

Genetic Optimization of Hybrid Load Balancing Algorithm for Grid Computing Environment

¹Jyoti Prakash, ²Satya Prakash Sahoo, ³Manas Ranjan Kabat

^{1,2,3}Dept. of CSE, VSS University of Technology Odisha, Burla, Sambalpur, India

Abstract

Recent advances in internet technology has lead to the formation of distributed computing environment, geographically spreading over the globe for solving high-end computational tasks. To increase the potential of such large-scale distributed systems, it requires proper load-balancing algorithm arises. Since the problem of Load Balancing is NP-Complete, therefore it requires heuristic or approximation algorithm. In this paper, we have conducted an experimental verification of Genetic Optimization of the Hybrid Load Balancing Algorithm, by considering Value Function as optimization parameter. With throughput as performance parameter, we have evaluated the algorithm on a grid based on MetaCentrum Project (Czech Republic National Grid Infrastructure). Further, we have used a Check Point Based Recovery Technique in establishing the reliability of the grid.

Keywords

Grid Computing, Genetic Algorithms, Bio-Inspired Computing, Load Balancing, High Performance Computing

I. Introduction

Rapid development in communications fueled by requirement of High-Performance Computing Environment has paved way for Grid Computing. Grid Computing is based on computing as a utility inspired by "Electric Power" being delivered as a utility. One of the examples of Grid Computing project, by India is "Garuda" or Global Access to Resources using Distributed Access. It is a national Grid Project of CDAC, connecting all the national educational institutions and research institutions for enhancing quality of Higher education and research. It promises to provide cost-effective solutions to access to computing resources.

With demand of computers for scientific computations, the demand for powerful and high-performance computing devices has grown manifold. Proper distributions of load to the resources would define the efficiency of the system. Therefore, there arises a need for optimizing the usage of resources and increasing its efficiency.

II. Literature Review and Background

With the growth of Virtual Organizations in Grid Computing [9], Grid Computing came to be used in various global high-performance computing environments. Wang et. al. have given a novel approach of combining static load balancing and dynamic load balancing stages [3-7] which has been proposed as Hybrid Load Balancing Algorithm. Load Balancing algorithms comes under the category of global load balancing algorithm and dynamic scheduling algorithms [6]. Various approximate approaches have been proposed by various researchers such as Unconstrained FIFO, Cost-constrained FIFO, and Balanced Constrained FIFO etc. [6]. Balanced Constrained is a technique in which jobs are dynamically balanced between the waiting queues of different processors. Cost constrained technique employs a similar technique to balanced constrained but associates cost to different resources.

In regard of Load Balancing, various optimization functions have been proposed. A Load Balancing Algorithm may be Applications

centric or Resource Centric [6, 10], owing to the heterogeneity of the resources. Resource centric optimization optimized the usage of resources utilization and takes parameters such as throughput. If the resources and jobs are known before-hand, then optimal distribution could be determined. However, at times sub-optimal solution is also satisfied, owing to the NP-Complete nature of the job. Approximate algorithms such as Total Process Cycle Consumption Algorithm finds [4] a suboptimal solution, on a formal method of computation. However, heuristic approaches [2-4] of the problem have also shown better results. In a general sense, a Grid Load Balancing Algorithm has to deal with Resource Heterogeneity, Site Autonomy, Resources Non-dedication, Application Diversity, Resources Selection and Data-selection [5-6].

A. Hybrid Load Balancing Algorithm

A brief review of Hybrid Load Balancing Algorithm [2] is given here. It comprises of two stages, initially when a node is ready to give its resources to the grid, the GIS (Grid Information Service) collects all its parameters. On the submission of a job, it is broken up into many logical tasks and then distributed over the grid. When some node fails to provide services, then again the middle-ware (GIS) collects information about the resources and "re-balances" the load according to the availability of the resources. In this regard, Wang et. al. have introduced a concept of Value Function, defined as $VF = W_{i,j} f(x_i)$ [2] where i is the resource and j is the node and $W(i,j)$ is the weight given to that resource for that node.

