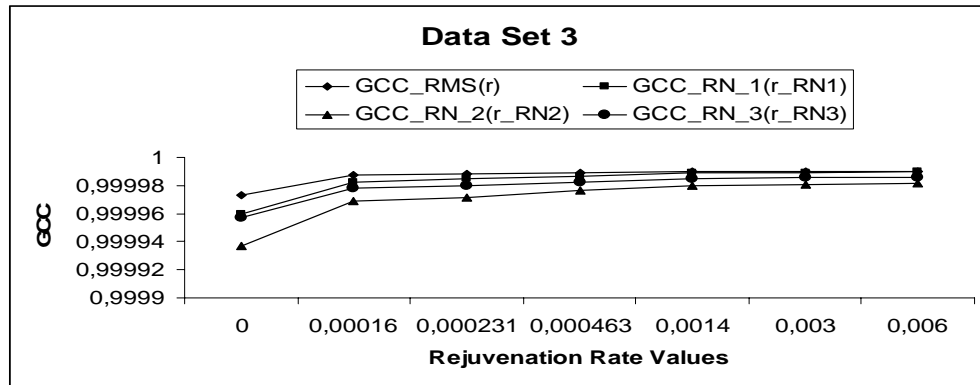


Fig. 6. Contribution to Grid Availability of all the nodes for Data Set 3

B. Total Grid Availability

The Total Grid Availability and the way that is affected by the rejuvenation policies followed at each Grid node, can be also studied.

In the case of data set 1, where there is no benefit earned by performing rejuvenation, there is no meaning of determining the optimal rejuvenation policy. In contrast, for the data set 2, where only the RMS can contribute to the Grid availability when rejuvenation actions are triggered, the behavior of the Total Grid Availability with respect to the RMS's rejuvenation rate r_{RMS} value can be studied, by setting $r_i = 0$, $i \in \{RN_1, RN_2, RN_3\}$. When no rejuvenation action is triggered on the RMS the corresponding Total Grid Availability with respect to the variable r_{RMS} ($TGA(r_{RMS}, 0, 0, 0)$), is $TGA(0, 0, 0, 0) = 0,999973334$ according to equation (21). As the rejuvenation rate r_{RMS} increases, the $TGA(r_{RMS}, 0, 0, 0)$ also increases, indicating that rejuvenation has to be triggered on the RMS immediately when the application running enters the failure probable state P. The behavior of $TGA(r_{RMS}, 0, 0, 0)$ is presented in Figure 7.

For data set 3, where all of the Grid nodes contribute to the Grid availability, when rejuvenation is performed on the applications executed on them, the Total Grid Availability $TGA(r_{RMS}, r_{RN_1}, r_{RN_2}, r_{RN_3})$ is increased as the rejuvenation rate values increase. Thus the rejuvenation policy proposed for the applications running on the Grid nodes characterized by data set 3, consists of triggering rejuvenation on the RMS and the RNs immediately when the applications enter the failure probable state in order to avoid failures and hence achieve higher levels of the Grid availability.

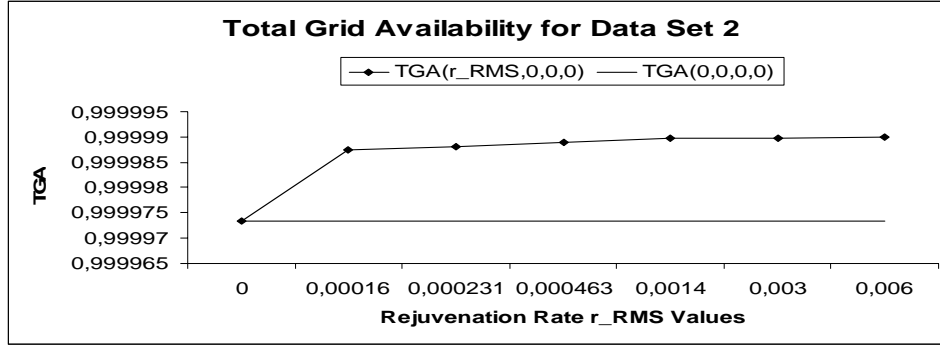


Fig. 7. Total Grid Availability with Respect to the RMS's Rejuvenation Rate r_{RMS} Values for Data Set 2

