Journal of Computer Science 7 (12): 1881-1887, 2011
ISSN 1549-3636

Secure Selection of Multiple Resources Based on Virtual Private Network for Computational Grids

G. Kavitha and V. Sankaranarayanan
Department of Information Technology,
B.S.A. Crescent Engineering College, Chennai 600 048

Abstract: Problem statement: Grid computing provides a virtual framework for controlled sharing of resources across institutional boundaries. In computational grids, a client application is executed on the available set of resources that satisfy the user QoS requirements. Some applications require exhaustive computation power for execution of its tasks. In general, these tasks are assigned to a single available resource on the grid that has the required computation power. Therefore, the client application waits indefinitely until a suitable resource is found. **Approach:** In this study a novel multiple resource selection strategy is presented, which selects multiple resources based on trust and QoS parameters and the tasks are mapped to the appropriate resources for parallel execution. Selection of resources is based on the trust value of the resource, the available computation power at the time of job submission, the speed of the connectivity link, the time deadline and the budget constraints. The proposed method performs task grouping and selects the optimum number of resources for task execution. The tasks are executed in parallel among the multiple resources and the results are aggregated and transferred to the client within the specified time deadline. Security for the user tasks is strengthened by creating a Virtual Private Network (VPN) to the selected resources and tasks are further mapped to the resources through the secured VPN channel. **Results:** Simulations results show that there is a significant improvement in the overall resource utilization of the grid with a high success rate of jobs and reduction in the total execution time of submitted jobs. **Conclusion:** The tasks are scheduled to available multiple resources with VPN security. As optimum number of resources is selected for parallel execution, the resources are utilized to a maximum and there is a reduction in the percentage of pending jobs on the grid.

Key words: Task group, resource utilization, Task Manager (TM), Virtual Organization (VO), Required Computation Power (RCP), Task Sets (TS), Available Computation Power (ACP)

INTRODUCTION

The Grid is defined as a framework for flexible, secure, coordinated resource sharing among dynamic collection of individuals, institutions and resources (Foster and Tueke, 2001). It allows researchers in different administrative domains to use multiple resources for problem solving and provides an infrastructure for developing larger and more complex applications. A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high end computational capabilities (Foster and Kesselman, 2004). In such a heterogeneous grid environment, the major challenges to be addressed are the selection of appropriate resources for the user application and security for the user's job executed on the grid. Large scale grid applications are executed in parallel among multiple remote resource sites that demands a high security for the user's application and the participating resource. Selection of resources based on multi criteria is discussed by (Jamaludin and Ishak, 2011).

In a computational grid, achieving fair and secure resource allocation among multiple users is an important issue to be addressed. The classification of security aware grid models is discussed by Humphrey and Thompson (2002). An optimal strategy that maps the user's task to machines based on a non cooperative symmetric game for modeling the user behavior is presented by Kolodziej and Xhafa (2011). Each user tries to choose the machines in order to minimize the total cost of scheduling. In a multi-user grid environment, the user may behave in a selfish manner by holding too many resources for a very long time and puts the other user jobs in the wait state. The number of QoS qualified hosts for scheduling the user's job is

Corresponding Author: G. Kavitha, Department of Information Technology, B.S.A. Crescent Engineering College, Chennai

chosen as required by the size and complexity of the job (Hsu and Chen, 2010). To reduce the number of resource that participate in job execution the tasks are re-dispatched to lightly loaded resources without concern of security and job migration time.

Allocation of multiple types of resources simultaneously for a certain period of time for every request is difficult. If there are not enough resources available when a user request occur, the resource allocation will be delayed until the required capacity of the resource is available (Kuribayashi, 2011). In the existing methods of resource allocation, the resources are assigned based on the number of jobs that are waiting in the queue. Some proposed methods select the resources based on the processing power or the data access cost. A multiple resource selection strategy is discussed in the present work considering the processing power of the resource, time deadline of the user and the security of the user's job.

Security of user's job is essentially important as security of the resources. The security for the client's task is built in two levels: In the first level, the resource that satisfies the required trust level for the application is selected for job execution. The trust about a resource is calculated by combining the subjective and objective trust values (Kavitha and Sankaranarayanan, 2010). The security is further strengthened in the second level by building a secure VPN channel to the selected resources. The task sets are sent over the VPN channels to the appropriate resources for execution. Therefore, the client tasks are protected against intrusive network attacks, code modification and any other malicious activity.