B. Genetic Algorithm

Genetic Algorithm is a search heuristic that mimics the process of natural evolution. It belongs to a group of evolutionary algorithms that generate solutions to optimization problems using techniques inspired by nature such as mutation, cross-over etc. [11]. The basics step in genetic algorithm is

1. Population Generation
2. Selection according to fitness function
3. Cross Over
4. Mutation of the population
5. Maximization of fitness function
6. Termination when a suitable fitness value is obtained

C. Cross-Over Techniques

1. Single Point Crossover: This involves, exchanging the sub-strings at a single point.
2. Two Point Crossover: This involves selecting two points and exchanging the sub-strings among them.
3. K Point Crossover: It involves modifying the chromosomes at k-locations
4. Matrix Crossover: It is a two-dimensional cross-over technique.

III. Proposed Work

A. Checkpoint based Recovery Technique

Checkpoint based Recovery technique is based on storing checkpoints for sub-tasks of a particular task submitted to the grid. Here, we propose a theoretical frame-work for a Checkpoint based technique. Suppose a job S is divided into k sub-tasks. (see fig 1).

The task is divided into a number of independent sub-tasks and each sub-task is allocated to a resource according to the decision of the Load Balancing Algorithm. The sub-tasks are then collected into a group of dependent and independent sub-tasks, with each independent sub-tasks forming a group with its dependent sub-tasks. Each group is then allocated to a resource. After completion of each sub-task of a group, the outputs are reported to the logger and a checkpoint is created. Whenever, a node fails to provide the resources, then the task is restored from the latest checkpoint (see fig. 1).

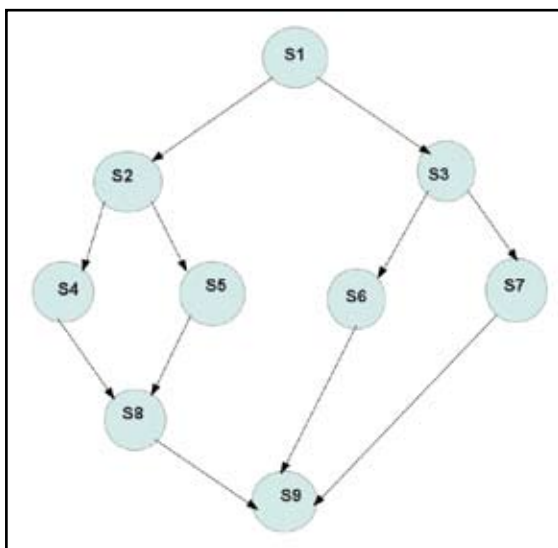


Fig. 1: Precedence Graph of a Task Being Broken Into Sub-Tasks

Algorithm 1 Checkpoint Recovery Technique

1. loop
2. For each task
3. Step 1: Divide it into a group of sub tasks S_n
4. such that S_i, S_j are independent and group the by
5. dependence
6. Step 2: Group sub-tasks such that sub-task S_i is
7. dependent on one sub-tasks
8. Step 3: Allocate Resources R_N to the group of
9. independent jobs S_n according to GVFA
10. For each group of jobs G_{ji} , allocate resource R_j to G_{ji}
- and keep check-point when each sub-task is completed
11. Step 4: If the resource R_j fails on some account, then
12. call GVFA and restore the sub-task from its
13. previous
14. checkpoint
15. end loop

B. Genetic Value Function Algorithm (GHVF)

Here, we propose a Genetic Hybrid Value Function Algorithm. This algorithm finds the optimal pair of Resources for allocation from the pool of resources.

Algorithm 2 Genetic Value Function Algorithm

1. Population Initialization: Initialize a pool of Resources and assign weight for various decision variables. Let it be $W[\text{no of resources}][\text{value of decision variables}]$
2. Initialize NODES[no of resources][no of decision variables]
3. Encoding: Use value-based encoding
4. Fitness Function: Maximize Value-Function
5. Value Function: For each node compute value function using the method described above.
6. Crossover: Apply k-point crossover to W
7. Recompute value function until it is maximized

C. Design of System Environment

Here, we propose a system environment, that can support the ideas presented above. It consists of a grid system (shown as 4-5 monitors). The grid system is in interaction with checkpoint system that stores the checkpoint of various sub-tasks.

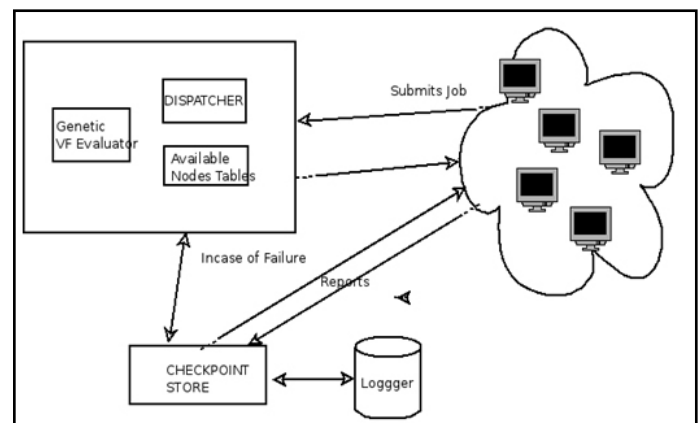


Fig. 2: Proposed Architecture of the System

Job is under execution and the nodes fails: When a job is under execution and the nodes fails, then the sub-task mapped to that node is recovered from its latest state stored in checkpoint. Then, Dispatcher calls Genetic Value Function Algorithm and new target destination for that sub-task is determined and mapped to that resource.

Node is working fine, but the sub-tasks take a longer time to execute: Whenever, a job takes much longer time to execute (e.g. local scheduling policy of the node), then it checks whether the time taken exceeds threshold value or not. If it exceeds, then the job is again restored from its previous checkpoint and again maps it to a new target resource (see fig. 2).

IV. Simulation Architecture and Results

The proposed algorithm was experimented with the help of GridSim Simulator on a Intel® Core™ 2 Duo Processor and Fedora R 16 (Verne) Platform using 2.0 GB RAM. The simulation programs were written in Java™ 2 Standard Edition using Eclipse IDE. GridSim is a discrete event simulator which facilitates simulation of Grid Resources. In this project, we have used the GridSim to simulate a Load Balancing. It provides creation of Gridlets which are jobs of specific computational length, along with the size of its input and output properties (8).

A. Architecture for Simulation

The architecture of the Grid Simulation consisted of a heterogeneous collection of individual machines and clusters. Each of the cluster or machine has a specific MIPS, and were under the administration of a single user. The role of the user was to create the jobs. Each of the network were connected to a network, with a link of specific bandwidth (see fig. 3).

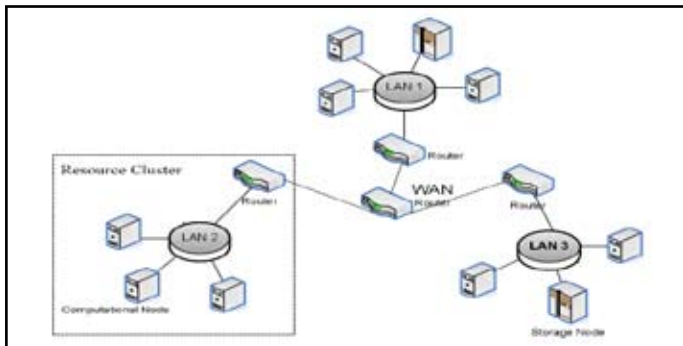


Fig. 3: Model of the Architecture Considered for Simulation

For the purpose of our simulation, we considered queuing policies such as FCFS, LIFO, CPU-Based and Value-Function based (1). Based on our model grid and jobs, we simulated the above policies and find its performance analysis by determining the execution time and average throughput of the system.