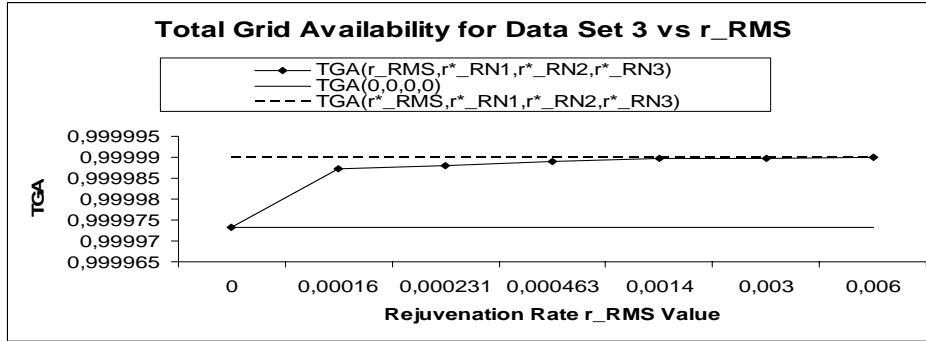


Fig. 8. Total Grid Availability with Respect to the RMS's Rejuvenation Rate r_{RMS} Values for Data Set 3

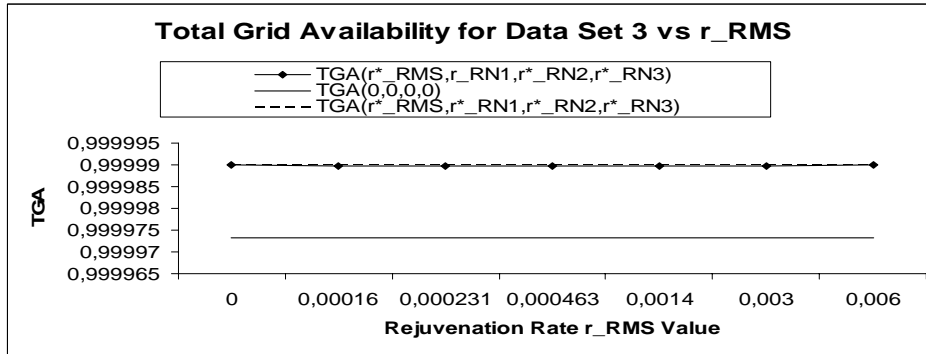


Fig. 9. Total Grid Availability with Respect to the RN₁'s Rejuvenation Rate r_{RN1} Values for Data Set 3

In Figure 8, the TGA for data set 3 is presented, with respect to the rejuvenation rate r_{RMS} . The solid line represents the TGA when no rejuvenation action is taken at any node and the dashed line represent TGA corresponding to the values $r^*_{RMS}, r^*_{RN1}, r^*_{RN2}, r^*_{RN3}$ indicating triggering rejuvenation at the time that the application executed at each node enters the failure probable state. That is $r^*_{RMS} = r^*_{RN1} = r^*_{RN2} = r^*_{RN3} = 0.006$, which corresponds to performing rejuvenation once a week. In fact, the above rate values are used to illustrate our statement and are not restrictive as rejuvenating immediately after entering the failure probable state means that $r_i = \infty, i \in \{RMS, RN_1, RN_2, RN_3\}$. Figure 9 presents the TGA behavior with respect to r_{RN1} . The effect of r_{RN2}, r_{RN3} on the TGA is similar with the one presented in Figure 9 for r_{RN1} .

By Figures 8 and 9 the great effect of the RMS on the TGA can be distinguished, as the increase of r_{RMS} causes an important increase on the Total Grid Availability in contrast with r_{RN1}

whose increase has meaningless effect on the Total Grid Availability.

C. Generalized Approximate Inverse Preconditioning

In the case of studying a Grid structure with many RNs, the complexity of the study increases considerably, since the number of states for each application running on a certain node is increased.

The sparse linear system was solved by the **EPGCGS** and **EPBICG-STAB** method. The Explicit Preconditioned Conjugate Gradient - type methods were terminated when $\|r_i\|_2 < 10^{-6}$. In Table 4, the convergence behavior of the **EPGCGS** and **EPBICG-STAB** method is presented for various values of the “retention” parameters δl and $\delta u = \delta l - 1$. The performance (in s.hh) of the **EPGCGS** and **EPBICG-STAB** method is given in Figure 5.

