ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014

NGA based Load Balancing in Computational Grid

Smitha.T.Chandran, A.N Karthikeyan

Abstract: In the present Grid computing environment, the scheduling approaches for resources only focus on the current state of the entire system. Most often they fail to consider the system variation and historical behavioral data which causes system load imbalance. To present a better approach for solving the problem of resource scheduling in a Grid computing environment, this project demonstrates a genetic algorithm based resource scheduling strategy that focuses on system load balancing. The genetic algorithm approach computes the impact in advance that it will have on the system after the new resource is deployed in the system, by utilizing historical data and current state of the system. It then picks up the solution, which will have the least effect on the system. By doing this it ensures the better load balancing and reduces the number of dynamic migrations. The approach presented in this project solves the problem of load imbalance and high migration costs. Usually load imbalance and high number of migrations occur if the scheduling is performed using the traditional algorithms.

***Keywords*- Computational grid, Fault tolerance, Job scheduling, Load balancing.**

I. INTRODUCTION

Computational grid is an aggregation of geographically distributed network of computing nodes specially de-signed for compute intensive applications. GRID computing has emerged as the next-generation parallel and distributed computing methodology that aggregates dispersed heterogeneous resources for solving various kinds of large-scale applications in science, engineering, and commerce. In large-scale grid environments, the underlying network connecting them is heterogeneous and bandwidth across resources varies from link to link [3]. Not limited to grid, in many of today's distributed computing environments (DCEs), the computers are linked by a delay and bandwidth limited communication medium that inherently inflicts tangible delays or inter resource communications and load exchange[4]. To be able to fully benefit from such grid systems, resource management and scheduling are key grid services, where issues of task allocation, load balancing and fault tolerance represent a common challenge for most grids [2].

In the present Grid computing environment, the scheduling approaches for resources only focus on the current state of the entire system. Most often they fail to consider the system variation and historical behavioral data which causes system load imbalance. To present a better approach for solving the problem of resource scheduling in a Grid computing environment, this project demonstrates a genetic algorithm based resource scheduling strategy that focuses on system load balancing. The genetic algorithm approach computes the impact in advance that it will have on the system after the new resource is deployed in the system, by utilizing historical data and current state of the system. It then picks up the solution, which will have the least effect on the system. By doing this it ensures the better load balancing and reduces the number of dynamic migrations. The approach presented in this project solves the problem of load imbalance and high migration costs. Usually load imbalance and high number of migrations occur if the scheduling is performed using the traditional algorithms [7].

In the current Grid computing environments, resource scheduling only considers communication cost and replication cost ignores the current system condition and the previous state of system which causes the system load imbalance. Number of migrations is more when most of the load balancing takes place. The entire migration cost becomes a problem when all the resources are migrated this is worse than replication and communication cost. This is largely due to granularity of resources and the large amount of data transferred in the migration with suspension of service. This system does not consider these factors they only consider communication and replication cost. This project provides a scheduling strategy to enable effective load balancing. The method used in this project will compute its influence on the system in advance, when current resources are allocated to every physical node and will opt for the deployment that will have the least load on the system. This is achieved using genetic algorithm, historical data and the current state of the system.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014

In this paper, develop a future prediction mechanism based Genetic Algorithm for Grid Load balancer considering communication cost and replication cost. Implement a new algorithm which considers both present and future loads.

II. PROPOSED SYSTEM

Develop a future prediction mechanism based Genetic Algorithm for Grid Load balancer considering communication cost and replication cost. Implement a new algorithm which considers both present and future loads. A data mining system is used for future prediction. Heart of Genetic Algorithm is a weight factor. We develop a mathematical formula for calculating weight for each allocation. This formula considers waiting time, idle time, communication cost, replication cost, node weight and future load.

Steps in the Genetic Algorithm:

1. Start-produce random population of n chromosomes (coding structure can be selected according to the problem domain).

2. Fitness-calculate the fitness value f(x) of every chromosome in the given population.

3. New population-generate the new population by reiterating the following steps till the creation of new population is done.

3.1. Selection-select two parent individuals from the population according to the fitness value.

3.2. Crossover-by using the crossover probability, generate the new off spring by reforming the parents.

3.3. Mutation-with the probability of mutation, mutate the new child at some positions.

3.4. Accepting-now the new offspring the part of next generation of population.

4. Replace-use the new generation as the current generation.

5. Test- if the stopping condition is satisfied then ends the algorithm and returns the individual with the highest fitness value.

6. Loop go to step 2.

Advantages of Proposed System

- Implement a new algorithm which considers both present and future loads.
- Considering communication cost, replication cost and also weight factor (processor speed, cache and memory).
- The main issue in the existing system i.e, security problems are also considered.
- Implementing a load balancing model in the real world grid environment.
- Employing passive replication scheme so memory wastage is avoided.