The number of resources to which the tasks are assigned should be decided based on the size and complexity of the tasks and the available computation power of the resources. Therefore, the tasks are grouped into appropriate task sets and executed in parallel among the multiple resources that are available in the Virtual Organization (VO).

MATERIALS AND METHODS

Multiple resource selection in computational grid: A client application is modeled as parallel independent tasks and scheduled to multiple resources available in a VO. The key issues remain in finding the number of resources to be selected for a client application and providing security for the tasks that are mapped to the selected resources. The above mentioned issues are addressed in the present work. The number of resources required for an application is based on the available computation power of every trusted resource and the time deadline specified for the completion of the job. By mapping the parallel tasks to more than one resource, the execution time of the job is reduced and the available resources are utilized efficiently. There is an improvement in the success rate of the submitted jobs as the tasks are mapped to the resources only through VPN channels.

Security of client tasks based on VPN: The resources hold the complete control of the tasks submitted to it. The information contained in the tasks has to be highly secured and the integrity of the data has to be preserved. As the tasks are executed at a remote site that belongs to different organizations, security is a vital concern. To have a secure application execution environment, it is proposed to have a VPN gateway on all the resource sites of the VO and also at the client side. There is a VPN server at the middle layer that connects the resources and clients through VPN gateway. The resources and the client tasks securely communicate by establishing a VPN tunnel between the client and the VPN server port. The tasks of the application are grouped into different task set and appropriate resource for every task set is determined. The task set is then transferred to the selected resource through the VPN tunnel established between resource and VPN server. The VPN connection to the resource is released as the task set is successfully executed and the results returned to the result aggregator.

To enforce security policy, all the VPN gateways work together. The entities are to be authenticated by the VPN server before the tunnel is established. Authentication is performed by the exchange of certificates during the connection establishment phase. For connecting 'N' resources with the VPN server, N tunnels are required. The connection pattern is a star topology which also minimizes the cost of VPN tunnel construction.

The star topology of VPN configuration is implemented in our work and shown in Fig. 1. When a resource Ri has completed the execution then the VPN connection to that resource is released and that resource is utilized for other applications. The advantage is that the application does not hold all the selected resources until the application completes its execution. Every selected resource is connected to the VPN server until the task set assigned to it is completed and after returning the results the connection to the resource is released. By using the star topology the resources within the VO are effectively utilized and the idle time of the resources are reduced.

Security assured multiple resource selection in grids: In a collaborative grid environment the highly CPU intensive user's tasks may demand the computation power that cannot be provided by a single resource. Hence, the client's task will wait indefinitely until a suitable resource is available. To overcome the above problem, an algorithm for selection of multiple resources within a VO is proposed with a prime concern for user security. The whole system is based on the dynamic state information of the resources obtained during runtime.

The framework for secure multiple resource selection is shown in Fig. 2. It is three tier architecture with the participating resources at the physical layer. The resources in a VO differ in terms of CPU power, storage capacity, operating system and network connectivity. A resource is a heterogeneous cluster that has multiple compute nodes and performs compute intensive tasks. Every client application is modeled as independent tasks that are present in the application layer.

The middle tier is the resource allocation layer that performs the selection of appropriate number of resources for the user application and dispatches the tasks to these selected resources for computation. The resource pool contains the list of resources registered in the VO. From the resource pool the trust agent selects the list of trustworthy resources for the client application depending on the security level given by the user. The trust evaluation and trust update procedure is presented (Kavitha and Sankaranarayanan, 2010).

The resource allocation layer contains a Task Manager (TM) that performs grouping of the tasks into Task Sets (TS) depending on the size and complexity of the client job. Every task set is then mapped to a multi cluster resource for further execution. The task set is transferred to the selected resources through VPN tunnel. The VPN connection exists until the task set execution is complete and then the resource is released. The result aggregator collects the results from multiple resources and sends it to the client by the secure VPN tunnel established between the client and VPN server. Hence, the tasks are submitted to multiple trustworthy resources that has the required computation capacity for task execution.

Selection of optimum number of resources: In a shared distributed environment, the efficient utilization of the available resources improves the performance of grid system as a whole. When the tasks of the job are allotted to multiple resources, there is a reduction in the execution time of the jobs but there is a necessity to find the optimum number of resources on which the tasks are to be assigned such that it provides a fair allocation of resources to the clients. In the proposed method, the resources are allotted such that the resource capacity is utilized to its maximum for the specified duration of time.