Further, it should be noted that approximate inverse preconditioning has been implemented for solving efficiently sparse linear systems on multiprocessor system, using Java multithreading and OpenMP, and on multicomputer systems, using MPI communication library, cf. [12], [16].

TABLE 4
CONVERGENCE BEHAVIOR OF THE EPGCGS AND EPBICG-STAB

n	t_1	t_2	EXPLICIT PRECONDITIONED CG-TYPE METHOD								
			EPGCGS				EPGCGS-STAB				
			$\delta l=1$	$\delta l=t_1$	$\delta l=2t_1$	$\delta l=3t_1$	$\delta l=1$	$\delta l=t_1$	$\delta l=2t_1$	$\delta l=3t_1$	
32	4	9	3	2	2	2	3	2	2	2	2
64	7	17	3	2	2	2	3	2	2	2	2
128	14	33	3	2	2	2	3	2	2	2	2
256	25	65	3	2	2	2	3	2	2	2	2
512	50	129	3	2	2	2	3	2	2	2	2
1024	91	257	3	2	2	2	3	2	2	2	2

TABLE 5
THE PERFORMANCE OF THE EPGCGS AND EPBICG-STAB METHODS

n	t_1	t_2	TIMING (given in ss.hh)									
			EXPLICIT PRECONDITIONED CG-TYPE METHOD									
			EPGCGS				EPGCGS-STAB					
$\delta l=1$	$\delta l=t_1$	$\delta l=2t_1$	$\delta l=3t_1$	$\delta l=1$	$\delta l=t_1$	$\delta l=2t_1$	$\delta l=3t_1$					
256	25	65	0.03	0.05	0.07	0.12	0.03	0.05	0.08	0.12		
512	50	129	0.22	0.36	0.60	1.05	0.23	0.37	0.62	1.06		
1024	91	257	0.96	3.22	5.81	10.32	0.99	3.25	5.83	10.42		

VIII. CONCLUSIONS

In this paper, the new and evolutionary technology of Grid computing was taken under examination in terms of availability. A Grid computing environment with star topology consisting in the Resource Management System, distributed Root Nodes and communication links between these nodes was considered. Software rejuvenation performance on each of the Grid’s nodes was modeled in order to increase the Grid’s availability, when possible. Indicators about the contribution of each node to the Grid availability were derived and their behavior with respect to software rejuvenation was also studied. In the case that a node’s availability was increased when software rejuvenation is performed, the rejuvenation policy that provides higher levels of the Grid availability was proposed. Moreover, explicit approximate inverse preconditioning schemes were used, in order to solve sparse linear system of equations obtained when many Roto Nodes participate into the Grid. The effectiveness and applicability of explicit approximate inverse preconditioning of solving sparse linear system was illustrated.

