World Journal of Engineering Research and Technology
WJERT
www.wjert.org
SJIF Impact Factor: 5.218

DESIGN OF A MOBILE MULTI AGENT-BASED RESOURCE ALLOCATION AND COORDINATION SYSTEM IN A GRID ENVIRONMENT

[1]Emuoyibofarhe O. J., *[2]Amusan E. A. and [3]Salahu A. S.

[1,2]Computer Science and Engineering Department, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria.

[3]Kwara State College of Education, Ilorin, Nigeria..

Article Received on 10/04/2018 **Article Revised on 01/05/2018** **Article Accepted on 21/05/2018**

***Corresponding Author**
Amusan E. A.
Computer Science and Engineering Department, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria.

ABSTRACT

Availability, discovery and proper sharing of resources on a grid need proper attention to optimize the expected goal. The existing static resources allocation mechanisms are not able to handle the dynamically changing characteristics and capabilities of the resources in grid environment. Mobile agent technology is used as an alternative approach for the design of distributed systems to the traditional centralized architecture for effective allocation and coordination of resources. However, bandwidth utilization, resource discovery time and rejection of agents are some of the drawbacks limiting the efficiency of existing agent-based systems. Hence, this work developed a mobile agent-based resource allocation and coordination model that focused on searching, allocating and coordinating resources that change dynamically which is also characterized with high percentage bandwidth utilization, low resource discovery time and a decrease in rejection of agents. The system was implemented using C# programming language run in the .NET framework and the designed model simulated. Simulation results showed that an increase in number of clusters produced a corresponding increase in the percentage bandwidth utilization (PBU) especially in Tree transfer topology more than single step and random topologies. Also, Rejection of agent (ROA) decreased by 20% when number of clusters was increased.

KEYWORDS: Computational grid, mobile agents, resource allocation, resource coordination.

1. INTRODUCTION

Grid computing is primarily concerned with sharing and allocation of resources such as computing power, networks and database, rather than mainly file exchange (Chevaleyre *et al*, 2006). Grid technology has been applied to computationally intensive scientific, mathematical, and academic problems through volunteer computing, and it is used in commercial enterprises for such diverse applications as drug discovery, economic forecasting, seismic analysis, and back office data processing in support for e-commerce and Web services (Wolski *et al*, 2001).

Allocation and coordination of resources involve searching for appropriate resources and allocating them to solve specific jobs. When allocating these resources, they may change dynamically, which makes resource coordination very critical. Resources allocation could be either centralized or distributed. In the centralized case, a single entity decides on the final allocation of resources amongst agent, possibly after having elicited the preferences of the other agents in the system. Typical examples of the centralized approach are combinatorial auctions, in which the central entity is the auctioneer and the reporting of preferences takes the form of bidding. On the other hand, allocations emerge as a result of sequence of local negotiation steps in distributed approach. Such local negotiation is often restricted to bilateral trading as in the classical Contract-*Net* approach, but systems allowing form bilateral exchanges of resources between more than two agents are also possible (Chevaleyre *et al*, 2006).

Current grid systems are somewhat rigid and inflexible in terms of their interoperability and interactions, while agent based systems can be used as serious distributed systems which are robust, secure, requires scaling, and with bandwidth constraint (Foster, Jennings, and Kesselman, 2004). A major limitation of the agent-based approach discussed above is that whilst multi-agent systems are used to trade for grid resources, the trading tends to happen at the higher “service” level and not at the base “resource” level. In this study, a multi-agent architecture based on the hierarchical organization is proposed in other to overcome this limitation.

2. LITERATURE REVIEW

Mobile agents are considered one of the most powerful forms of code mobility. They can exploit the high processing power available in the server machines by shifting the computations into the server side. A mobile agent is a software module able to migrate to among the hosts of a network and carry on a specific task. The state of the program that runs is saved, transported to the new host, and restored, allowing the program to resume where it left off. Mobile agent technology seems to be very adequate to cope with systems' heterogeneity (Valliyammai *et al*, 2010). Mobile agents are autonomous and intelligent programs that move through a network, searching for and interacting with services on the user's behalf and possess inherent navigational autonomy. Valliyammai *et al*, (2010) explained that mobile agents' role is vital in grid because to support the local monitoring, local correlation on management devices, deploying agents dynamically, good scalability, continuing execution on device when link-down or unreliable and return results when available. Mobile Agents provide an effective way of migrating code and data together and return the results to the original user. (Valliyammai *et al* 2010).