The client application tasks are grouped into task sets. For the given 'n' tasks of the application, sort the tasks in descending order of Required Computation Power (RCP). The list of trustworthy resources is sorted in decreasing order of Available Computation Power (ACP). In order to utilize every resource to its maximum, find the number of tasks that can be grouped into a Task Set (TS) and assign the tasks set to the trustworthy resource in the list. Every task set 'k' has 'm' independent tasks that are grouped according to the RCP of the tasks and the resource capacity Eq. 1:

$$TSk = \sum_{j=1}^{m} RCP_j \quad (1)$$

Where:

$$\sum_{j=1}^{m} RCP_j \le RC_{TD} \text{ and } m \le n$$

Hence, the total computation power of 'm' tasks should be within the resource capacity. For every job, appropriate number of resources is allotted and the trustworthy resources are effectively used to its maximum capacity. The Resource Capacity RC_{TD}, computed for the user specified Time Deadline (TD) is given as Eq. 2:

$$RC_{TD} = ACP_R * TD \quad (2)$$

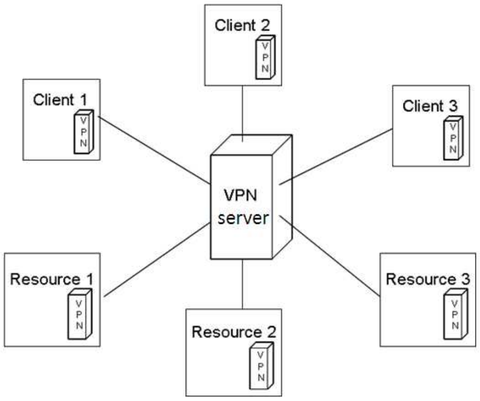

Fig. 1: VPN connection set up

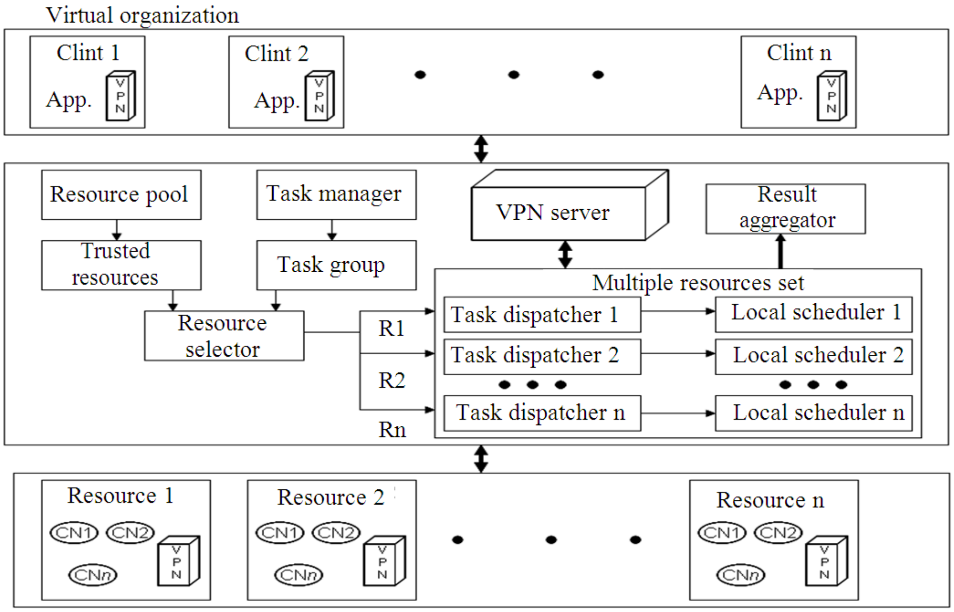

Fig. 2: Secure multiple resource selection framework

After the tasks are grouped for a selected resource, it is removed from the list and assigned for execution on that resource. Similarly, assign the tasks to the next available resources in the list until all the tasks are scheduled for execution. The number of Task Set (TS) for the given application is based on the available set of resources and the size and complexity of the user's task. If the available resources do not satisfy the computation power requirement for the job it is identified in the initial phase and the job is put in pending state.

