Intelligent Method for Resource Allocation in Grid Computing Using Multi Agent

Bhagwat M. Fulmante[1,*], Raj Singh[2], Abhijieet Kulshreshtha[3] and Umashankar Sharma[4]
[1,*]Research Scholar, Jodhpur National University, Jodhpur, India.
[2]Research Scholar, Jodhpur National University, Jodhpur, India.
[3]Professor, Jodhpur National University, Jodhpur, India.
[4]Professor RJIT, BSF, Tekanpur, Gwalior, M.P., India.

In emerging technologies resource management is the key problem. This paper describes how to reduce the search time for the best available resources and assure instant provisioning of the lately added resources to the grid thereby using Artificial Intelligence, Multi-agent and Case Base Reasoning (CBR). To overcome the overhead of resource availability in grid computing, we have identified the formula which can efficiently identify the available nodes in the grid environment. The algorithm will efficiently search the nodes available as per requirement of the request of resources and searching is faster as compare to other existing algorithms. We have designed an algorithm which will identify the nodes available and the ideal node can also utilize. Each node will be equally loaded and resource allocation will be efficiently done. This approach significantly reduces the computational time of resource allocation. The Multi–agents has been used for decision making, selection and allocation faster for improving the productivity by reducing the searching time for resource allocation in grid environment. Each agent is assigned with some important task and will be communicating with other agents as input for next activity.

Keywords: Case Base Reasoning (CBR), Grid environment, Resource allocation and multi-agent.

1. INTRODUCTION

The last decade has seen a substantial increase in commodity computer and network performance, mainly as a result of faster hardware and more sophisticated software. Nevertheless, there are still problems, in the fields of science, engineering, and business, which cannot be effectively dealt with using the current generation of supercomputers. In fact, due to their size and complexity, these problems are often very numerically and data intensive and consequently require a variety of heterogeneous resources that are not available on a single machine or in single organization. These two factors combines and leading to the possibility of using distributed computers as a single, unified computing resource, that is popularly known as Grid computing.

Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed autonomous resources (owned by different organizations) dynamically at runtime depending upon their availability, capability, performance, and cost and users quality of service requirement. Grids are distributed

computational systems that allow users to access resources owned by different organizations [1]. It's because of the size and complexities of the problems in terms of computation and data which are rigorous and therefore results in the use of a variety of heterogeneous resources distributed across the globe to bring enormous computing power at lower costs that are not available in a single organization [4].

In the grid computing environment, the resources are not managed centrally and are autonomous [2] that is they can enter and leave the grid environment at any time. In grid computing the proficiency of the entire grid system is dependent on the resource allocation. As there is considerable change in the computational performance, the network bandwidth and resource availability, so there is a need for scheduling algorithms that can cope up with the changing environment. One of the main challenges is to allocate the best available resource in terms of CPU processing capability [3] primary memory and the associated network bandwidth in order to execute the job in minimal computational time.

Dorigo and Blum [5] propose Ant Colony Optimization (ACO) which is based upon the heuristic approach. It's an effective algorithm used for solving the resource scheduling problem in grid computing. It is based upon the natural behavior of the Ant which ousts chemical pheromone on its path in search for the food from their nest and the colony of ants work together to find the shortest path to the food source from their nest. The pheromone value defines the chemical substance that the ants release when they move. The higher the pheromone value signifies shorter is the path.

Stutzle [6] proposes Max-Min Ant System (MMAS) is a hybrid algorithm, which combines local search method for the quadratic assignment problem with the MMAS which makes strong impact upon the performance depending upon the type of instance i.e. structured or unstructured.

Yan et al. [7] applied the basic idea of ACO which improve the ant algorithm for job scheduling by adding encouragement, punishment coefficient and load balancing factor during the pheromone update function. The pheromone value of each resource is based upon the status and the jobs are assigned to the resource with the highest pheromone value. The pheromone value of each resource is updated by the update function. The encouragement and punishment and local balancing factor is defined by users which is used to update the pheromone value of the resource. If the job is completed successfully encouragement coefficient is added which will increase the pheromone value in order to be selected for next job execution. If the resource fails to complete a job, it will be punished by adding less pheromone value. The load of each resource is taken into account and balancing factor is applied to change the pheromone value of each resource.

In this paper we have described methods for resource allocation using CBR techniques and Multi-agent to effectively manage the matching, selection and allocation of the resources using the formula which can efficiently identify the available nodes in the grid environment. To make the system for efficient and productive multi-agent has been used for different task. This will ensure instant provisioning of those resources and reduces

the search time for the optimal available resource so as to reduce the computational time of the job in the grid environment.

2. METHOD

The proposed method mainly consists of two steps: (1) Selection of best nodes. (2) Selection of best path. The algorithm has been implemented with the help of multi – agent. We have used the four agents. For evaluating the resources request submitted by any user in grid environment we used request-submit agent. The second agent is matching and selecting agent. This agent is used for finding the available nodes in the grid environment with node details like node-ID and resources available. Based on this a list of available resources which can accommodate the requested resources will be generated. Then the third agent which is named as load calculator which will calculate the congestion and distance between the available candidates node and based on that the path delay for each node will be calculated, which in response to input from the second agent i.e. matching and selecting agent. Then the fourth agent named as intelligent allocator whose work is to analyses the output which is calculated by third agent i.e. load calculator and based on the total Path Delay (PD) will be compare with the candidate nodes and final allocation will be done based on the less value of path delay.

