A Guaranteed Service Resource Selection Framework for Computational Grids

Ganesh Kavitha and V. Sankaranarayanan

*Department of Information Technology,
B.S. Abdur Rahman University, Chennai, India*

gkavitha.78@gmail.com, sankarammu@yahoo.com

Abstract

The computational grid is a collection of resources from multiple administrative domains and selection of appropriate resources for a client application is a critical task. The process of resource selection should not be a simple matchmaking mechanism for a reliable execution environment. Therefore, appropriate resources are selected from the available grid resources that satisfy the application requirements of trust, computation capacity and network capacity and establish dynamic service level agreements to provide guaranteed services to the client. The proposed framework implements dynamic service level agreements by monitoring the job execution and extending the resource usage by predicting the future load on the resources. Simulation results show that there is a significant improvement in the success rate of the jobs submitted to the grid and considerable reduction in the SLA creation time.

Keywords: *Dynamic service level agreement, Trust, Load prediction, Network latency, Computation capacity, Quality of service*

1. Introduction

Grid computing provides mechanisms for controlled sharing of diverse resources that are distributed geographically across different organizations. A Virtual Organization (VO) is a collaborative computing environment that enables sharing of resources from multiple physical organizations in a controlled manner according to the stated sharing policies of resource providers [1, 11]. In a coordinated resource sharing grid environment, the real challenge is to deliver non trivial qualities of service to the grid users that demands complex requirements. The quality of service delivered to the client depends crucially on the selection of appropriate subset of available resources [2].

The existing grid resource management mechanisms tend to schedule the job to resources according to a resource allocation policy that satisfies user quality of service (QoS) requirements in terms of time deadline and cost using best effort approach [15, 19]. As grid is an uncertain environment and the availability of resources changes from time to time it makes the existing best effort resource selection approach to be unsatisfactory from the end user's perspective. To provide guaranteed services to the client, it is necessary to negotiate and establish a contract known as service level agreement (SLA) between the participating entities. A multi-criterion resource selection mechanism is proposed in the present work as an improvement over the existing mechanisms.

The aim of this research article is to present an integrated resource selection framework that selects appropriate resources with a concern on security, requirement satisfiability, resource availability, load and access policy restrictions. The jobs are allotted to the selected resources after establishing dynamic SLA to provide guaranteed service for grid clients. The dynamic service level agreement provides flexibility in the usage of resources for an extended duration based on forecasted future load. The SLA expires once the job is successfully completed. The different components of resource selection are evaluated using different mechanisms and integration of all the components results in a Guaranteed Service Resource Selection (GSRS) framework.

The main contributions in this research paper are (a) Development of an efficient resource selection strategy for global scheduling (b) Integrate the different components of resource selection service (c) Creation of templates for every client application and (d) Establish dynamic service level agreements between clients and resource providers.

The different resource selection strategies in computational grids are discussed in Section 2. Section 3 presents the different phases of the GSRS framework. The evaluation of the proposed GSRS strategy is presented in Section 4. Section 5 elucidates the related work in this research topic and conclusions are presented in Section 6.

2. Methods of Resource Selection

Computational grids enable sharing of resources across administrative domains in a coordinated fashion to provide computing power and deliver various qualities of service to grid clients. The QoS requirements for a client can be functional requirements as task and data dependencies and/or non-functional requirements as time deadline, budget restrictions, and guarantee on bandwidth for data transfer [15]. Further, the resource selection strategy has to decide on the optimal number of resources required for job execution as over provisioning increases the complexity of execution and minimizes the utilization of resources. Hence, the problem of resource selection is studied in three different phases and is discussed in detail in the following sub sections.

2.1. Quality of service in resource selection

The provisioning of QoS in grid computing is essential to employ grids in commercial domains apart from scientific research. The resources can join or leave the VO at any time. Therefore, the users need to know whether they are interacting with the legitimate resources so that the data and computations are protected against malicious activities.

The required level of security is provided by integrating trust as a part of resource selection and it assures the basic entry level security for the client jobs [14, 18]. The resource that satisfies all the job requirements, but not granting access will not be available for job execution. Hence, policies at the site level are important for selection of resources. The run time failures, reallocation and job migrations are avoided by evaluating the availability of computation and network resources at the time of job submission [5]. Selection of resources based on user stated QoS parameters as time and cost provides only best effort services for client jobs.