Trustworthy Multiple Resource Selection (TMRS) strategy: The multiple resource selection strategy considers the trustworthiness of the resources, the available computation capacity of the resource and the time deadline specified by the user. The trustworthiness of the resources is evaluated based on subjective and objective parameters (Kavitha and Sankaranarayanan, 2010) as past history of transactions and present environment conditions of the resource.

The parameters of the client application are Job ID, required trust level, time deadline and cost constraints. The entry level security for the client job is provided by the computed quantitative trust value about the resource. The Overall Trust Value (OTV) is given Eq. 3:

$$OTV = \alpha * SBT + \beta * OBT \qquad (3)$$

The values of the Subjective Trust (SBT) and the Objective Trust (OBT) are weighed equally to obtain the quantitative trust value. The weights α and β are assigned a value of 0.5 and OTV varies between 0 and

1. After selecting a trustworthy resource from Eq. 3, determine the Available Computation Power (ACP) of the selected resource. The computation power is measured as the aggregate CPU speed across all computing nodes with in a resource Ri. Based on this current information, sort the resources according to the available computation power. The ACP of a resource Ri is the difference between the Overall Computation Power (OVCP) and the Occupied Computation Power (OCP) of the resource Eq. 4 (Kavitha and Sankaranarayanan, 2011):

$$ACP_R = OVCP_R - OCP_R \qquad (4)$$

From the obtained information of computation power, the total number of resources required in parallel to execute the client job is found. In addition to computation power, it is also necessary to consider the speed of the network bandwidth and the distance of the selected resource. The transfer time of the jobs would be shorter if connected to a high speed link and goes through minimum number of network switches.

Every client job after grouping into task set is assigned to selected multiple resources in parallel. The Expected Completion Time (ECT_{TSK}) of every Task set 'k' on a resource is derived as the sum of VPN Connection Set up Time (CST), Time to Compute (TTC_{TSK}) and Job Transfer Time (JTT_{TSK}) Eq. 5:

$$ECT_{TSk} = TTC_{TSk} + JTT_{TSk} + CST_R \qquad (5)$$

The Time to compute for every Task Set (TS) is different and it depends on the computation power required by the various ‘m’ independent tasks of the TS and the computation power available at the selected resource Ri and is expressed as Eq. 6:

$$TTC_{TSk} = \frac{\sum_{j=1}^{m} RCP_j}{ACP_R} \qquad (6)$$

The Task set are transferred to the selected resources for further computation through the VPN tunnels. The Job Transfer Time depends on the speed of the bandwidth link and the size of the job to be transferred Eq. 7:

$$JTT_{TSk} = \frac{\text{Filesize}}{\text{linkcapacity}} \qquad (7)$$

The task sets are executed in parallel on the selected trusted resources and after completion the results of execution has to be sent to the result aggregator that combines the results of the individual task sets and transfers it to the client. After Job completion, the Total Execution Time (TET) for the client application has to be found to evaluate the satisfaction of the user Eq. 8:

$$TET = FT - ST \qquad (8)$$

where, FT is the finish time of the task that completed its execution at the last on any resource and ST is the start time of the task that started first on the resource.

TMRS algorithm:

Input: Resources Ri, minimum trust level, time limit, cost limit, job Ji:

```
1. For every resource Ri
        a. calculate Overall Trust Value (OTV) given the
        alues of Subjective Trust (SBT) ,Objective
        Trust(OBT) , weights α and β.
        b. for each client’s job request, the Minimum
        Trust Level (MTL) required for the job is given
        by the client.
    if (OTV >= MTL )
    select the resources and add it in the list L
2. For every resource Ri in L, the following values are ascertained based on the current conditions of grid
    a. calculate the Available Computation Power
        (ACP) with the current values of Overall
        Computation Power (OVCP) and Occupied
        Computation Power (OCP).
    b. Rank the resources based on the available
    computation power and add it in Ranked List.
3. Given the job Ji, determine the set of tasks to be
        grouped based on the RCP of the tasks and $RC_{TD}$
        If ($RCP_m$<=$RC_{TD}$)
        Select the resource and assign the task set
        (TS) to that resource.
 Else
        Put the job in pending state.
4. Assign m tasks to Ri. Pending Tasks to be allotted is
 n = n-m
5. Select the next available resource in L and assign the maximum number of tasks for execution
6. Repeat steps 3-5 until n = 0
7. Calculate the Total Execution Time (TET) for the job Ji after executed on multiple resources and evaluate the user satisfaction and resource utilization
```

Performance metrics: In the present work, the following performance metrics are examined.