These systems have following modules:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014

- Grid Client:- submit job to Grid Scheduler
- Grid node:-Node which perform job execution
- Resource Manager Client:- Periodically sent load information to Resource Manager
- Resource Manager:- collect resource information various Grid Nodes and calculate processing capability of each node
- Grid Scheduler: - schedules job to various nodes based on NGA results.

A. *THE MODEL:*

The relationship between the physical machines and the s can be seen from figure 1. Consider P as a set of all the physical machines in the entire system, where P = {P1, P2, P3 …. PN}. N is total number of the physical machines and an individual physical machine can be denoted as Pi, where i denote the physical machine number and range of i is (1 <= i <= N). Similarly, we have a set of jobs s on each physical machine Pi, Vi = {Vi1, Vi2, Vim} here m is the number of s on the physical server i .

If we want to deploy V on the present system, then we have a solution set denoted by S = {S1, S2, S3 …. SN}, it represents the mapping solution after V is assigned to each of the physical machines. When the V is arranged with the physical machine Pi we get the mapping structure denoted as Si.

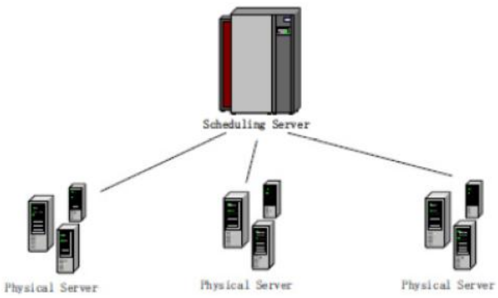

Fig1: System model

B.THE MATHEMATICAL MODEL:

Here we propose a model for broker mechanism that allocates jobs to different VM according to our criteria. For that we use genetic algorithm. By using this algorithm, we generate different job scheduling sequence and select best sequence. Best sequence selection is based on a rank. This rank calculation is shown below.
How weight is calculated

$$R = \sum_{1}^{Nj} \frac{\frac{Wi}{Max(Wi)}}{Nj} + \sum_{1}^{Nj} \frac{\frac{Max(Pi)-Pi}{Max(Pi)}}{Nvm} + \sum_{1}^{i=Nvm} \frac{\frac{Ti}{Max(Ti)}}{Nvm} + \sum_{1}^{Nj} \frac{\frac{Min(Ci)-Ci}{Max(ci)}}{Nj}$$

R- Rank
N_j – Number of jobs
W_i – Weighting time of i^{th} Job
P_i – Communication Cost of i^{th}Job
N_{vm} – Number of Jobs
T_i – Idle time i^{th} VM
C_i- Replicationcost of i^{th} Job.

C. *THE GENETIC ALGORITHM:*

Genetic algorithm is a random searching method that has a better optimization ability and internal implicit parallelism. It can obtain, and instruct the optimized searching space and adjust the searching direction automatically through the optimization method of probability [7]. With the advantages of genetic algorithm, this

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014
project presents a balanced scheduling strategy of resources in Grid computing environment. By considering the current states and historical data, this method will compute in advance on its influence over the entire system.

Methodology:

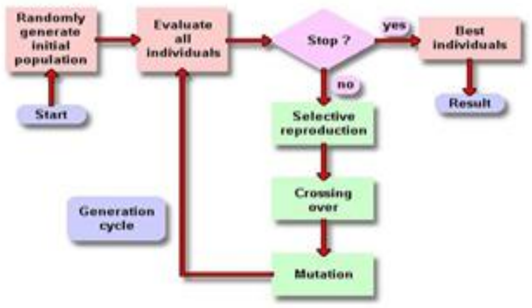

Fig 2: Methodology

In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible [24].

The evolution usually starts from a population of randomly generated individuals and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

A typical genetic algorithm requires:

1. A genetic representation of the solution domain,
2. A fitness function to evaluate the solution domain.

A standard representation of each candidate solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size, which facilitates simple crossover operations. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in genetic programming and graph-form representations are explored in evolutionary programming; a mix of both linear chromosomes and trees is explored in gene expression programming. Once the genetic representation and the fitness function are defined, a GA proceeds to initialize a population of solutions and then to improve it through repetitive application of the mutation, crossover, inversion and selection operators.

Initialization:
Initially many individual solutions are (usually) randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, allowing the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.
Selection:
During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014
preferentially select the best solutions. Other methods rate only a random sample of the population, as the former process may be very time-consuming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid or 0 otherwise.

In some problems, it is hard or even impossible to define the fitness expression; in these cases, a simulation may be used to determine the fitness function value of a phenotype (e.g. computational fluid dynamics is used to determine the air resistance of a vehicle whose shape is encoded as the phenotype), or even interactive genetic algorithms are used.