2.2. Service level agreements in resource selection

Service level agreements ensure guarantee for the service quality delivered to the user as committed by the resource provider. The SLA specifies the Service Level Objectives (SLO) that must be met by the resource provider. In grid environments, SLA establishment has to be supported at the meta-scheduler because the access restrictions on the resource usage are enforced at the meta-scheduler level. The users or the resource providers are penalized for violations against the agreements [13]. The various phases of SLA management are SLA negotiation, SLA creation, SLA monitoring and SLA violation. An agreement is created between the client and resource provider stating the service terms and guarantee terms and then the tasks are submitted for execution. The SLA is an integral part of resource selection in grid as the resource providers and users are from different domains bounded by different access and usage policies.

2.3. Policy based resource selection

The site level resource access policies and resource usage policies are stored in the policy repository of every virtual organization. At the meta-scheduler level, the site level policies are considered for allotting resources to client jobs. The access policy expresses the authentication and authorization policies of resources where as the usage policies specify amount of resources contributed to the grid. The resources that match the user requirements are checked against the site level and usage policies stated by the resource providers. The local resource policies are taken care by the local resource management systems. In the proposed framework, the policy information is considered during SLA negotiation and dynamic SLAs are established for guaranteed service.

3. Guaranteed Service Resource Selection Strategy

In the next generation computing systems, the user application increasingly rely on grids for their execution environments rather than local resources administered by its organization. There are several challenging issues that are to be addressed apart from the basic resource management functions of locating a resource and assigning jobs. These include cross domain trust, managing diverse resource policies, concurrent allocation of multiple resources, prediction of workload and commitments to provide required quality of service.

The computational grid architecture is presented as a three tier architecture as shown in Figure 1. The lower layer is the physical layer that contains a pool of resources from different administrative domains. The middle layer is the resource selection and allocation layer that provides controlled access to the appropriate resources for execution of submitted application. The higher layer is the application layer where different clients submit their jobs for execution on the grid. The proposed Guaranteed Service Resource Selection (GSRS) Strategy provides a unified framework for selecting appropriate resources by integrating the quantitative trust of a resource provider and the computation and network capabilities. The resource selection framework based on guaranteed service resource selection strategy is shown in Figure 1.

The resource pool contains resources from different domains and the trust evaluation process filters the resources that satisfy the security level required by the client. The computational resource management selects single or multiple resources for job execution based on the resource capacity available at the time of job submission. To efficiently manage the network bandwidth of the connected resources, parallel striping is employed to transfer data and client code to and from grid resources. The predicted load on selected resources helps to avoid sudden increase in the response time of the job due to local jobs. The selected resources are engaged in template exchange and SLAs are created with satisfied resources. The execution of the jobs is monitored and the resource capability can be dynamically extended by the job for successful job completion. The status of jobs is evaluated and the resource trust values are updated according to user satisfaction.

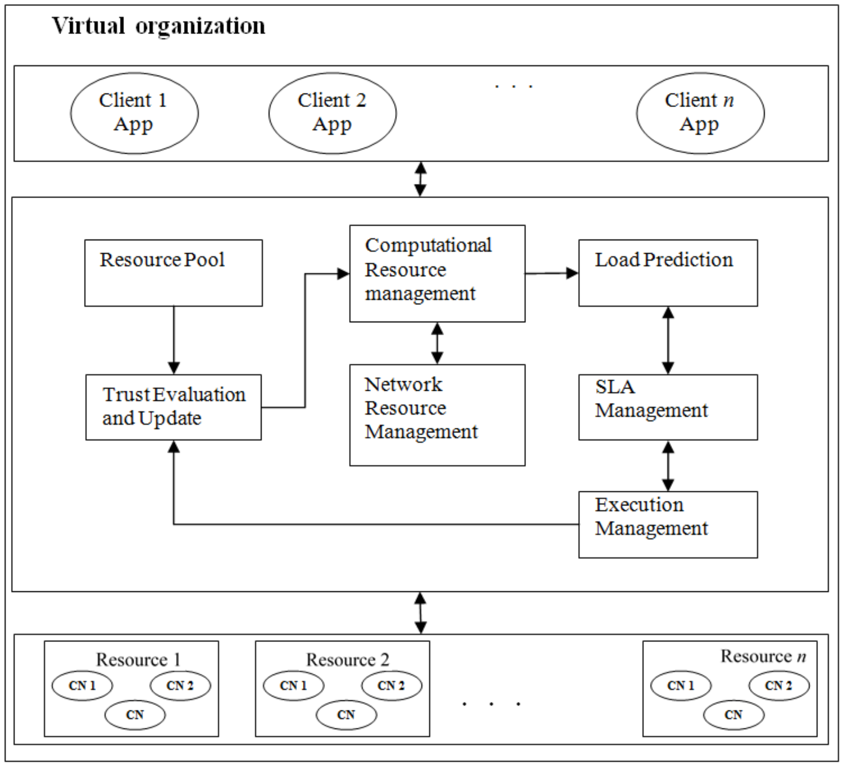

Figure 1.The proposed architecture for guaranteed service resource selection framework

The following section discusses the different components of the resource selection framework.