REFERENCES

- [1] M. Baker, R. Buyya and D. Laforenza, "Grids and Grid technologies for wide-area distributed computing," *Softw. Pract. Exper*, 32(15), 2002, pp. 1437-1466.
- [2] F. Berman, R. Wolski, H. Casanova, W. Cirne, H. Dail and M. Faerman, "Adaptive computing on the Grid using AppLeS," *IEEE Trans Parallel Distribut Syst*, 14, 2003, pp. 369-382.
- [3] J. Carlier and C. Lucet, "A Decomposition Algorithm for Network Reliability Evaluation," *Discrete Applied Mathematics*, 65, 1996, pp. 141-156.
- [4] Y. S. Dai and G. Levitin, "Optimal service task partition and distribution in grid system with star topology," *Reliability Engineering & System Safety*, 9, 2006, pp. 1071-1082.
- [5] S.K. Das, D.J. Harvey and R. Biswas, "Parallel processing of adaptive meshes with load balancing," *IEEE Trans Parallel Distribut Syst.*, 12, 2001, pp. 1269-1280.
- [6] T. Dohi, K. G. Popstojanova and K. S. Trivedi, "Estimating software rejuvenation schedules in high assurance systems," *Computer Journal*, 44(6), 2001, pp. 473-485.
- [7] H. Eto and T. Dohi, "Determining the Optimal Software Rejuvenation Schedule via Semi-Markov Decision Process," *Journal of Computer Science*, 2, 2006 no. 6, pp. 528-534.
- [8] I. Foster and C. Kesselman (eds.), "The Grid: Blueprint for a New Computing Infrastructure". Morgan Kaufmann: San Fransisco, CA, 1999.
- [9] I. Foster, C. Kesselman and S. Tuecke, "The anatomy of the grid: enabling scalable virtual organizations," *Int J High Perform Comput Appl*, 15, 2001, pp. 200-222.
- [10] I. Foster, C. Kesselman, J.M. Nick and S. Tuecke, "Grid services for distributed system integration," *Computer*, 35(6), 2002, pp. 37-46.
- [11] S. Garg, A. Puliafito, M. Telek and K. S. Trivedi, "Analysis of software rejuvenation using Markov Regenerative Stochastic Petri Net," *Proceedings. Of 6th Intl. Symposium on Software Reliability Engineering*, 1995, pp. 180-187.
- [12] K.M. Giannoutakis, G.A. Gravvanis, B. Clayton, A. Patil, T. Enright and J.P. Morrison, "Matching high performance approximate inverse preconditioning to architectural platforms," *The Journal of Supercomputing*, to appear
- [13] G.A. Gravvanis, "Domain decomposition – finite difference approximate inverse preconditioned schemes for solving fourth order equations," *Journal of Mathematical Modelling and Algorithms*, 3, 2002, pp 181-192.
- [14] G.A.Gravvanis, "Explicit Approximate Inverse Preconditioning Techniques," *Archives of Computational Methods in Engineering*, 9(4), 2002, pp 371-402.
- [15] G.A. Gravvanis, "A note on the rate of convergence and complexity of domain decomposition approximate inverse preconditioning," *Computational Fluid and Solid Mechanics, Proceedings of the First MIT Conference on Computational Fluid and Solid Mechanics*, eds. K.J. Bathe; Elsevier, 2, 2001, pp 1586-1589.
- [16] G.A. Gravvanis, V.N. Epitropou and K.M. Giannoutakis. "On the performance of parallel approximate inverse preconditioning using Java multithreading techniques", *Applied Mathematics and Computation*, to appear.
- [17] Y. Huang, C. Kintala, N. Koletis and N. D. Fulton. "Software rejuvenation: analysis, module and application". *Proc. of 25th Symposium on Fault Tolerant Computing*, 1995, pp.381-390.
- [18] V.P. Koutras and A. N. Platis, "Applying software rejuvenation in a two node cluster system for high availability," *Proceedings of International Conference on Dependability of Computer Systems*, 2006, pp. 175-182.
- [19] K. Krauter, R. Buyya and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," *Software Pract Exper*, 32(2), 2002, pp. 135-164.
- [20] A. Kumar, "An efficient SuperGrid protocol for high availability and load balancing," *IEEE Trans Comput.*, 49(10), 2000, pp. 1126-1133.
- [21] G. Levitin and Y.S. Dai, "Service reliability and performance in grid system with star topology," *Reliability Engineering & System Safety*, 92(1), 2007, pp. 40-46.
- [22] C. Lucet, J. Manouvrier and J. Carlier, "Evaluating Network Reliability and 2-Edge-Connected Reliability in Linear Time for Bounded Pathwidth Graphs," *Algorithmica*, 27(3), 2000, pp. 316-336.
- [23] R.M. Smith, K.S. Trivedi and A.M. Rameshi, "Performability Analysis: Measures, an algorithm and a case study," *IEEE Transactions on Computers*, 37(4), 1998, pp. 406-417.
- [24] H. Stockinger, "Defining the Grid: A Snapshot on the Current View", *The Journal of Supercomputing*, to appear.
- [25] K.S. Trivedi, G. Giardo, M. Malhorta and R. A. Sahner, "Dependability and Performability analysis. Performance Evaluation of Computer and Communication Systems," *Lecture Notes in Computer Science*, L. Dontatiella, R. Nelson (eds), Springer-Verlag, 1993, pp. 587-612.
- [26] X. Shi, H. Jin, W. Qiang and D. Zou,"Reliability Analysis for Grid Computing," *In Proceedings of theThird International Grid and Cooperative Computing*, 1993, pp. 787-790.
- [27] D. Wang, W. Xie and K.S. Trivedi, "Performability analysis of clustered systems with rejuvenation under varying workload," *Perform. Eval.*, 64(3), 2007, pp. 247-265.