Genetic operators:
For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each new child, and the process continues until a new population of solutions of appropriate size is generated. Although reproduction methods that are based on the use of two parents are more "biology inspired", some research suggests that more than two "parents" generate higher quality chromosomes.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions, for reasons already mentioned above.

It is worth tuning parameters such as the mutation probability, crossover probability and population size to find reasonable settings for the problem class being worked on. A very small mutation rate may lead to genetic drift. A recombination rate that is too high may lead to premature convergence of the genetic algorithm. A mutation rate that is too high may lead to loss of good solutions unless there is elitist selection. There are theoretical but not yet practical upper and lower bounds for these parameters that can help guide selection.

Termination:
This generational process is repeated until a termination condition has been reached. Common terminating conditions are:
• A solution is found that satisfies minimum criteria.
• Fixed number of generations reached.
• Allocated budget (computation time/money) reached.
•The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results.
.• Manual inspection.
• Combinations of the above.

III. DISCUSSION

In the present Grid computing environment, the scheduling approaches for resources only focus on the current state of the entire system. Most often they fail to consider the system variation and historical behavioral data which causes system load imbalance. To present a better approach for solving the problem of resource scheduling in a Grid computing environment, this project demonstrates a genetic algorithm based resource scheduling strategy that focuses on system load balancing.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014

Several other algorithms such as Ant colony scheduling algorithm[16], Load Balanced Min-Min Algorithm[5] for Static Meta-Task Scheduling in Grid Computing, DMHS algorithms[4] are used for scheduling the task in grid computing.ACO algorithm can be interpreted as parallel replicated Monte Carlo (MC) systems. This algorithm is used to search for the best tasks scheduling for grid computing. This algorithm does not consider security and rapid dynamic change.

Load Balanced Min-Min (LBMM) algorithm is proposed that reduces the make span and increases the resource utilization. The tasks are rescheduled to use the unutilized resources effectively. This algorithm does not considering low and high machine heterogeneity and task heterogeneity.The main idea of the DMHS algorithms is to execute jobs optimally. Workload traces are required to produce dependable results is the issue. Using passive replication schema and Security is not considered.

A hybrid load balancing strategy [8], Static strategy and dynamic adjustment[12], A two way load balancing policy, competitive equilibrium approach are decentralized load balancing policy[25] for the grid computing environment. Main objective is to minimize the overall job mean response time and maximize the system utilization and throughput. It does not consider current and future loads. In these existing system does not consider suboptimal solution on a multi node system.

Genetic algorithm is a random searching method that has a better optimization ability and internal implicit parallelism. It can obtain, and instruct the optimized searching space and adjust the searching direction automatically through the optimization method of probability. With the advantages of genetic algorithm, this project presents a balanced scheduling strategy of resources in Grid computing environment.

IV.CONCLUSION

One of the key concerns of grid computing is to develop autonomic computing systems that have the abilities of self configuration and self-optimization in dynamic environment. We generate a new future prediction mechanism based genetic algorithm which considers communication cost and replication cost. This new algorithm considers both present and future loads. Data mining system is used for future prediction. Existing system only considers present system and doesn't consider current allocation and future loads. Issues related to security have not been considered in the existing system. Prediction mechanism and this NGA don't consider rapid dynamic change in load. At that situation, large amount of resource is wasted due to error in allocation.

REFERENCES

[1] Jasma Balasangameshwara and Nedunchezhian Raju,(2013) Performance-Driven Load Balancing with a Primary-Backup Approach for Computational Grids with Low Communication Cost and Replication Cost, IEEE transactions on computers, vol. 62, no. 5.

[2] Reza Salimi*, Navid Bazrkar, Mostafa Nemati,(2013),Task Scheduling for Computational Grids Using NSGA II with Fuzzy Variance Based crossover, College of Computer Science, Tabari Institute of Higher Education,Babol,Iran.

[3] Prabhat Kr.Srivastava, Sonu Gupta, Dheerendra Singh Yadav,(2011),Improving Performance in Load Balancing Problem on the Grid Computing System, International Journal of Computer Applications (0975 – 8887) Volume 16– No.1.

[4] Saga Dhakal, Majeed M. Hayat, Jorge E. Pezoa, Cundong Yang, and David A. Bader,(2007), Dynamic Load Balancing in Distributed Systems in the Presence of Delays: A Regeneration-Theory Approach, IEEE Transaction on parallel and distributed systems Vol 18, NO. 4.

[5] T. Kokilavani, Dr. D.I. George Amalarethinam,(2011), Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing, International Journal of Computer Applications (0975 – 8887) Volume 20– No.2.