3.1. Trust evaluation and update

Trust has been recognized as an important factor for selection of appropriate resources in a grid. Trust is evaluated as a combination of qualitative trust and quantitative trust [14]. The quantitative execution trust about a resource is evaluated by considering the components of Subjective Trust (SBT) and Objective Trust (OBT) of a resource and both weigh equally in the calculation of Overall Trust Value (OTV). The overall trust value about a resource is computed as,

$$OTV = \alpha * SBT + \beta * OBT \quad (1)$$

where, $\alpha=\beta=0.5$ and OTV varies between 0 and 1.

The qualitative trust is expressed as a function of user satisfaction and is evaluated from the actual experiences of the user involved in the transactions with the selected resource. The User Satisfaction (US_R) about a resource is calculated as given in eqn.(2).

$$US_R = ST_j * JS_j * SSLO_R \quad (2)$$

where ST_j represents the Status of job completion, JS_j the Job Size and $SSLO_R$ represents the Satisfied Service Level Objectives (SSLO) of the resource. The SSLO is determined as the ratio of number of attained SLO to the number of committed SLO. The user satisfaction is the direct feedback about the allocated resource. Hence, the qualitative trust value is used as a factor for trust update in the Direct Trust Table (DTT) of the resource and in the Overall Trust Repository (OTR) of the VO.

3.2. Computational resource management

In a computational grid, a known strategy for efficient execution of a huge application is to partition the application into multiple independent tasks and schedule those tasks over a set of available processors. It is important to choose optimum number of resources for a job as too many resources increases the complexity of job scheduling and communication time for staging necessary input data on these resources [6].

A client application is modeled as a set of tasks and the tasks are grouped according to resource capacity. The Resource Capacity RC_{TD}, computed for the user specified Time Deadline (TD) is given as,

$$RC_{TD} = ACP_R * TD \qquad (3)$$

where, ACP_R represents Available Computation Power (ACP) of a resource R and TD the user specified time deadline. The grouped tasks known as task sets are dispatched and scheduled on optimum number of suitable resources.

3.3. Network resource management

The resources in a grid are connected through high latency wide area networks (WAN). In practice, the network bandwidth is shared by multiple entities and the speed of the link is relatively stable only for a short period of time and forecasts does not provide accurate network information [9, 20]. When communicating over a wide area network most applications choose TCP for its ease of use and reliability. The link throughput is improved through TCP striping and optimal number of stripes is opened in parallel between the communicating entities. The input data is sliced into equal sized blocks according to stripe size and transferred in parallel to selected resources. The ratio of the Required Link Capacity (RLC) and Available Link Capacity determines the Optimal Stripe Size (OPT_S) [7].

$$OPT_S = \frac{RLC}{ALC} \qquad (4)$$

The network resource management component efficiently manages the network resources according to the present environmental conditions of the grid and utilizes the available bandwidth to its maximum potential in high latency networks.

3.4. Load prediction

The resource management in computational grid combines the computational resource management and network resource management. The computational resource management allocates appropriate resources by considering the uncertainty of load conditions that prevails in grid due to local job execution and burst load from remote nodes. The pattern of load on resources varies from time to time and hence predicting the workload characteristics is of prime concern in the resource selection process. The current information of resource capacity

(RC_{TD}), makes it possible to predict the workload on a resource at the n^{th} time instant. The Available Computation Power at the forecasted n^{th} time interval (ACP_{FT}) is evaluated as,

$$ACP_{FT} = RC_{TD} - FL_{n-1} \quad (5)$$

where, FL_{n-1} is the forecasted load at the n-1^{th} time instant. The load prediction is based on the statistical properties of load on the resource [12]. The forecasted value of load X_{t+1} in exponential smoothing method is given as,

$$X_{t+1} = \omega X_t + \omega(1-\omega) X_{t-1} + \omega(1-\omega)^2 X_{t-2} + ... + \omega(1-\omega)^{t-1} X_1 \quad (6)$$

where, X_t represent the actual load values in present time, ω the weights assigned to past observed load values. The value of ω varies between 0.01to1.0 and it is highly correlated to the level of uncertainty. The state of the resource changes dynamically and hence short term forecasts offers good prediction accuracy and is useful for making appropriate scheduling decisions.