2.1. Matching and Selection of Best Nodes

In this step, we selected the best nodes (BN) which can fulfill the user demand. The resources that are available at each node in the network are stored in tabular form. The user resources demands are matched with all available resources at each node. Nearest neibourhood method was used matching and selection.

2.2. Selection of Best Path

In this step we select the best path (BP) from source to best node as follows:

If a path exist from i^{th} node to j^{th} node is $V_{i...}\ V_k \ldots\ V_{j..}$ i.e. V_k node lies between the path from V_i to V_j, and C_{ij} is delay in edge from V_i to V_j vertices, l_{ij} is the edge length from V_i to V_j vertices, then

The Path delay from V_i to V_k node is $pd_{ik} = C_{ik} * L_{ik}$

And total delay from source node Vi to node Vj is $PD_{ij} = \sum_{k=1}^{n} pd_{ik} + pd_{kj}$

BP=The path having delay Min (D_{ij} of Paths candidate node 1, D_{ij} of candidate node 2 ..)

Algorithm:

Step1. Request of resources submitted (Agent 1)

Step2. Searching of Available nodes by similarity nodes (Agent 2)

Step3. If node path found then Step4 else go to Step 7 (Agent 3)

Step4. Total delay from source node Vi to node Vj. (Agent 3)

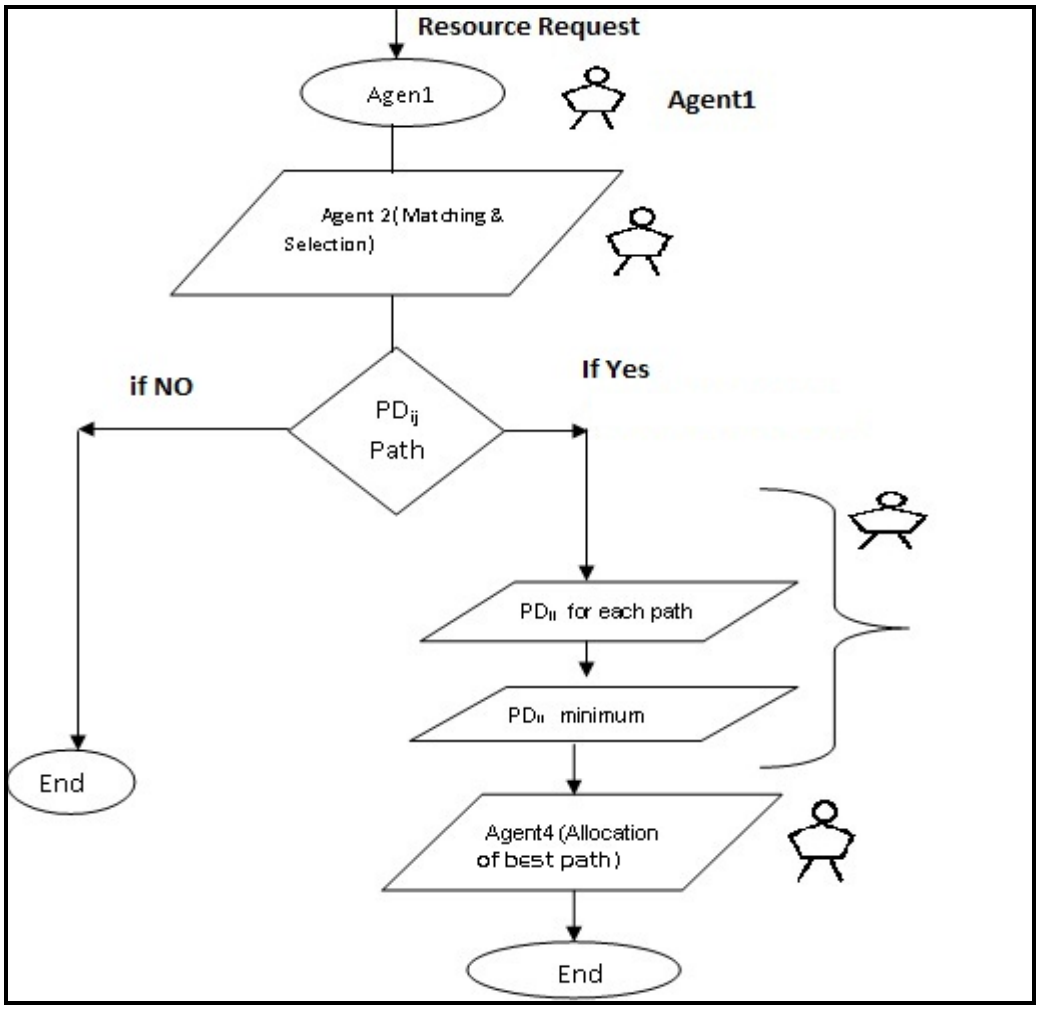
$$PD_{ij} = \sum_{k=1}^{n} pd_{ik} + pd_{kj}$$

Step5. BP=The path having delay Min (D_{ij} of Paths candidate node1, D_{ij} of candidate node2 …..)

Step6. