International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Job Scheduling with Fault Tolerance in Grid Environment Using Service Oriented Architecture

V. Indhumathi[1], G. M. Nasira[2]

[1]Reasearch Scholar, Department of Computer science, Periyar University, Salem
[2]Head / Computer Applications, Chikkanna Govt. Arts College, Tirupur, Tamilnadu
[1]sabarishindhu[at]gmail.com, [2] nasiragm99[at]yahoo.com

Abstract: ***Grid Computing constructs the vision of a easy and huge dominant self administration system exposed of great compilation of associated heterogeneous systems distribute a variety of resources which direct to the dilemma of load balancing. The major objective of load balancing is to offer a dispersed, low cost, proposal that stability the load diagonally the entire resources. Service Oriented Architecture (SOA) is a familiar devise model for dispersed systems. Its concepts of self-motivated and loose coupling are destined for self adaptive systems. Opening through a meaning of service and SOA, we explain and study diverse formation plus the method of service invocation with a five step model. in addition, we use Adaptive Service Oriented Architecture(ASOA) portray probable faults and describe SOA specific faults as a novel fault class that make bigger the identified fault set for dispersed systems with load balancing.***

Keywords: Grid Computing, Load balancing, SOA, Fault Tolerance, ASOA

1. Introduction

Grid is able to be distinct as a large-scale geographically dispersed hardware and software communications collected of diverse networked resources possessed plus common with various managerial organizations which are synchronized to offer apparent, reliable, persistent and steady computing sustain to a broad collection of applications. These applications are able to carry out moreover dispersed computing, elevated throughput computing, on-demand computing, data-intensive computing, collaborative computing or multimedia computing.

A faulty system suitable to several cause throughout dealing out a few job can origins a little compensation. A job consecutively on real time dispersed system be supposed to be practicable, consistent and scalable. The existent time dispersed system such as nuclear systems; robotics, air traffic control systems, grid etc. are extremely reliable on time limit. A fault in real time dispersed system be able to outcome a system into failure if not correctly perceived plus improved at time. As grid applications run in a extremely diverse computing situation, fault tolerance is significant in sort to make sure their accurate performance.

Center of attention on the subject of how to stipulation an application-specific policy to facilitate for using by a client toward scuttle a dispersed computing job, on the same time as gathering the task requirements and SLAs. To provision proposed baseline for a task, it have got to assign suitable computing and associated resources together. This type of architecture have a (logical) topology parallel with the aim of the assignment graph on behalf of that task with the exception of probably with supplementary boundaries and nodes for fault tolerance, make use of devoted but possibly possessions, in addition to are accomplished of energetic configuration to convene the dynamic necessities of the function. SOA is an architectural approach to creating systems built from autonomous services. The architectural concepts coupled with SOA are not new – many have evolved from ideas formerly initiated by CORBA, DCOM, DCE and others. SOA, however, is an architectural attitude and is not essentially an implementable perception. Service Orientation (SO) is the accepted development of present maturity models.

1.1 Literature Survey

The user's tasks negotiating with resource providers based on their essential Quality of Service and on the equivalent price to reach a Service Level Agreement using algorithm Quality Particle Swarm Optimization (QPSO) .Resource scheduling is carries out using Application heuristics Execution meeting user deadline. It implements above algorithm for mold and optimization of resource prophecy models based on Deadline distribution and planning distribution in V. Indhumathi (2016) "Improved Fault Tolerant in Workload execution Through Quality Particle Swarm Optimization for Grid Environment".[1] SOA focus on a new dynamic load complementary algorithm beside among fault tolerant scheduling approach through which successful load balancing and fault tolerance have been accomplished in V. Indhumathi and G. M. Nasira(2016) "Service Oriented Architecture for Load balancing with fault tolerant in Grid computing"[2].

Major achievements Improved Fault Tolerant in Workload Execution through Quality Particle Swarm Optimization for Grid Environment includes plan plus appraisal of scheme structural design for grid resource scrutinize and prophecy through Meta heuristic conditions in V. Indhumathi and Prof. Dr. G. M. Nasira (2015) "Resource Monitoring and Prediction with Fault Tolerance in Grid Environment through meta Heuristics"[3]. Energetic load balancing algorithm is planned in a pretend grid environment which fulfils the goal to attain elevated recital computing by best practice of geographically dispersed and diverse resources in the grid environment in Kapil B. Moreyl and Sachin B. Jadhav (2016)"Grid Computing approach for Dynamic Load balancing" [4].

2. Proposed System

Licensed Under Creative Commons Attribution CC BY

2.1 Service Oriented Architecture

Service-Oriented Architecture is broadly acknowledged as intend pattern for distributed systems. The Web Services standard uses the Web Service Description Language (WSDL) to describe services and SOAP for communication with services.

2.1.1. SOA structure

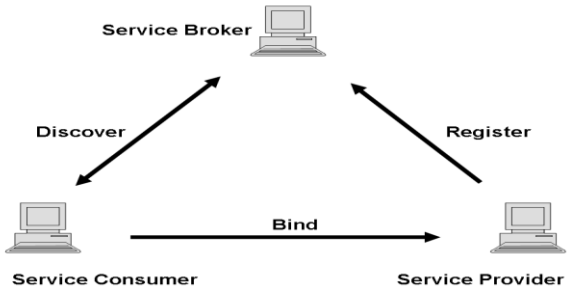


Figure 1: SOA Structure

Service providers record their services at the broker; service consumers demand a service from the service broker, which returns an identified provider for the requested service. Consumer and provider agree on the semantics. The consumer subsequently binds himself to the service provider and utilizes the service. The structure of this architecture is shown in Figure 1. Conversely, this is not the simply probable structural design for a SOA. The Adaptive Service Grid (ASG) project sees SOA as a grid which holds all service discovery and binding. Consumers request the grid for an exact service. The grid then strives to locate the particular service amongst its recorded service providers. Suppose it not identified the specified service the grid strives to arrange the service as out of the available ones based on the systematic description. The only communication partner for consumers respectively service providers is the grid (Figure 2).



Figure 2: ASG Platform Structure

2.1.2 Steps Used in SOA

SOA essentially includes of five steps: service publishing, service discovery, service composition, service binding, and service execution. These steps are explained in this section.

1) Publishing: Initially Service provider provides services over the network. The communication network provides services with their descriptions. All services have to be self-descriptive. In SOA Web services are portrayed by WSDL documents. The first version of WSDL was inadequate for a SOA. A complete SOA service depiction has to wrap all service belongings and semantics. So the prepared system like WSDL conservatory is worn for the explanation of non-functional properties.

2) Discovery: Consumer waits for a proper service to carry out preferred job. He should have apparent knowledge about requisite service from service description presented by service providers. Also the description of the specified service and the services provided by the service providers has to be compared with each other. If both are equal, the requisite service has been successfully discovered.

3) Composition: In case no one match with each other, it is possible to compose the service out of existing one. Service composition depicts the grouping of one or more services to one complex service. This is achieved by using either service choreography or service orchestration.

4) Binding: After finding out the suitable service, the consumer binds itself to the service for the execution. Precautions and AAA (authentication, authorization and accounting) concerns descend into this sort.

5) Execution: Formerly the service is bound to a consumer, the service or the service composition can be executed. The input parameters of the service are conveyed to the service provider, the service is compiled and result is returned to the consumer.

2.1.3 Fault tolerance methods

In the above, all the five steps face the problem of fault, which will lead to failure. For handling that failure, we propose SOA fault taxonomy structure for handling those types of errors, e.g. service break down preserve through replacement of those services or utilize through equal service by a diverse provider. In addition with SOA faults, the other faults can occur in distributed system such as hardware faults, software faults, network faults and etc.

In figure 3, service consumer submit job to Grid resource broker, whereas it split jobs to various resources. The status of the jobs sends to service provider for binding that job status with Grid resource information (GIS). GIS register job status with service broker for discovers which job is currently going to executed. After discovering, the job status sends to Grid resource broker for make scheduling that job.

Licensed Under Creative Commons Attribution CC BY

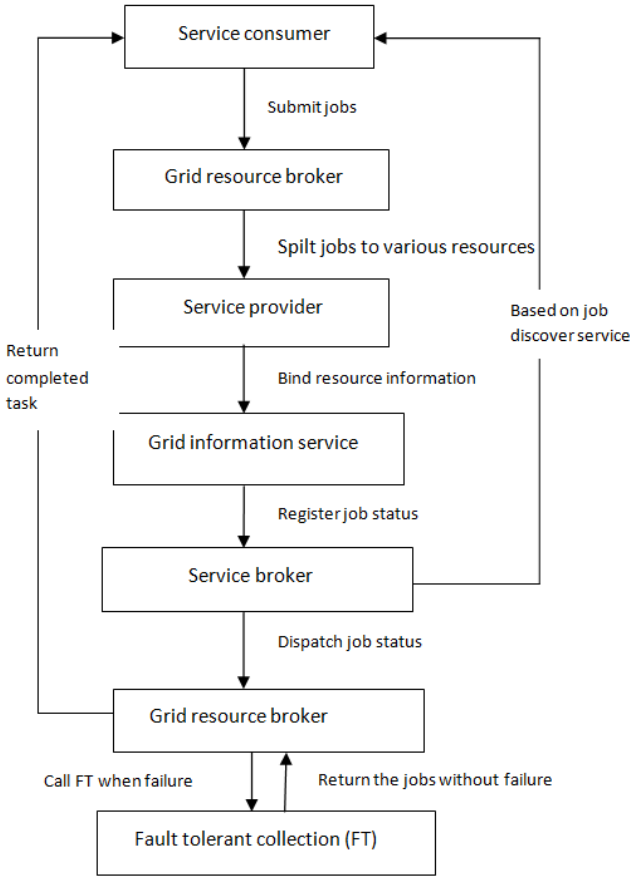


Figure 3: Proposed System Architecture

In case any fault will be occurring in that job scheduling that is recovered by using Fault tolerant Collection (FT) is illustrated in Figure4.

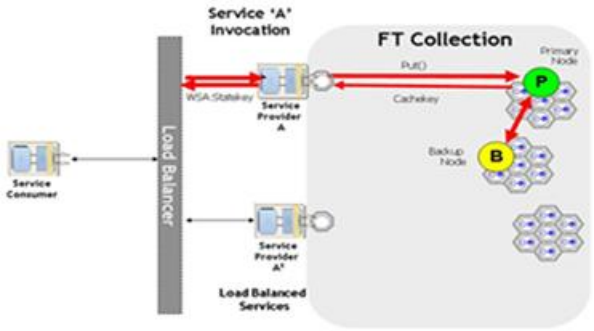


Figure 4: working model of FT Collection

2.1.4. ASOA scheduling Algorithm

Given N jobs, where every job is represented by three elements followed as start time, finish time and its value. Based on these elements we create sorted job list as SOL_t. Figure 5 illustrates ASOA scheduling algorithm. First step of this algorithm is receiving service request from service consumer. Sort the job list based on finishing time of each job, for example Number of jobs N=3 then the job sorted based on {start time, finish time, value}. (i.e.) Job 1: {1, 3, 50}, Job 2: {4, 6, 30}, Job 3: {7, 18, 150}.

Each request is processed based on time interval T. According that call service request available in waiting state $sr_{t,j}$ in T. calculate estimated execution time $et_{t,j}$. The sorted list $sr_{t,j}$ in SOL_t, for each service request, the Service provider provides services over network (publish). Consumer waits for proper service to carry out preferred service (discovery). In case no one match with each other, it is possible to compose the service out of existing one. Service composition depicts the grouping of one or more services to one complex service. After finding out the suitable service, the consumer binds itself to the service for the execution. Precautions and AAA (authentication, authorization and accounting) concerns descend into this sort. Formerly the service is bound to a consumer, the service or the service composition can be executed. The input parameters of the service are conveyed to the service provider, the service is compiled and result is returned to the consumer.

Algorithm for Job scheduling

Receive service request from service consumer
First Sort jobs based on finishing time.
For each time interval T:
Call service request in waiting state $sr_{t,j}$ in T.
Compute the estimated execution time, $et_{t,j}$
Sort the list $sr_{t,j}$ and generate sorted list SOL_t
For each service request in SOL_t:
Service provider provides services over network (publish)
Consumer waits for proper service to carry out preferred service (discovery)
Call Service composition
After binding suitable service, the consumer binds itself to the service for execution (Binding)
Service composition can be executed (Execution)
Return results to service consumer.

Figure 5: Adaptive SOA scheduling algorithm

3. Experimental Results

The Adaptive SOA scheduling algorithm (ASOA) sorts the service requests inputs and arbitrarily choose resources and hosts, which be able to fulfill the requirements of given service.

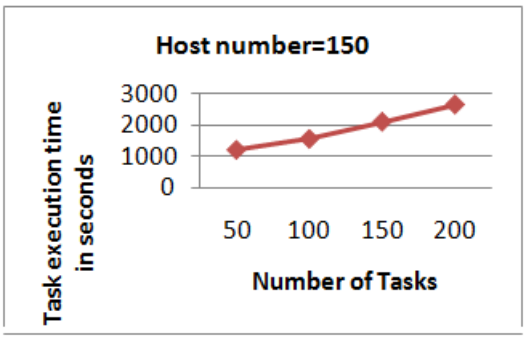


Figure 6: Performance of ASOA when increases

Licensed Under Creative Commons Attribution CC BY

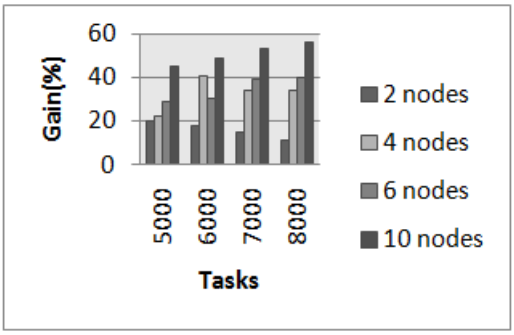


Figure 7: Mean response time during ASOA

When the host number is fixed to 150, Figure 6 shows the performance of our proposed ASOA scheduling algorithm. Figure 7 illustrates the improvement of the mean response time, obtained by our ASOA scheduling strategy. We can justify this by the instability of the Grid state (Most nodes are under loaded or even idle).

4. Conclusion

In this paper an algorithm that is adaptive, decentralized and distributed for task allocation among the aggregated resources in the grid. It is found that our proposed ASOA scheduling algorithm provides better performance when number of tasks increases and also provides better mean response time compared with previous algorithm used for resource allocation. Result of our proposed ASOA scheduling algorithm reduces execution time when increase number of tasks.

References

[1] V. Indhumathi "Improved Fault Tolerant in Workload execution Through Quality Particle Swarm Optimization for Grid Environment" International IEEE conference on Computing for sustainable Global Development, INDIACOM 2016 pp.5040 – 5045, Mar 2016.
[2] V. Indhumathi and G. M. Nasira "Service oriented Architecture for load balancing with fault tolerant in Grid computing" IEEE International conference on Advances in computer Applications, Oct 2016.
[3] V. Indhumathi and Prof. Dr. G. M. Nasira, "Resource Monitoring and Prediction with Fault Tolerance in Grid Environment through Meta Heuristics", International Journal of Applied Engineering and Research, vol 10, no 24, pp43993-44000, Dec 2015.
[4] Kapil B. Moreyl and Sachin B. Jadhav "Grid Computing for Dynamic Load Balancing", International journal of computer science and Engineering, vol 4, Jan 2016.
[5] Celaya, Javier, and Unai Arronategui, "A Task Routing Approach to Large-Scale Scheduling, " Future Generation Computer Systems, vol.29, pp. 1097-1111, 2013.
[6] Balasangameshwara Jasma and Nedunchezhian Raju, "Performance-Driven Load Balancing with Primary-Backup Approach for Computational Grids with Low Communication Cost and Replication Cost, " IEEE Transactions on Computers, vol. 62, no. 5, pp. 990-1003, 2013.
[7] Hao Y, Liu G. and Wenc N, An enhanced load balancing mechanism based on deadline control on GridSim, Future Generation Computer Systems, Vol.28, 2012, pp. 657- 665.
[8] Khan, Rafiqul Z., and Md F. Ali. "An Efficient Diffusion Load Balancing Algorithm in Distributed System." International Journal of Information Technology and Computer Science (IJITCS) 6, no. 8 (2014): 65.
[9] N.Susila, S.Chandramathi, Rohit Kishore," A Fuzzy-based Firefly Algorithm for Dynamic Load Balancing in Cloud Computing Environment, "Journal Of Emerging Technologies In Web Intelligence, Vol.6, No.4, November 2014.
[10] Colony based Multi-Objective Load Balancing Technique for Cloud Computing Environment, "International Journal of Computer Applications (0975-8887), Vol.114, No.4, March 2015.
[11] Harshit Gupta, Kalicharan Sahu,"Honey Bee Behaviour Based Load Balancing of Tasks in Cloud Computing, "International journal of Science and Research, Vol.3, Issue6, June 2014.
[12] Wei-Tao Wen, Chang-Dong Wang, De-Shen Wu, Ying-Yan Xie, "An ACO-Based Scheduling Strategy on Load Balancing in Cloud Computing Environment, "Ninth International Conference on Frontier on Computer Science and Technologyn, No.364-369, 2015.
[13] Santanu Dam, Gopa Mandal, Kousik Dasgupta, Paramartha Dutta, "An Ant Colony Based Load Balancing Strategy in Cloud Computing, "Advanced Computing, Networking and Informatics - Volume 2, Smart Innovation, Systems and Technologies 28, Springer International Publishing Switzerland 2014.
[14] Liang Bai, Yan-Li Hu, Song-Yang Lao, Wei-Mng Zhang, "Task scheduling with load Balancing using Multiple Ant Colonies Optimization In Grid Computing, " Sixth International Conference on Natural Computation ICNC, IEEE Circuits and System Society, 2010.
[15] Garima Joshi, S.K.Verma,"Load Balancing Approach in Cloud Computing using Improvised Genetic Algorithm:A Soft Computing Approach, "International Journal of Computer Applications(0975- 8887), Vol.122, No.9, July 2015.
[16] S. Prakash, D. P. Vidyarthi," Load Balancing in Computational Grid sing Genetic Algorithm," Advances in Computing Vol.1, NO.1, PP. 8-17, 2011.
[17] Jingyi Ma, "A Novel Heuristic Genetic Load Balancing Algorithm in Grid Computing" Second International Conference on Intelligent Human-Machine systems and Cybernetics, IEEE computer Society, 2010.
[18] Naglaa Reda,"An Improved Suffrage Meta-Task Scheduling Algorithm In Grid Computing Systems, " International Journal of Advanced Research, October 2015.
[19] Irfan Darmawan, Kuspriyanto, Yoga Priyan, Ian Joseph M," Integration Of Genetic And Tabu Search Algorithm Based Load Balancing For Heterogeneous Grid Computing, " International Conference on Computer, Control, Informatics and Its Applications, IEEE, 2013.

Licensed Under Creative Commons Attribution CC BY

[20] Adamuthe A.C., Thampi G.T., Bagane P.A., ”Genetic Algorithms and Tabu Search for Solving Workflow Scheduling Application in Cloud, ”ICCN, pp.216-223, Elsevier Publications, 2013.
[21] Zhang Wenpeng, Liu Hongzhao, ”A Load Balancing Method Based on Genetic Clonal Annealing Strategy in Grid Environments, ” International Conference on Educational and Network Technology, IEEE, 2010.
[22] V.Indhumathi, and G.M.Nasira, “Fault Tolerance in Job scheduling through Fault Management Framework Using SOA in Grid”, ICTACT Journal on Soft Computing, Volume 7, Issue-2, Page No. 1381 – 1385, January 2017

Licensed Under Creative Commons Attribution CC BY