The Double Exponential Smoothing Approach (DESA) is implemented as the prediction strategy because it produces n-step ahead accurate workload prediction based on the observed load values of the past. The prediction errors are smoothed out for the present time period and the future workload for the n^{th} time period is predicted.

In the proposed GSRS framework, workload prediction plays a key role in 2 phases. In the pre-execution phase, the selection of appropriate resources through workload prediction strategy improves the acceptance rate and success rate of jobs as job failures due to uncertain load and insufficient computation power is avoided. In the second phase, the execution stage, the jobs allocated to the resources are monitored for committed SLOs and status of job completion. The jobs currently in execution, if not completed within the agreed time is allowed to extend the execution on the same resource by determining the future availability of resources for a future time period using workload prediction strategy.

3.5. SLA management

In a dynamic service environment, SLA management should be a dynamic process. The critical issue in grid environments is to select optimum number of appropriate resources and bind these resources to the client according to the service level agreements. The existing work assumes that parties know about SLA negotiation protocols and about SLA templates before entering the negotiation [15]. This assumption does not fit in a grid environment as the clients and resource providers meet each other dynamically and on-demand.

In this work, the idea of dynamic SLA is presented where the clients can extend the resource usage over a time period based on the forecasted load of a resource. To establish dynamic SLA, templates are exchanged and service negotiations are performed and agreements established. The resources can be used over an extended period through dynamic SLA in the same agreement. As every client application is different and the job requirements vary, the template is not a static structure and the template elements are decided for every application based on the application requirements.

The compute intensive jobs are executed in a computational grid and hence we have created templates suitable for these jobs. The basic structure of a template contains elements relating to domain identification, site level security policy, trust level, resource usage, response time, support for extension and negotiation protocol. The templates are exchanged and negotiations are initiated with resources that accept the template and other resources that are not willing to provide the required information are neglected. Negotiating SLA only with the selected resources improve the service quality delivered to the client and significantly reduces the time spent for negotiations between the entities.

The SLA expiration time is not specified as the usage of a resource can be extended on which the job is currently in execution. The extension of resource usage is made possible by forecasting the load on computation resource for the future n^{th} time period. The forecasted load determines the available computation power of the resource for the next time interval and provides its capabilities beyond the time deadline specified in SLA. The SLA expires and terminates when the execution is successfully completed. The only limitation of DSLA is that extension would not be possible if the resources are reserved in advance for the specified time duration.

The proposed method of dynamic SLA avoids renegotiations with the connected resource or submission of jobs to other resources for execution. The above mentioned approaches waste a lot of time in relocation of jobs and leads to performance degradation of grid jobs. Dynamic SLAs provide a promising solution to the above stated problems and provides a guaranteed service without job migrations and runtime failures that occur due to forced terminations.

3.6. Performance metrics

The performance of the proposed approach is evaluated based on the following performance metrics.

Resource Utilization: It is defined as the percentage of the utilized resource power to the available resource power for the resources present in the grid.

Job Completion Time: It is defined the total time taken by the resources for successful completion of the job and includes the job execution time, communication time and SLA creation time.

Job Success Rate: It is defined as the total number of jobs successfully completed to the total number of jobs submitted to the grid.

4. Simulation and Results

The performance of the proposed Guaranteed Service Resource Selection Strategy (GSRS) is analyzed and discussed. The simulation was based on the grid simulation toolkit GridSim Toolkit 4.0 which allows modeling and simulation of entities in grid computing systems [8]. For simulation purposes we have considered five heterogeneous resources with different characteristics such as number of Processing Elements (PE) in a machine, MIPS rating of a processing element, type of operating system, site security policy and cost of using the machine. The simulation is done for multiple client applications that submit different types and size of jobs. We consider the compute intensive jobs and a job can be divided into multiple tasks of varying size according to the resource capacity and resource availability. The information of resources used in the simulation is shown in Table 1.

Table 1. Resource information

Res ID	No.of nodes	Processing power/node (MIPS)	No.of PE /Node	Band width (Mbps)	Trust Value
R1	5	2000	1	2.0	0.7
R2	8	1200	1	1.5	0.6
R3	5	1000	1	1.0	0.5
R4	6	1500	1	1.5	0.8
R5	5	1200	1	1.2	0.5

In the Guaranteed Service Resource Selection (GSRS) strategy, the resources are allotted to the client jobs in the first come first serve (FCFS) basis. The client submits the job stating the requirements of security, time limit and cost limit. The GSRS approach selects single or multiple resources from the VO that satisfies trust as an entry level security, site security policy, available resource capability and time deadline. In the present work, we have simulated the widely employed GridWay Meta-scheduler [20] and compared the performance of GSRS strategy with the GridWay meta-scheduler. The length of the submitted job varies from 50000MI to 200000MI and the size of the input data to be transferred varies from 1MB to 4MB.The time deadline specified for the client application is 5 seconds. The client applications are submitted to appropriate resources according to GSRS strategy and the execution time and communication time obtained for the jobs are shown in Figure 2.

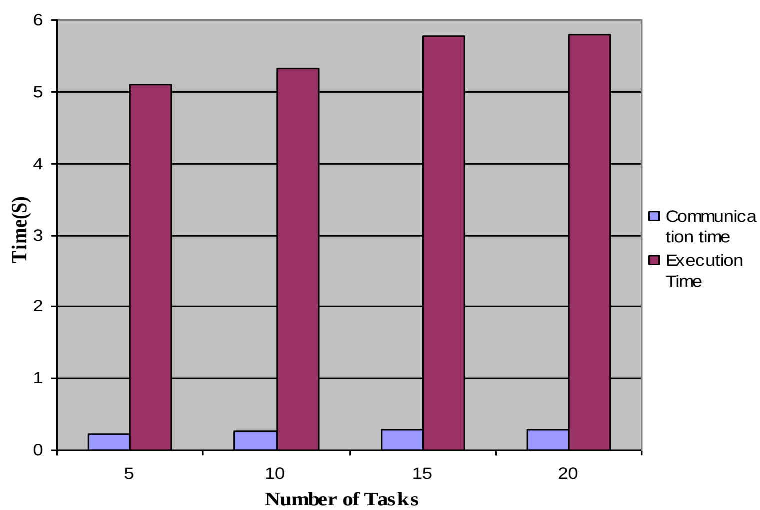

Figure 2. Job execution time

From the above, it is evident that the communication time is less than 10% of the job execution time as the data is transferred in multiple stripes to and from the remote resources. In the existing approaches, the communication time increases linearly with respect to the number of resources employed for job execution. But, this drawback is overcome in the GSRS strategy as it employs parallel striping mechanism. The GSRS strategy performs divisible job scheduling and executes the task set in parallel to achieve optimal schedules.

Establishing SLAs between resource providers and clients provide reliable services to requesting clients. The SLA creation time for different client jobs is shown in Figure 3. The client job executed on more number of resources requires the SLA to be created on all the resources. But, in the GSRS strategy, optimal number of resources are selected based on the resource availability and SLAs are established only with these selected resource providers. Therefore, the time for SLA creation has been reduced to 65% compared to GridWay which in turn reduces the completion time of jobs.

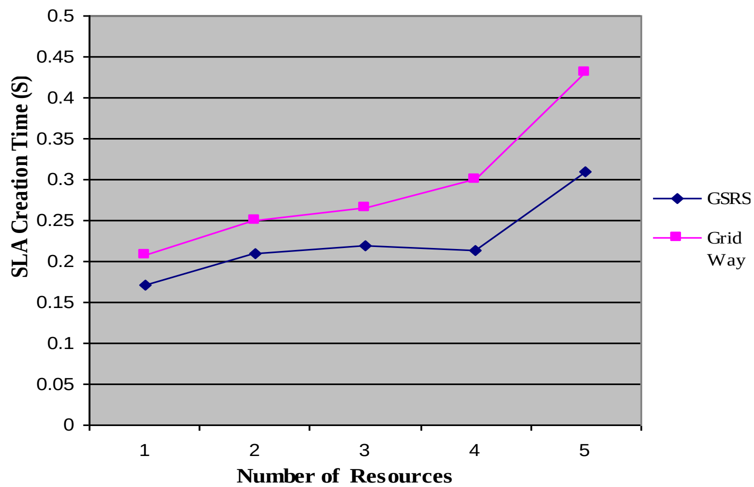

Figure 3. SLA creation time

The total completion time of jobs submitted to grid using GSRS strategy and GridWay is depicted in Figure 4. The user specified a time limit of 5s and the proposed GSRS strategy completes the job within 6s and the maximum extension of resource usage availed is only about 10 to 15% from the stated time limit. The SLA creation time, communication time and execution time are reduced to about 40-60% in the proposed method compared to GridWay.

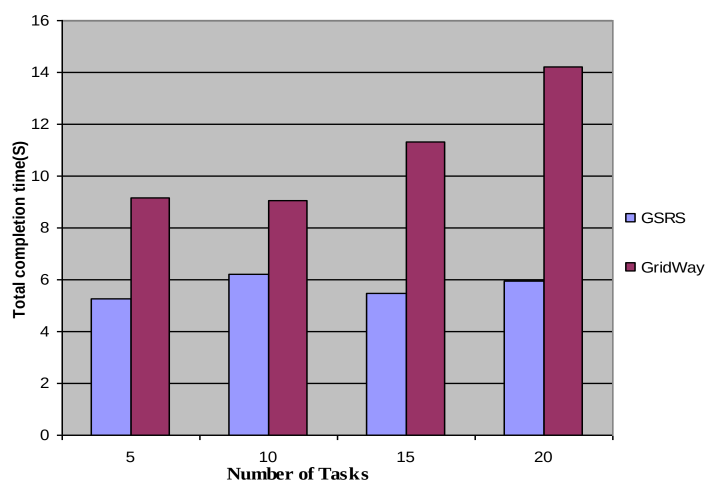

Figure 4. Job completion time

The reduced completion time is due to the load prediction strategy which considers local and remote jobs and predicts the available computation power accurately. The dynamic SLAs avoid renegotiations and creation of new SLAs to other resources for the jobs currently in execution. The jobs are completed with a small increase in the time deadline given by the user. This increased completion time does not affect the performance of either the resource or the job. But, it improves the reliability of the resources as abrupt job terminations are avoided.

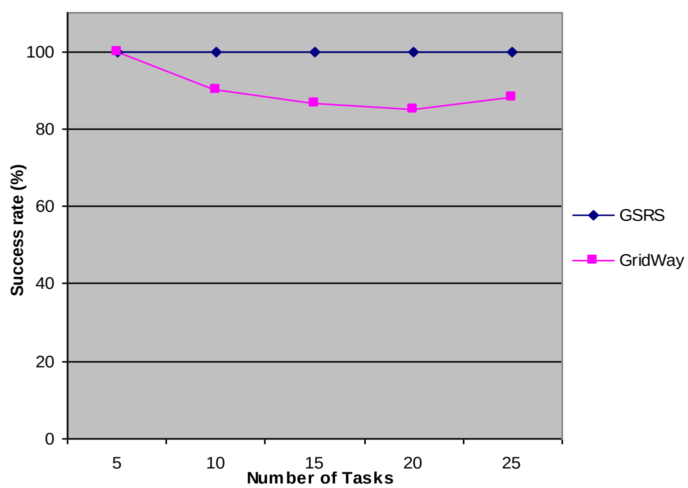

Figure 5. Job success rate

The success rate obtained for the submitted grid jobs is shown in Figure 5. It is evident that the proposed GSRS strategy performs well compared to Gridway and achieves about 20% higher success rate than Gridway. The proposed GSRS strategy is very reliable as more focus is on selection of resources for client application. The selected resources are allocated to client jobs after establishing dynamic SLAs and avoid forced termination due to insufficient resource capabilities. The GSRS algorithm is evaluated by executing multiple applications and the achieved success rate is 100%. The job is put in a pending state if suitable resources are not found for its execution but the probability of this state is less than 5%. The job acceptance rate is about 95% for the proposed algorithm.

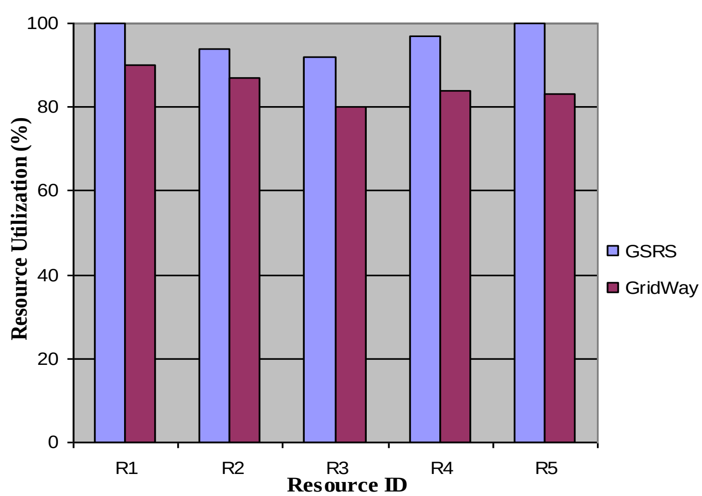

Figure 6. Resource utilization

The utilization of computational grid resources using the GSRS strategy is depicted in figure 6. The proposed method allocates task sets to resources based on their present availability and forecasted load. The GSRS approach achieves a high resource utilization of about 95% for resources that are present in the grid. The utilization of R1 and R4 are very high as these resources have highest capacity with more number of fast CPUs, high trust value and high bandwidth. The jobs are allotted and executed on the resources based on the capacity of the resources that best matches the client requirements and maximize the utilization of grid resources.

5. Related work

The traditional resource management techniques fail to efficiently utilize the computing resources in the grid environment. The quality of service depends crucially on the selection of appropriate subset of resources. This section reports the work being developed in this area. Youngjoo Han and Chen-Hyun proposed a resource aware policy administrator that makes optimal resource allocation policy satisfying the user QoS requirements [2]. The admission control algorithm discussed is based on the information of the resource queue, deadline and profit. The uncertain load conditions and the network status information that decides the job completion time are not considered in RAPA for allocation policy. Our work concentrates on the resource dynamism and provides a flexible resource allocation strategy by implementing dynamic service level agreements.

In [4], Ana Carolina Barbosa et al have presented different architectures to perform SLA auditing both qualitatively and quantitatively. The architecture has a inspector to forward the user request and calculate the service level indicator for the service provider. This method adds up to the performance loss if inspector is introduced at a remote site far away from the user. To overcome this challenge, we have implemented a trust model that evaluates trust between participating entities that aids in qualitative SLA auditing and communicates without the need for a inspector.

Linlin Wu and Rajkumar Buyya have presented a comprehensive survey of SLA creation, Management and its use in utility computing environment [17]. In SLA oriented Grid projects, SLAs are renegotiated for supporting dynamic changes in job requirements and thereby require change in the capability of resources. The limitation is that it does not support run time negotiation to adapt to dynamic operational and environmental changes. In our work we have implemented a flexible SLA to avoid renegotiation and it enables the user to extend the resource usage according to the dynamic availability of resources.

Catalin L. Dumitrescu and Ian Foster have described an architecture and toolkit for enforcing resource usage service level agreements at VO level [16]. The GRUBER engine decides the best resource sites in a VO based on the number of waiting jobs and available CPUs. But, the queue length is not very effective in determining the resource capacity. In the present work, site level policies are considered prior to job scheduling and scheduling the jobs to resources are taken care by local resource management systems. In addition, to determine the resource capacity, the proposed work forecasts the load and evaluates the available computation capacity and thus selects appropriate resources for job submission.

The work proposed by P.Balakrishnan and S.Thamarai Selvi in [13] is improved further by establishing dynamic service level agreements between the client and resource provider. The resources selected for negotiation are based on ordering of available resources based on the deviation value of QoS parameters. Negotiating the terms with all the available resources increases the SLA creation time but in GSRS strategy, templates are exchanged and the resource providers that accept the template are considered for SLA negotiation. This approach minimizes the SLA creation time as only optimal number of resources are considered for establishing dynamic SLAs.

6. Conclusion

Resource management is a central part of computational grids. The grid to be employed in commercial domains should provision suitable resources with performance guarantee to execute critical jobs that demands different performance levels. We have described a resource selection framework that integrates multiple components for incorporating security, user QoS, TCP striping, load forecasting and dynamic service level agreements. This unified framework is implemented in the resource selection and allocation layer of a virtual organization to select and allocate appropriate resources to user jobs. The proposed framework minimizes the SLA creation time and communication time and thereby reduces the total completion time for the jobs submitted to grid. The GSRS strategy achieves a very high success rate as dynamic SLAs allow the jobs to complete by extending the usage of allotted resources and avoids job migrations and forceful job terminations. Trust employed at the initial phases of job submission minimizes the chances of malicious attacks and runtime failures. The GSRS strategy maximizes the utilization of the resources present in the Grid and provides a high level of satisfaction for the users of the Grid.