Distributed information processing is a primary application of agent technology. Consider a mobile computing scenario with low bandwidth connection links. Agents suit them in a better way because the code to be executed migrates into the network leaving the portable device and performs necessary actions in the remote execution site. Only the result is returned back to the mobile device The techniques devised for protecting the agent platform are software-based fault isolation, safe code interpretation, signed code, authorization and attribute certificates, state appraisal, path histories and proof carrying code. While those for protecting an agent are partial result encapsulation, mutual itinerary recording, itinerary recording with replication and voting, execution tracing, environmental key generation computing with encrypted functions and unclear code (Valliyammai *et al*, 2010).

2.1 Related Works

The Network Weather Service (NWS) is a distributed system used by several grid systems for producing short term performance predictions of computational and network resources. It involves monitoring and prediction but does not include (re)scheduling of tasks. NWS works on small time scales; hence, it is not suitable for applications which take hours to run. Besides, uncertainties on applications demands are not accounted in predictions. Errors in estimating the execution time on the order of 25% were reported. Moreover, NWS produces

graphics for bandwidth availability predictions in which the differentiation between predicted and measured values are not easy to evaluate (Daniel and Nelson, 2008).

The Grid Architecture for Computational Economy (GRACE) allocates resource on the basis of supply and demand dynamics. Resource broker deals with resource discovery and adaptation of resource allocation given changes in availability. It presents the grid to the users as a single computational system. However, the major drawback of this proposal is the lack of flexibility to adjust the resource allocation given changes of resource availability. The major contribution of this research is to maximize incentives to the provider of resources (Boutilier, Bacchus and Brafman, 2004).

The Cactus Worm system employs code migration and monitoring to allow applications to adapt its resource allocation, when required performance level is not achieved. However, intermediate nodes can become a bottleneck. Cactus Worm does not deal with uncertainties. It was evaluated on the GrADS testbed, a grid composed of American universities. Experimental results show the advantage of Cactus Worm adaptive mechanism. However, there is no concluding data on the performance of the whole system (Daniel and Nelson 2008).

The Framework for Dynamic Grid Environments was targeted to promote changes when fault occurs as well as when resource availability increases. It was evaluated on TRGP testbed and a decrease of execution time of the order of 30% was observed. Experiments were conducted involving only CPU intensive applications and, therefore, there is a need to consider data intensive applications to derive broader conclusions on its performance (Daniel and Nelson 2008).

Cao *et al,* (2002) developed an agent based resource management system (ARMS) on grids. Its design methodology focused on solving the problems associated with scalability and dynamism in large multi-agent systems. It is not clear whether agents in the ARMS system bundle CPU resources together to form a service or indeed what the differences in granularity are between the PACE scheduler and the agent based resource negotiation however, it seems that the use of agents is primarily targeted toward service discovery whilst the PACE scheduler is targeted toward the machine level scheduling.

3. METHODOLOGY

The research method employed in this work is design and simulation. The first step was to design a mobile agent-based resources allocation and coordination system in a grid environment. Next was to simulate the designed grid environment model, and lastly, evaluate the performance of the developed system.

3.1 System Design

The grid model was designed based on three agent types, which are Job Request Agents (JRAs), Broker Service Agents (BSAs) and Resource Producer Agents (RPAs).

The JRAs were developed to represent the users for completing the job requests and submitting the jobs to the Broker Service Agents (BSAs), while BSAs served as resource schedulers as well as the brokers in the system. The system was developed using Random, Single Step and Tree topologies where percentage bandwidth utilization (PBU), Average Resource Discovery time (ARD) and Rejection of the agents (ROA) were used as specifications. Resource Producer Agents (RPAs) reside in the nodes of the local clusters, set the price of resource to maximum, average and minimum price. The system's physical layout is as shown in Figure 1.