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 success rate: It is defined as ratio of the number of jobs successfully completed to the total number of jobs submitted to the grid.

Total execution time: It is defined as the total time taken by the resources to complete the tasks in parallel after the tasks are allotted to it.

RESULTS

The performance of the proposed TMRS algorithm is analyzed. The simulation was based on the grid simulation toolkit GridSim Toolkit 4.0 which allows modeling and simulation of entities in grid computing systems (Buyya et al., 2002). 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 and cost of using the machine. The simulation is done for the client job that consists of multiple tasks of varying size and the performance metrics are evaluated. The simulation set up is shown in Table 1.

In Multiple Resource Selection scheme, the resources are allocated in the First Come First Serve (FCFS) basis and the resources are released only after completion of the present Task Set (TS) of the user. We illustrate the resource selection process using the proposed method with five resource sites and fifty jobs.

Table 1: Resource information

Resource ID	No. of nodes	Processing power/node (MIPS)	No. of PE/node	BW (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	1.5	0.5

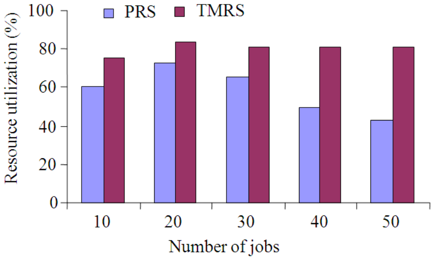

Fig. 3: Overall resource utilization

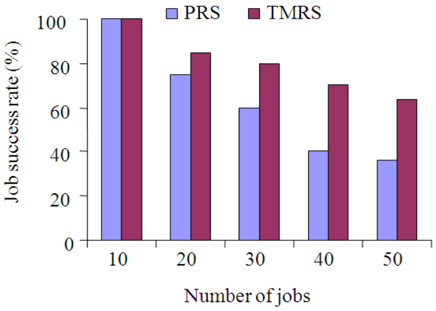

Fig. 4: Success rate of jobs in grid

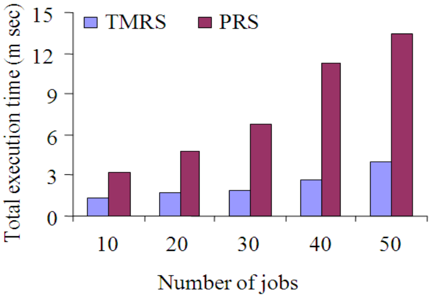

Fig. 5: Total execution time of submitted jobs

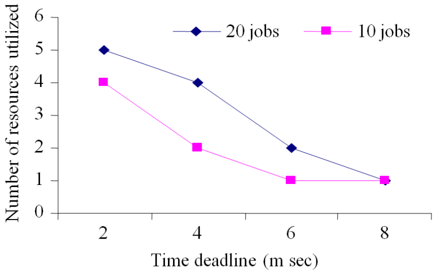

Fig. 6: Effect of time deadline

The jobs are characterized by the demands of compute power, trust level for security and the time deadline. The Information of resources used in the simulation is shown in Table 1. The length of the jobs varies from 2000-50000MIPS. The same set of jobs is submitted to the Power Based Resource Selection model (PRS) and proposed Trustworthy Multiple Resource Selection model (TMRS) and their performance is analyzed.

The overall resource utilization in the grid for a time deadline of 2 ms is depicted in Fig. 3.The available resources are utilized to about 81%. The success rate of the proposed method is always high than the success rate of the PRS method as shown in Fig.4. The total execution time is determined for the jobs that are successfully executed on the grid. It is evident from Fig. 5 that the execution time for the jobs varies drastically when jobs are submitted to multiple resources. The total execution time depends on the time deadline given by the user. The simulated result is for the different number of jobs between 10 and 50 and the user time deadline of 2 ms.

The effect of time deadline on the number of resources utilized is shown in Fig.6. If the time deadline is increased there is a decrease in the number of resources utilized for the job execution. The results of simulation are shown for 20 and 10 jobs submitted to the grid.

DISCUSSION

The result of the proposed TMRS algorithm is briefly discussed in this study. The maximum resource utilization for 50 jobs is about 81% as there are resources that don’t satisfy the minimum power required for execution of task set. But compared to the power based resource selection strategy, the proposed method provides improvement in resource utilization of about 38%.

