Reliability Analysis of Grid Computing Networks with PetriNet

Navid Reza Gholizadeh¹, Mahdi Bazargani² and Nasrin Mohammadi Golbous³

Abstract—Grid computing systems are distinct from current distributed systems due to focus on too much common resources. In these systems, processors and communications have a significant impact on reliability. A useful criterion in analysis of grid network systems is the probability of successful implementation of all programs distributed throughout the network, so it is important to be able to access to the required files even in case of failure. In this article, PetriNets, a tool as modeling, and then evaluation method of reliability in grid computing network and its modeling with PetriNet and the related tool are surveyed first.

Keywords: Grid computing, Petri Net, Reliability, Distributed Systems
I. INTRODUCTION

Nowadays, advancing in different sciences without using computer is practically impossible. Grid computing networks, modern architecture of memory resources and data processing prepare a large volume of servers, supercomputers and memories as a flexible and required resource for all computing and storage requests of companies. In technology arena, grid computing is being progressed. Pioneers of information technology arena such as: IBM, Oracle and Sun Microsystems companies propose the technology based on grid and many other companies are preparing to adopt this new technology.[1]

The most common description for grid computing is to compare it with the power network. When an electrical device is plugged to the city power, you expect convenient voltage to be available for operating the desired device, while the actual source of electricity is unknown for you. Regional electricity company provides an interface in a complex network consisting of power generators and gives an acceptable quality of service to supply electrical energy. Electricity network infrastructure is actually a virtual generator with high reliability and compatible with consumers needs of energy. There is also a similar approach for grid computing. A virtual computer with high reliability and compatible with the needs will be accessible to the users. This virtual computer consists of many different computational resources but these resources are not individually visible to users. As, electric energy consumer is not informed of how electricity is produced, therefore, the initial purpose is considering the computations as a convenient and present facility everywhere. While the network gives users the possibility to share the information everywhere, the grid allows users to share the computational power with each others. [5]

Today, companies are looking to reduce their costs and increase the systems and processes performance. Grid computing network, increases commercial resources efficiency with a way to unite the hardware for processing and removing virtual parts, so, you can create a centralized complex of computational resources and dynamically allocate them to priority of your organization. Based on anticipations, in the next ten years, scientists will have a tool which put the computers connected to internet or any wide communication network as a source of powerful computation and a unit under their control. This powerful tool will be the grid.

II. PETRI NET, A TOOL FOR MODELING

PetriNet is a tool for study of systems. PetriNet theory allows a system to be modeled and providing the mathematic of this system. Very useful informations obtain from dynamic behavior and structure of the system modeled by PetriNet analysis which can be used to evaluate and surmise the improvement or alterations in the system. Therefore, the development of PetriNet theory is based on its application in modeling and system design. PetriNets are designed so that, can specifically model the type of systems consisting of components simultaneous with interaction effect. Mr. Carl Adam Petri introduced the PetriNet in 1962 which was regarded by MIT and several conferences have held on about that from 1970 - 1975 to next. He obtained most of his developments from working on information system and certain groups were formed in Germany and many other countries to develop their researches on PetriNets application.[3]

Scientific application for systems design and analysis is possible in several methods. One of these methods is the description of PetriNet as an auxiliary tool to analyze the system. For this application, special design techniques are used to define the system. Then, the system is modeled as PetriNet and the model will be analyzed. From analysis viewpoint, it is important to find the error in present design and the design should be modeled to correct the errors. This defined design can be modeled and analyzed frequently. To design by PetriNet, the above application needs to convenient convertors between the designed system and PetriNet model. There is a mutual option in which all designs and processes definition are performed as the parts of PetriNet. Analysis technique is only performed when PetriNet design is error-free. Thus, the transition
between PetriNet and a real system is working.[6]

These two above perspectives necessitate different researches on PetriNet. At first, modeling techniques should be developed to convert the systems to PetriNet model and in the second, implementation techniques for transferring the PetriNet model to the system, in both cases we need to know the properties of the PetriNet.[4]

PetriNet application is in modeling and analysis of the systems. At first, the system is modeled as PetriNet model and then it analyzed. Correct understanding of analysis results conducts us to a useful system which is the researches on automated modeling and analysis techniques (the model converts complicated system to simple system).

III. RELIABILITY ANALYSIS

In this part, according to figure 1 the reliability results were produced using nationwide repair models and these results validated utilizing probable expressions and equations based on MFST. For large models, these expressions could be produced using software packages such as SYREL which are designed to evaluate multi-terminal reliability. Version 5 of software package (SPNP) is used for modeling the obtained model. [7]

Consider a six node system according to figure 1 which indicates the allocated locations for nodes and links.

The assumed program p1 will be run in nodes No. 1 and 6 with FN={F1,F2,F3}. The allocations made for files and programs are shown as configuration a1 in figure 1. Logical expression (g), was obtained by counting the spanning trees of minimum file :

\[ g = g_{12}g_{31}g_{12}g_{13}g_{12}g_{23}g_{12}g_{3g} g_{13g}g_{24}g_{5g}g_{6g}g_{45g}g_{56g}+g_{4g}g_{5g}g_{6g}g_{45g}g_{46}+g_{4}g_{5g}g_{6g}g_{56g}+g_{g}g_{12}g_{4g}g_{12}g_{4g}g_{5g}g_{13}g_{35}g_{45g}+g_{4g}g_{5g}g_{6g}g_{23}g_{24}g_{46}+g_{g}g_{12}g_{5g}g_{6g}g_{23}g_{35}g_{5}g_{6} \]