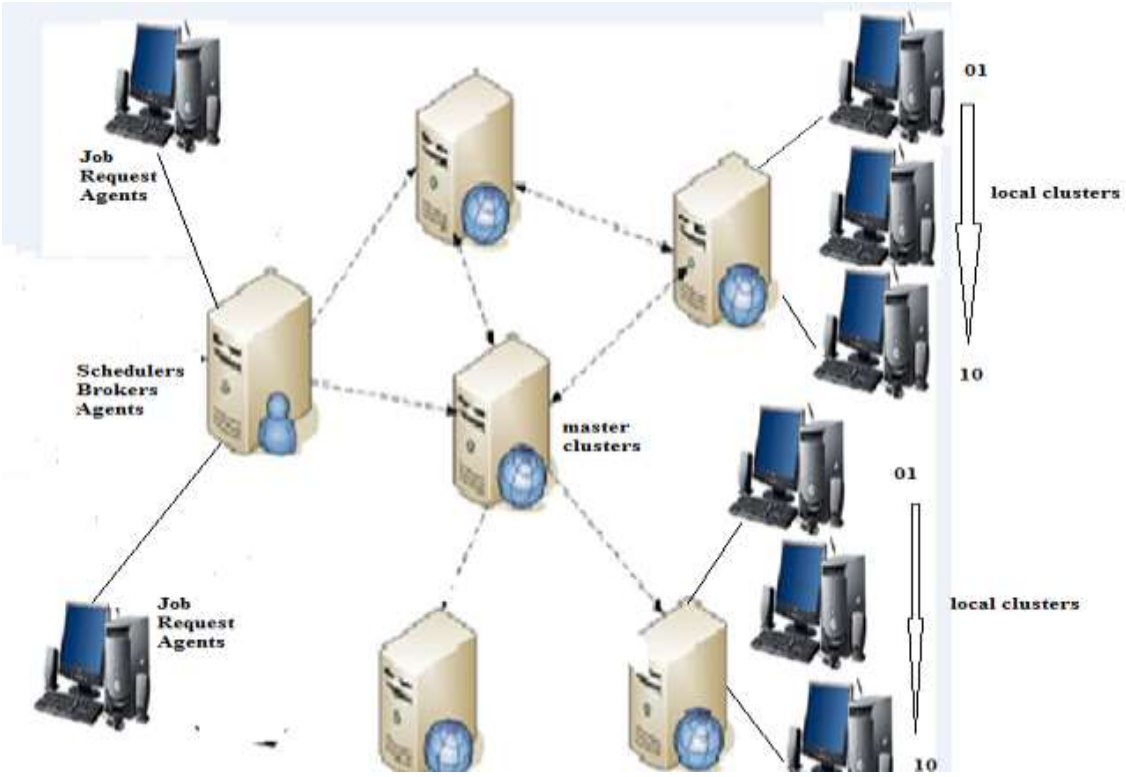

Figure 1: Physical Layout of the Designed System.

The system is a collection of user nodes, resource owner nodes and local and global servers, all connected to networks, mobile agents representing users, resource owners and taking care of its job execution. The middleware consist of wrappers, each wrapping a user program, and user programs written in C# programing language run in window .NET. The system assumes the availability of an agent platform at all the nodes of the network. The system communicates through the internet platform only. It is assumed that each grid node consists of a finite number of resources such as buffers, processing power, bandwidth, and storage devices.. Figure 1 shows the physical layout of the system while Figure 2 depicts the grid model that consists of master cluster servers, local cluster servers, routers, adapters and nodes, grid information servers, grid resource brokers, and users.

3.2 System Architecture

Multi-agent system architecture is proposed for grid resources allocation in this research. The architecture of model is as shown in Figure 2. The agents in the architecture are organized in hierarchies and comprise the Job Request Agents (JRAs), Broker Service Agents (BSAs) and the Resource Producer Agents (RPAs). The system model consists of master cluster servers, local cluster servers and nodes, grid information servers, grid resource brokers, and users. Local cluster has a set of machines, cluster server and was implemented as a local area networks (LAN), which incorporate local scheduler. Several local clusters are grouped and controlled by a master cluster server implemented as a wide area network (WAN) which incorporates global scheduling. The master cluster server collects information about the resources in its local clusters and then stores the information in its own database and also supplies the information to the grid information servers. The resource brokering agents discover the resources through the grid information servers.

The scheduling system used the scheduler that match available jobs to available resources and ensure that submitted jobs meet their deadline.

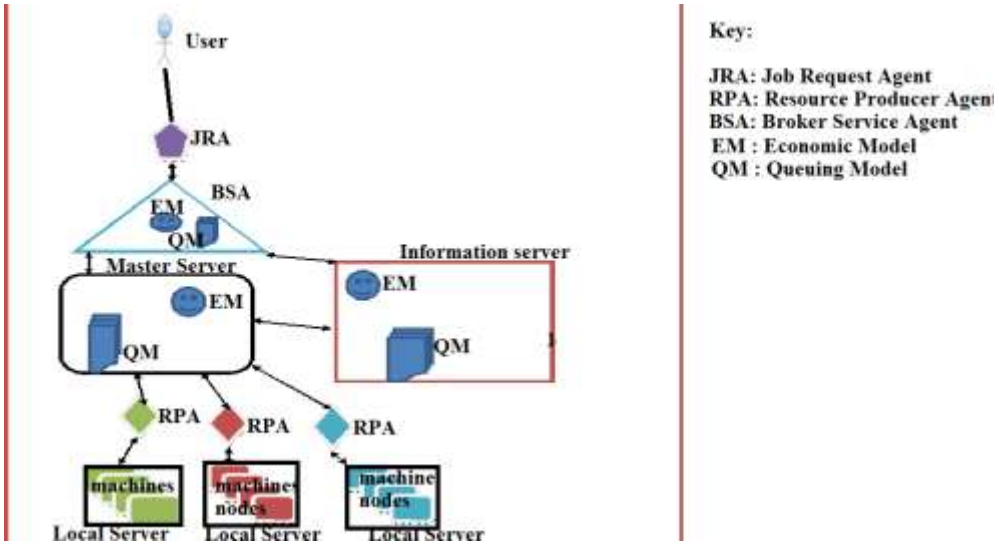

Figure 2: System Model Architecture.

3.2.1 Job Request Agents (JRAs)

Job Request Agents (JRAs) are mobile agent that represents a user and it accepts the job requests of the user, searches and sends the job to BSA. This system performs brokerage with the agents in the resource brokering nodes by negotiating within the given price limits. JRA reserves the resources by paying a higher price to BSAs.

3.2.2 Brokering Service Agents (BSAs)

The BSAs act as the resource schedulers as well as brokers for the users to submit the jobs (through the JRAs). The BSAs incorporate some economic and queuing models to address the dynamic behavior of the grid systems. The BSAs that present in different nodes communicate with each other in order to coordinate the pricing of the resources.

The job is submitted and queuing model of an RBA groups the JRA's. The grouping is based on the First-in First-Out (FIFO) manner. In this grid, BSAs are arranged in several different topologies. The BSAs make decision about the migration of the unallocated JRAs, according to the nature of the topology. Thus, three different kinds of topologies considered in this model are single step transfer or circular topology, random transfer topology, and tree shape topology.

3.2.3 Resource Producer Agents (RPAs)

RPA monitors the resources either periodically or continuously and informs the status related to resource utilization, resource availability, and price of the resource, to its information server. Thus, the RPAs indirectly convey the information to the BSAs.

3.3 SIMULATION ALGORITHM

The simulation algorithm is as follows:

(a) Generate the grid with *Gn*grid nodes using a random, single step, and tree topologies. The number of users, number of master cluster, number of local cluster, number of job Agents, size of jobs, number of machines, amount of job Agent requirements that need to be processing, and the rate at which jobs are sent from the user, are all variables.

(b) Generate the resources producer agents at each of the local clusters. Each local cluster has a RPA with mean inter-arrival time of the jobs with Poisson distribution in msec (milliseconds). It also has its own local scheduler and JRA requirements for the resources are distributed uniformly between the range [*w*, *z*]. Each resource requirement is generated uniformly between the range [Q_a, Q_b], which is normalized. The basic prices of the resources at a grid are uniformly distributed between <*p*1, *p*2> units.

(c) Generate the BSA with minimum and maximum number of JRAs that are generated by the users. Where Minimum and Maximum utilization threshold values for the resources are *MUT* and *XUT* respectively

(d) Resource brokering nodes are generated for a set of grid nodes. The timeout for a JRA is considered as β seconds. The quantity of a resource available at each of the local clusters is distributed between the range [0, 1], which is normalized. The number of resources available is between the range [R_x, R_y].