The simulation was conducted and three decision variables were taken. The variables are

- No of Processing Elements Free
- Bandwidth between the source to the target node
- Memory Free

Memory Free parameter was calculated as

Memory Free = Total Memory of the Resource - Memory in use at present

Processing Element Free is calculated as the number of processors idle in a particular resource. It can be calculated as

Processing Element Free = No of Processing Elements - No of Processing Elements in use

B. Results

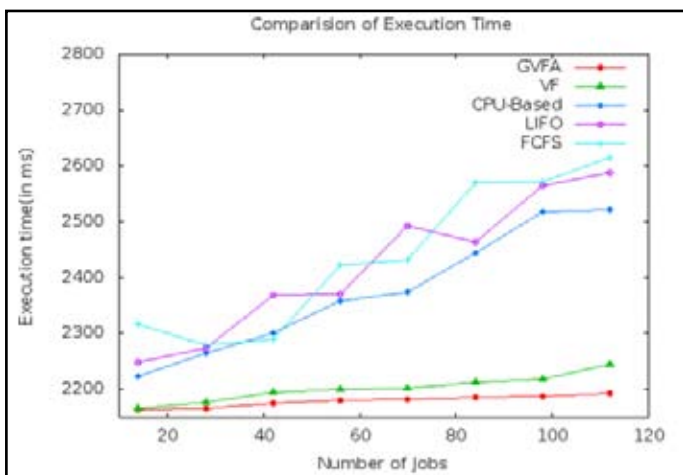


Fig. 4: Graph of Execution Time vs No of Jobs

We simulated for taking the number of jobs in multiples of 14 and their execution time in ms was observed. The algorithm was optimized for two-point crossover. Random set of weights were assigned to each node and their parameters (mentioned

above) and experiments was conducted. The simulation data was taken from <http://www.fi.muni.cz/~xklusac/meta2009/metacentrum.mwf.machines>. The simulation produced results of Execution Time are shown in Fig 4. Execution time for different values of k in k-point-crossover is shown in fig 5.

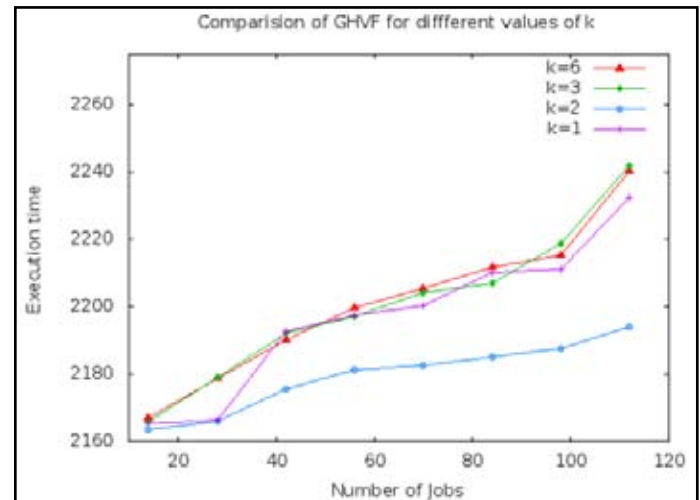


Fig. 5: Comparison of GHVF for Different Values of k

Fig 6 shows the throughput in MIPS for different policies.