[6] Hai-yun PENG1, Qian LI1,(2011), One Kind of Improved Load Balancing Algorithm in Grid Computing, International Conference on Network Computing and Information Security.

[7] S. Prakash, D. P. Vidyarthi,(2011), Load Balancing in Computational Grid Using Genetic Algorithm, School of Computer & Systems Sciences, Jawaharlal Nehru University, New Delhi, India.

[8] Yajun Li, Yuhang Yang, Rongbo Zhu,(2009), A Hybrid Load balancing Strategy of Sequential Tasks for Computational Grids International Conference on Networking and Digital Society.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 2, March 2014

[9] Majeed M. Hayat, Sagar Dhakal, Chaouki T. Abdallah, J. Douglas Birdwell, and John Chiasson,(2009), Dynamic Time Delay Models for Load Balancing Part II: A Stochastic Analysis of the Effect of Delay Uncertainty.

[10] David Fernandez-baca,(1989), Allocating Modules to Processors in a Distributed System, IEEE transactions on software engineering vol. 15. NO. 11.

[11] Marek Wieczorek, Andreas Hoheisel , Radu Prodan,(2009), Towards a general model of the multi-criteria work flow scheduling on Future Generation Computer Systems (2009) 237_256he grid.

[12] Peijie Huang, Hong Peng, Piyuan Lin, Xuezhen Li,(2009), Static strategy and dynamic adjustment: An effective method for Grid task scheduling, Future Generation Computer Systems 25 (2009) 884_892.

[13] Said Fathy EI-Zoghdy, Sultan Aljahdali,(2008), A Two-Level Load Balancing Policy For Grid Computing, Computer Science Dep., College of Computers & Information Technology, Taif University, AIF, KSA,Alzoghdy@tu.edu.sa.

[14] Junwei Cao1, Daniel P. Spooner*, Stephen A. Jarvis*, and Graham R. Nudd,(2003), Grid Load Balancing Using Intelligent Agents, C&C Research Laboratories, NEC Europe Ltd., Sankt Augustin, Germany Department of Computer Science, University of Warwick, Coventry.

[15] Jonathan White and Dale R. Thompson,(2007), Load Balancing on a Grid Using Data Characteristics, Computer Science and Computer Engineering Department University of Arkansas Fayetteville, AR 72701, USA.

[16] Stefka Fidanova and Mariya Durchova,(2005), Ant Algorithm for Grid Scheduling Problem, IPP – BAS, Acad. G. Bonchev, bl.25A, 1113 Sofia, Bulgaria.

[17] Fotohi, Mehdi Effatparvar(2013), A Cluster Based Job Scheduling Algorithm for Grid Computing, I.J. Information Technology and Computer Science, 2013, 12, 70-77 Published Online November 2013 in MECS.

[18] S. Gokuldev, Shahana Moideen, (2008),Global Load Balancing and Fault Tolerant Scheduling in Computational Grid, International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 11, May 2013.

[19] Ping Luoab, Kevin LÄuc¤, Zhongzhi Shia and Qing Hea,(2011), Distributed Data Mining in Grid Computing Environments, a Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences,100080.

[20] Sandeep Kaur1, Sukhpreet Kaur2,(2013), Survey of Resource and Grouping Based Job Scheduling Algorithm in Grid Computing, IJCSMC, Vol. 2, Issue. 5, May 2013, pg.214.

[21] David Abramson, Rajkumar Buyya, Jonathan Giddy, (2002) A computational economy for grid computing and its implementation in the Nimrod-G resource broker, Future Generation Computer Systems 18 (2002) 1061–1074.

[22] Amir Masoud Rahmani and Mojtaba Rezvani, (2009), A Novel Genetic Algorithm for Static Task Scheduling in Distributed Systems9), International Journal of Computer .

[23] R. Kashyap, D.P. Vidyarthi,(2011), Security-Driven Scheduling Model for Computational Grid using Genetic Algorithm, (IJEIT) Volume 2.

[24] K Shahu Chatrapati#, J Ujwala Rekha, Dr. A. Vinaya Babu*,(2005), Competitive equilibrium approach for load balancing a computational grid with communication delays, Journal of Theoretical and Applied Information Technology.

[25] Syed Nasir Mehmood Shah*, Ahmad Kamil Bin Mahmood, Alan Oxley, Dynamic Multilevel Hybrid Scheduling algorithm.

AUTHOR BIOGRAPHY

Smitha.T.Chandran, IInd Year M.E, Department Of Computer Science, Mahendra Institute Of Technology, Mallasamudram, Tiruchengode,Tamilnadu,India

A.N.Karthikeyan, Asst.Professor, Department Of Computer Science, Mahendra Institute Of Technology, Mallasamudram, Tiruchengode, Tamilnadu,India.