(e) Apply mobile agent for resources allocation and coordination model. That size of each JRA is α KB (kilobytes). The agent migration time on a single hop is uniformly distributed between <*t*1, *t*2>.

(f) Compute the performance parameters.

3.4 PERFORMANCE METRICS

In the simulation of this system, three metrics were utilized to measure system performance. They are:

(a) The total grid bandwidth utilization (total used computation time/total available computation time).

(b) The amount of time a mobile agent spends to perform assigned duty

(c) Amount of agents that do not acquire resources.

3.5 Simulation Parameters

The number of users, number of resources, number of jobs, size of jobs, amount of machines, amount of processors a machine had and their processing power, the budget for brokers in auctions, and the rate at which jobs are sent from the user were used as variables.

3.6 Experimental Configuration

Varying the number of users shows the sensitivity of the scheduler relative to the user load on the grid system, while varying resources shows the sensitivity of the scheduler relative to resource availability and repeated for the three topologies.

4. RESULT AND DISCUSSION

The simulation interface is as presented in Figure 3 in which the number of local servers, master servers, inters arrival time and resource requirement can be given minimum and maximum values to avoid grid lock and overcrowding of the resources. With all parameters ascribed, simulation start button can be pressed and the agents start the bidding process as shown in Figure 4. The agent painted red symbolizes the agent biding at a particular time. If the job resources to be used is found, the job is run on a local cluster, however if the job is too large the master cluster divides the job to various local clusters, the various bits are compiled afterward and sent to the agent which sends the job the owner. When the simulation is completed, a summary of all resources discovered are shown, rejection ratio and time used for the simulation is displayed as shown in Figure 5.

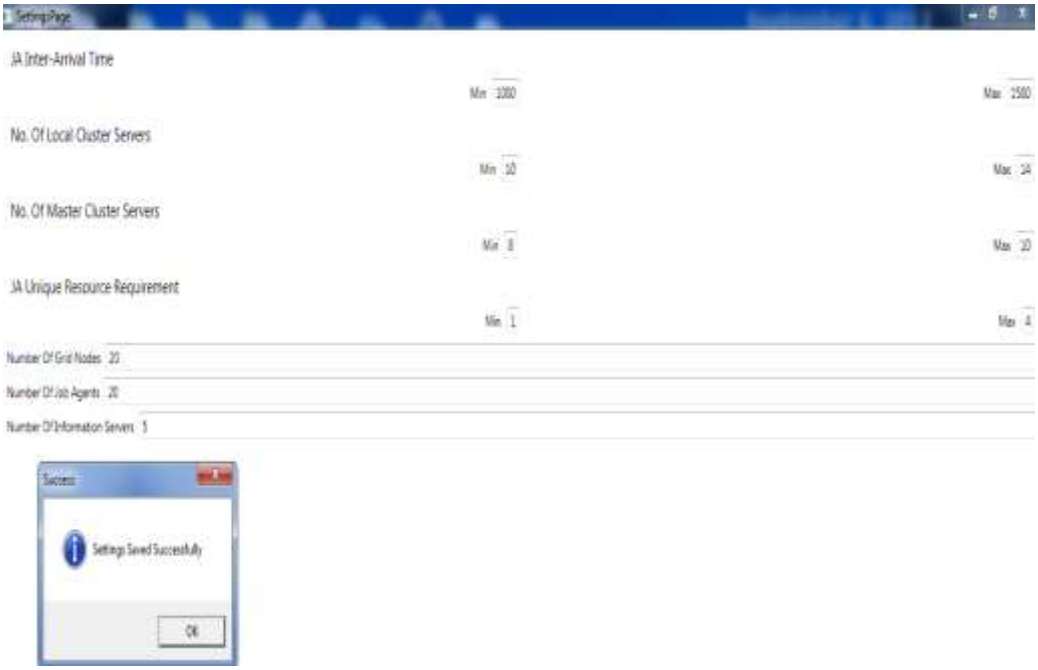

Figure 3: Simulation Interface.