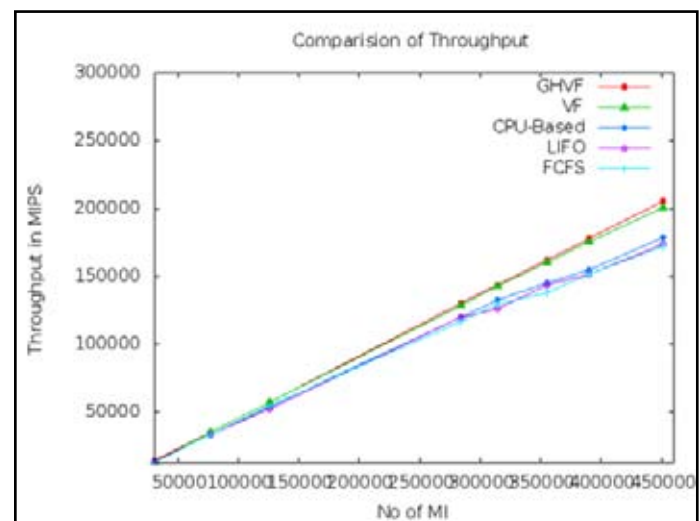


Fig. 6: Comparison of Throughput

V. Conclusion

In this paper, we have analysed our proposed algorithm with other task-scheduling policies. For evaluating the performance, we have taken throughput as performance criterion. From our Simulation, we have seen that throughput of our proposed algorithm is better than the previous policies. Further, we have proposed a Checkpoint based technique for recovering the subtask on account of failure of resource.

References

- [1] Yan, Wang, Chang, Lin, "A hybrid load balancing policy underlying grid computing environment, Elsevier", Computer Standards & Interfaces, pp. 29, 2007, pp. 161-173
- [2] Javier Carretero, Fatos Xhafa, "Genetic Algorithm based Schedulers for Grid computing environment", International Journal of Innovative Computing, Information and Control Volume 3, Number 6, December 2007.
- [3] Yagoubi, Lilia, Moussa, "Load Balancing in Grid computing", MedWell Online, Asian Journal of Information Technology

pp. 5(10), pp. 1095-1103, 2006

- [4] Fangpeng Dong, Selim G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems", Technical Report No. 2006-504, School of Computing, Queens University Kingston, Ontario, January 2006
- [5] Belabbas Yagoubi, Yahya Slimani, "Dynamic Load Balancing Strategy for Grid Computing", World Academy of Science, Engineering and Technology, pp. 19, 2006, 90-95
- [6] Carsten Ernemann, Volker Hamscher, Ramin Yahyapour, "Benefits of Global Grid Computing for Job Scheduling", Proceedings of Fifth IEEE/ACM International Workshop on Grid Computing
- [7] R.U. Payli, E. Yilmaz, A. Ecer, H.U. Akay, S.Chien, "DLB A Dynamic Load Balancing Tool for Grid Computing", Parallel CFD Conference, May 24-27, 2004, Grand Canaria, Canary Islands, Spain.
- [8] Manzur Murshed , Rajkumar Buyya, David Abramson, "GridSim: A Toolkit for the Modeling and Simulation of Global Grids", Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2002), June 24 - 27, 2002, Las Vegas, USA.
- [9] Ian Foster, Carl Kesselman, Steven Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of High Performance Computing Applications Fall, 2001, 15, pp. 200-222.
- [10] Ian Foster, Carl Kesselman, "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufman Publications.
- [11] [Online] Available: http://www.en.wikipedia.org/Genetic_Algorithms



Mr. Jyoti Prakash, is a student of B.Tech. IT of VSS University of Technology. His interests include Distributed Computing, Operating Systems and Linux Kernel Programming.



Mr. Satya Prakash Sahoo is Lecturer in Department of Computer Science and Engineering at VSS University of Technology, Burla. His research papers has been published and honored in various international conferences and journals in areas of Computer Networks, Soft Computing. His present areas of interests include Computer Networks, Artificial Intelligence and Soft Computing.



Dr. M.R. Kabat, is currently Lecturer (Senior Scale) in Department of Computer Science and Engineering at VSS University of Technology, Burla. His research papers has been published and honored in various international conferences and journals. His present areas of interests include Computer Networks, Real Time Systems, Artificial Intelligence and Soft Computing.