The jobs submitted to the grid achieve a high success rate if the jobs are assigned to multiple resources according to the TMRS strategy. In a PRS based resource selection strategy, jobs are allocated to a resource only if it satisfies the power requirement.

But in the proposed method, a job is split into multiple task sets and assigned accordingly to multiple resources if the computation power required for the job cannot accommodated within one resource. Because in many cases the required power for the job is more than the available power and to hence to meet the requirement the jobs are allotted to multiple trustworthy resources. Hence the pending state of the jobs is reduced and there is a significant improvement of about 45% in the success rate of the submitted jobs.

The execution time remains almost within the user specified time deadline even if the number of jobs is increased. Hence, the algorithm proves a satisfactory performance. A deviation of about 20% is acceptable as task grouping is based on the RCP of tasks and the resource capacity. The number of resources selected for parallel execution depends on the resource capacity and user specified time deadline. Therefore, more number of resources are required if the job has to be completed within shorter time.

In the proposed strategy, the jobs are assigned to resources such that they are completed within the user time deadline. Hence, if single resource doesn't satisfy the power and security requirement, the jobs are grouped into multiple task sets and executed on multiple machines with security consideration.

From the above results presented , it is evident that jobs submitted to multiple resources and executed in parallel among those resources improves the success rate of the jobs and resource utilization in the grid. The proposed algorithm provides a high degree of satisfaction for the user and maximum resource utilization in the Grid.

CONCLUSION

Resource management is a central part of computational grids. We have described a trustworthy multiple resource selection framework that aids resource selection and allocation in a dynamic grid environment. The multiple resource selection strategy executes the client jobs in parallel among the available multiple resources with a reduced total execution time for the submitted jobs. The present work provides an efficient resource selection methodology with security as a prime concern. The resource selection mechanism uses trust value as a basic parameter and has a secure channel to the selected resources for enhancing the security of user's job. The simulation results clearly indicate that the overall performance increases when the appropriate number of resources is assigned to user jobs. The task grouping strategy introduced in our work maximizes the utilization of the resources available in the grid. It is evident that the proposed algorithm improves the success rate of the jobs and hence provides a high level of user satisfaction.

REFERENCES

Foster, I., and S. Tueke, 2001. The anatomy of the grid: Enabling scalable virtual organizations. Int. J. High Perfor. Comput. Appli. Fall, 15: 200-222 DOI: 10.1177/109434200101500302

Foster, I. and C. Kesselman, 2004. The Grid: Blueprint for a New Computing Infrastructure. 2nd Edn., Amsterdam ; Boston : Morgan Kaufmann. London, ISBN: 9781558609334 1558609334, pp: 748

Humphrey, M. and M.R. Thompson, 2002. Security implications of typical grid computing usage scenarios. Cluster Comput., 5: 257-264. DOI: 10.1023/A:1015621120332

Kavitha, G. and V. Sankaranarayanan, 2010. Secure resource selection in computational grid based on quantitative execution trust. World Acad. Sci. Eng. Technol., 72: 149- 155,

Kolodziej, J. and X. Fatos, 2011. Meeting security and user behaviour requirements in grid scheduling, Simulation Modelling Practice Theory, 19: 213-226. DOI:10.1016/j.simpat.2010.06.007

Hsu, C.H. and T.L. Chen, 2010. Adaptive scheduling based on quality of service in heterogeneous environments. Proceedings of the 4th International Conference on Multimedia and Ubiquitous Engineering (MUE), 11-13 Aug, IEEE Xploor, Cebu, pp: 1-6. DOI: 10.1109/MUE.2010.5575084

Kuribayashi, S.I., 2011. Optimal joint multiple resource allocation method for cloud computing environments. Int. J. Res. Rev. Comput. Sci., (IJRRCS), 2: 1- 8.

Kavitha, G. and V .Sankaranarayanan, 2011. Resource selection in computational grid based on user Qos and trust. IJCSNS, Int. J. Comput. Sci. Network Secur., 11: 214-221.

Jamaludin, Z. and W.H.W. Ishak, 2011. A virtual repository approach to departmental information sharing. Am. J. Econ. Busi. Administ., 3: 18-23. DOI: 10.3844/ajebasp.2011.18.23

Buyya, R. and M. Murshed, 2002. GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing. Concurrency and Computation, 14: 1175-1220. DOI: 10.1002/cpe.710