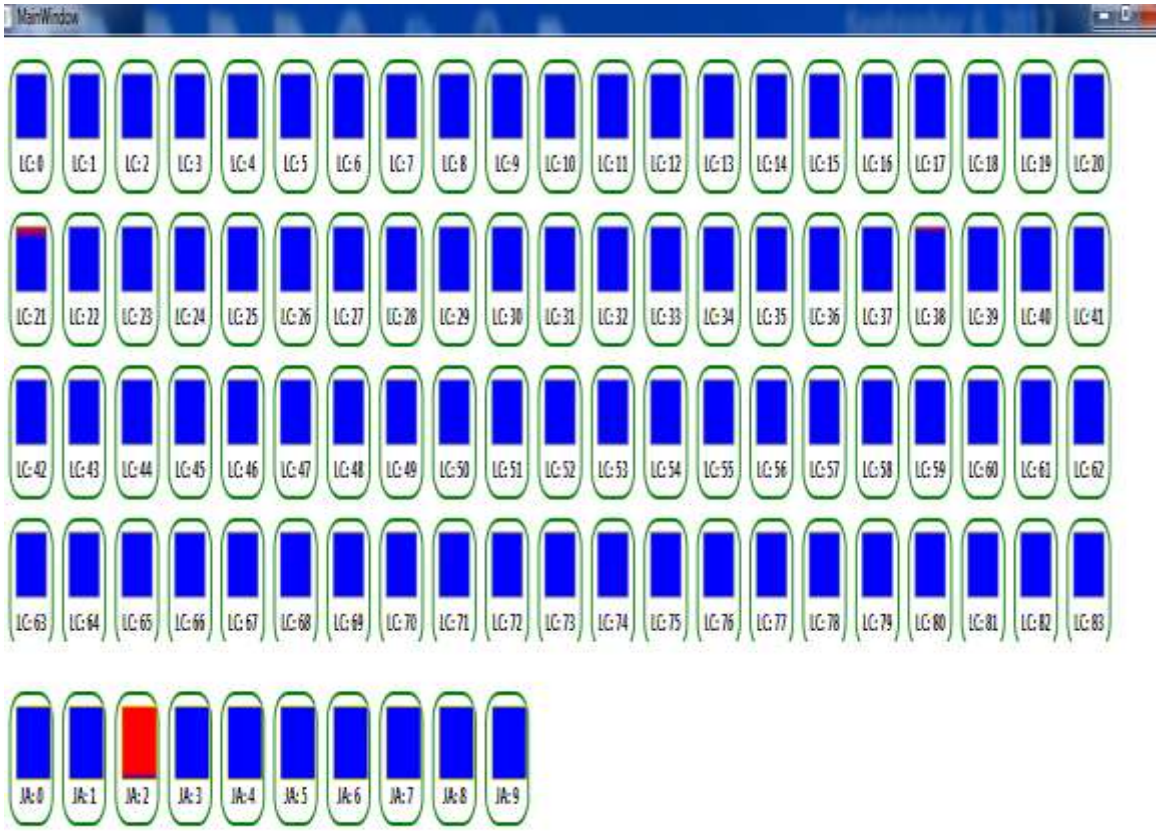

Figure 4: Pictorial Representation of Simulation Interface.