References

[1] I. Foster, C. Kesselman and S. Tueke, "The anatomy of the grid: Enabling scalable virtual organizations", International Journal of Supercomputing Applications, vol. 15, no. 3, **(2001)**.
[2] Y. Han and C. -H. Youn, "A new grid resource management mechanism with resource-aware policy administrator for SLA-constrained applications", Future Generation Computer Systems, vol. 25, **(2009)**, pp. 768-778.
[3] R. Aron and I. Chana, "Formal QoS Policy Based Grid Resource Provisioning Framework", Journal of Grid Computing, vol. 10, **(2012)**, pp. 249-264.
[4] A. C. Barbosa, J. Sauve, W. Cirne and M. Carelli, "Evaluating architectures for independently auditing service level agreements", Future Generation Computer Systems, vol. 22, **(2006)**, pp. 721-731.
[5] G. Kavitha and V. Sankaranarayanan, "Resource Selection in Computational Grid Based on User QoS and Trust", International Journal of Computer Science and Network Security, vol. 11, no. 3, **(2011)**, pp. 214-221.
[6] G. Kavitha and V. Sankaranarayanan, "Secure Selection of Multiple Resources Based on Virtual Private Network for Computational Grid", Journal of Computer Science, vol. 7, no. 12, **(2011)**, pp. 1881-1887.
[7] G. Kavitha and V. Sankaranarayanan, "A Novel Resource Selection Framework to Improve QoS in Computational Grid", (Accepted for publication in International Journal of Computational Science and Engineering), **(2012)**.
[8] R. Buyya and M. Murshed, "GridSim: A Toolkit for the Modelling and Simulation of Distributed Resource Management and Scheduling for Grid Computing", Journal of Concurrency and Computation: Practice and Experience, **(2002)**, pp. 1175-1220.
[9] R. Wolski, "Experiences with Predicting Resource Performance On-line in Computational Grid Settings", ACM SIGMETRICS Performance Evaluation Review, vol. 30, no. 4, **(2003)**, pp. 41-49.
[10] E. Huedo, R. S. Montero and I. M. Llorente, "Evaluating the reliability of computational grids from the end user's point of view", Journal of Systems Architecture, vol. 52, no.12, **(2006)**, pp. 727-736.
[11] I. Foster and C. Kesselman, "The Grid: Blueprint for a New Computing", 2nd Edition, Elseiver series in Grid Computing, Morgan Kaufmann, **(2003)**.
[12] T. M. J. A. Cooray, "Applied Time Series Analysis and Forecasting", Narosa publishing House Pvt. Ltd., **(2008).**
[13] P. Balakrishnan, S. T. Selvi and G. R. Britto, "Service Level Agreement based Grid Scheduling", In: Proceedings of IEEE International Conference on Web Services, **(2008)**, pp. 203-210.
[14] G. Kavitha and V. Sankaranarayanan, "Secure Resource Selection in Computational Grid based on Quantitative Execution Trust", In: Proceedings Of International Conference on Information, Communication, Electrical and Computer Engineering, Singapore, **(2010)**, pp. 149-155.

[15] I. Brandic, S. Venogopal, M. Mattess and R. Buyya, "Towards a Meta-Negotiation Architecture for SLA-Aware Grid Services", In: Proceedings of the HiPC 2008 International Workshop on Service Oriented Engineering and Optimization, **(2008),** Bangalore.
[16] C. L. Dumitrescu and I. Foster, "GRUBER: A Grid Resource Usage SLA Broker", LNCS 3648 Springer-Verlag Berlin Heidelberg, **(2005)**, pp. 465-474.
[17] L. Wu and R. Buyya, "Service Level Agreements in Utility Computing Systems", Technical report CLOUDS-TR-2010-5, Cloud computing and Distributed Systems Laboratory, The University of Melbourne, Australia, **(2010)**.
[18] C. Lin, V. Varadharajan, Y. Wang and V. Pruthi, "Enhancing Grid security with trust management", In Proceedings Of IEEE International Conference on Services Computing, **(2004)**, pp. 303-310.
[19] R. Buyya, D. Abramson and J. Giddy, "Nimrod/G: An Architecture for Resource management and Scheduling System in a Global Computational Grid", In Proceedings of 4th International Conference on High Performance Computing, China, **(2000)**, pp. 283-289.
[20] L. Marchal, P. V. -B. Primet, Y. Robert and J. Zeng, "Optimal Bandwidth Sharing in Grid Environments", In Proceedings of 15th IEEE International Symposium on High Performance Distributed Computing, Paris, **(2006)**, pp. 144-155.

Authors

G. Kavitha received her B.E. in Electronics & Communication Engineering in 2000, M.E. in Computer Science & Engineering in 2002 from Madras University. She is pursuing Ph.D in Computer Science & Engineering, Anna University, Chennai. She has a teaching experience of 10 years. She is presently working as an Assistant Professor in Information Technology Department at B.S.Abdur Rahman University, Chennai,India. Her research interest includes grid computing, cloud computing and service oriented architectures.

Dr. V. Sankaranarayanan received his B.E. in Electical Engineering from Annamalai University in 1966, M.Sc. (Engg) in High Voltage from college of Engineering, Guindy, Chennai and Ph.D. in computer applications from IIT (Madras) in 1979. He worked as professor in Anna University for about 3 decades. He is currently Director at B.S.Abdur Rahman University Chennai, India. His research interests include MANETs, grid and cloud computing.