An anticipation of how an assumed configuration reacts against various elements and undiscovered failures may be caused by both hardware and software failures, is a possible approach in program reliability analysis.[2]

For configuration a1 shown in figure 1, table 1 shows program reliability p1 assuming two values for coverage factor (c=1, c=0.95) and two values for failure coefficient r (r=0, r=0.001). Failure coefficient was defined as the ratio between links failure rate \( \lambda_e \) and nodes failure rate \( \lambda_n \). (Here, all \( \lambda_{ij} \) are assumed equal, so, \( r=0 \) means all the links assumed free failure, while \( r=0.001 \) means the nodes suffer damage 1000 times more than links. In any case, \( \ldots \) is assumed. Due to existence of two copies of program p1, links failures supposing nodes failure have the low effect in ratio to the gained results. Although, when the value of c decreases from 1 to 0.95, mean time to failure (MTTF) significantly reduces.

![Figure 1. Graph of a six node system](image)

<table>
<thead>
<tr>
<th>T</th>
<th>r=0</th>
<th>r=.001</th>
<th>r=0</th>
<th>r=.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>c=1</td>
<td>0.95</td>
<td>0.95</td>
<td>0.95</td>
<td>0.93</td>
</tr>
<tr>
<td>0.5</td>
<td>0.48</td>
<td>0.48</td>
<td>0.48</td>
<td>0.45</td>
</tr>
<tr>
<td>0.7</td>
<td>0.29</td>
<td>0.29</td>
<td>0.29</td>
<td>0.26</td>
</tr>
<tr>
<td>0.9</td>
<td>0.16</td>
<td>0.16</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>1.0</td>
<td>0.12</td>
<td>0.12</td>
<td>0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>MTTF</td>
<td>0.56</td>
<td>0.56</td>
<td>0.53</td>
<td>0.53</td>
</tr>
</tbody>
</table>

Another approach in reliability analysis of these types of systems is that, the impact of different allocations of programs and files to be anticipated. Therefore, another configuration of programs and files is considered as a2 which the program p1 installed on the node no.1. But, all the required files remain unchanged. The function g for this configuration is as follows:
g=g1g2g3g12g23+g1g2g3g13g23+g1g2g4g5g12g24g45+g1g2g4g5g6g12
g3g13g23+g1g2g3g12g24g45+g1g2g4g5g6g12

A third configuration is also considered in which program p1 installed on x1, x2, x6 and similar distribution of the required files was assumed to previous configuration:

g=g1g2g3g12g23+g1g2g3g13g23+g1g2g3g12g24g45+g1g2g4g5g6g12

The diagrams shown in figure 2 indicate configurations effects of a1, a2, a3, r=0 assuming c=1 and error-free links. Nodes failure rate is supposed by \( \lambda_i \). The gained results is compared with a centralized configuration (a4) in which a copy of program p1 and all required files are located in the same node. If this node is the node no.1, so, g=g1 and program reliability is equal to this node reliability.

With considered assumptions, the results gained from configuration a3 show significant improvement in reliability and MTTF in comparison to configuration. In centralized state (a4), this improvement has been limited for set of chosen parameters in a short period of time.

IV. THE EFFECT OF NATIONWIDE REPAIRS

Consider a situation in which c=1, means all hardware failures has been discovered and all softwares are responding in expected time. Numerical results have been listed in table 2 for configuration a1. When the failures occur, a repair is done, not necessarily after the failures discovery, but with hardware support for the desired application.

This model is useful for estimating program availability which corresponds to availability of system hardware support for the program or the desired application.

Suppose that, in a month a defect is averagely discovered. For a mean time to repair (MTTR), a1 configuration is analyzed in 1, 10 and 24 hours. For investigation the effect of lower failure rates, the availability results a1 are plotted in diagram on figure 3 for nodes \( \lambda_i = 0.33 \) (except of the node no.1 \( \lambda_1 = 1 \)).

Figure 3 shows p1 availability program in six node system under a1 configuration with 1 hour mean time to repair and the previous supposed failure rates.

Now consider c<1 which repair team react on the reported failures. Because, undiscovered hardware or software failures were reported, the models could be developed to investigate the factor c in 3 independent circumstances:

a. Occurrence of discovered hardware failures

b. Occurrence of communication failures

c. Occurrence of failures related to software

If it is assumed that all hardware failures were detected then c is equal to 1 and it is possible that the system to be crashed
because of software or communication failures or errors on message transfer.

![Figure 3. The effect of hardware repair in p1 reliability program](image)

**V. CONCLUSION AND PROPOSALS**

Reliability analysis can be extended in nodes considering more assumptions like more life span and equipment technology. These conditions can also be considered in nodes with different failure rates for heterogeneous computer networks.

Similar models can be used for various configurations if a Stochastic Petri Net (SPN) is obtained for a system, since SPN elements including the links and nodes are not involved in activation functions. Therefore, the complexity of the produced models is specified by set of failure rates, internal connection structure and the size of function g. Modeling the internal connection structure of the system by immediate transition capacity, reduces Markov chain state space. For example, suppose that all nodes and links may confront to failure, so, 201 states are totally produced for configuration a1. While, the states reach to 2185 without considering immediate transitions in SPN model. But, both models show the same results.

Before running a program in grid computing system and distribution of files and programs throughout the network, an estimation of system reliability can be achieved for spanning tree extraction based on the assumed configuration, possible modes of distribution using present or genetic algorithms. Also, for simulation and analysis of the achieved models, other tools such as Sharpe or Mobius can be used.

**REFERENCES**