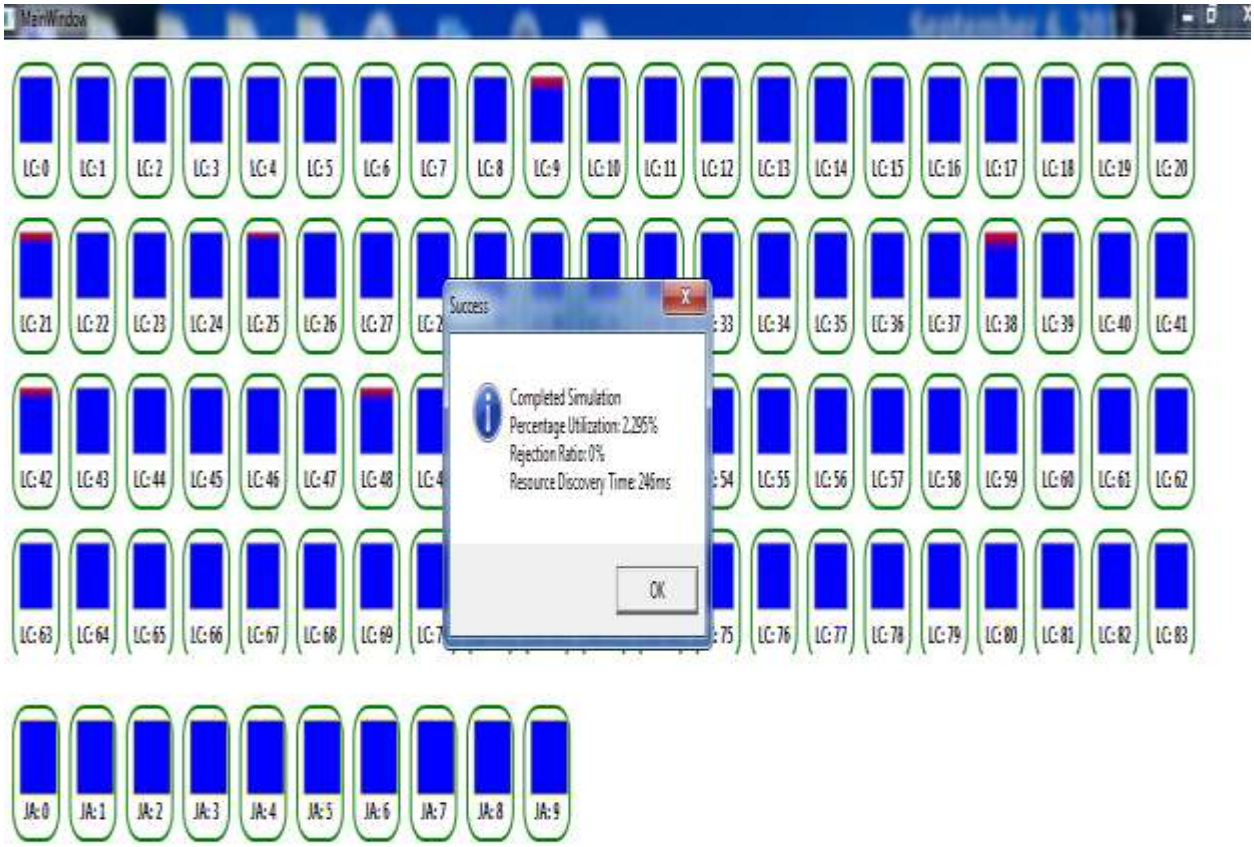

Figure 5: Completed simulation with Percentage utilization rejection ratio and resource discovery.

As shown in Table 1, when the number of job increase, the bandwidth utilization increases. The table also reveals that, as number of clusters and resources in clusters increased the bandwidth utilization is reduced. This is because, the allocated bandwidth is increasing suddenly and not in a uniform manner. The allocation of resources to execute certain job in an environment is not relevant with the requirements of potential requests for the bandwidth. It was observed that the bandwidth utilization value decreases as the number of JRAs increases. The tree topology gives better resource utilization when compared with the other two topologies, since the grid nodes are arranged in a tree-like hierarchical fashion. Mobile agent moved to the nearest resource broker agent in the tree structure and allocates the resources if available. If the resources are not available then the mobile agent moves to the parent of the previous resource broker agents and so on, until it gets the required resources to execute its job. It is noted from the figures that with an increase in the number of mobile agent requesting for resources, the mobile agent that arrive late have to move to more number of hops. This is because the nearest resources are already occupied by the mobile agents that started early. There is an increase in the overheads due to this problem.

For a tree topology, the resource utilization value decreases as the number of grid nodes increases and the number of JRAs decreases.

Table 1: percentage bandwidth utilization.

Clusters	Single step (%)	Random (%)	Tree (%)
200	54.9	48.5	68.8
163	44.0	48.7	68.5
150	40.6	65.7	68.3
137	36.5	65.9	66.1
124	41.9	80.7	62.5
111	40.6	57.2	57.8
98	48.4	33.7	75.0
85	36.5	68.9	81.2
72	56.2	45.5	81.5
59	60.3	73.0	81.7
46	60.3	41.4	83.9
33	56.2	42.8	87.5
20	14.0	71.6	92.2

The resource discovery process takes less time in the environments with more number of clusters and nodes as depicted by Table 2.

Table 2: Resource Discovery Time (gridsim sec).

Clusters	Single Step (ms)	Random (ms)	Tree (ms)
20	318	228	288
72	320	443	311
147	320	480	141
59	330	432	315
111	380	477	278
124	390	478	250
98	432	479	290
33	450	358	451
85	465	462	306
146	473	402	323
150	478	480	315
163	478	480	324
176	476	420	342

Table 3 illustrates the rejection of agents. The experiment behaves well with fewer numbers of resources, usually an increase in the number of jobs increases the number of job rejections. If the number of jobs exceeds the grid accommodating capacity, the rejection of the agents increases.

Table 3: Rejection of Agents.

Grid Nodes (N)	Single Step (%)	Random (%)	Tree (%)
180	0	3.3	0
297	13.4	13.5	6.7
531	37.6	56.5	9.4
648	33.8	67.9	12.3
765	7.8	0	15.6
882	15.8	9.0	2.2
999	18.0	16.0	4.0
1116	34.0	12.5	5.4
1314	45.6	38.1	6.1
1323	27.2	31.7	4.5
1350	60.0	20.6	10.3
1467	44.0	17.7	5.5
1584	19.0	15.8	7.6

5. CONCLUSION

The agent-based resource allocation model used a cluster based agent frame work, in which the nodes are clustered providing the same type of resource. It localized the searches to reduce the JRA travel thereby making it easy to avoid clue or priority for finding its requested resources. This type of arrangement has reduced discovery time, less bandwidth consumption, as the requests would be fulfilled in more number of hops and higher risk of rejection, as the time has an effect on the rejection. The bandwidth utilization, resource

discovery time and number of agent rejected are the main factors used as parameters to evaluate the performance of the system. On the other hand rejection of the agent had a direct relation with resource discovery.

Simulation results showed number of cluster increased from 20 to 200, PBU increased to 75% in Tree transfer topology more than single step and random topologies 48.4% and 57.2% respectively at 0 to 480 simulation time. The ROA decreased by 20% when number of clusters was increased. The results showed an improvement when compared with existing ones due to the reduction of the rejection of the agents (ROA). Also the system was able to serve large number of job request agents (JRAs) and percentage of bandwidth utilization (PBU) that is increased allow the resource owners to vary the price for cost benefit.

REFERENCES

1. Boutilier, C, Bacchus, F. and Brafman. R. "UCP networks: A directed graphical Representation of conditional utilities". In *Proc.* 17th Annual Conference on certainty in Artificial Intelligence, Morgan Kaufmann, 2004; 56-64.
2. Cao,J., S.A. Jarvis, S. Saini, D.J. Kerbyson and G.R. Nudd, "ARMS: An agent-based resource management system for grid computing". Sci. Program., 2002; 10: 135-148. http://www.dcs.warwick.ac.uk/~saj/papers/arms.pdf.
3. Chevaleyre, Y., Paul E, UlleE.,Lang, U., Estivie S., and Maudet. N. "Issues in Multi-agent Resource Allocation". In Engineering Societies in the Agents World, 2006; 123-124.
4. Daniel, M. Batista and Nelson L. S. da Fonseca. "A Brief Survey on Resource Allocation in Service Oriented Grids". Institute of Computing State University of ampinas Avenida Albert Einstein, 2008; 1251–13084-971.
5. Foster, I., Jennings, N. R., and Kesselman, C. "Brain Meets Brawn: Why Grid and Agents Need Each Other". Proceedings of 3rd Int. Conf. on Autonomous Agent and Multi-Agent Systems (AAMAS), New York, USA.
6. Valliyammai, C. ThamaraiSelvi S., Satheesh Kumar R., Pradeep E, and Naveen.K: "an alternate approach to Resource Allocation Strategy UsingNetworkMetricsinGrid", a paper presentation submitted to International Journal of Grid Computing and applications, 2010; 1: 1-8.
7. Wolski, R. Plank J. S., Brevik J., and Bryan T. "Analyzing Market-Based Resource Allocation Strategies for the Computational Grid". International Journal of HighPerformance Computing Applications, 2001; 15(3): 258-259.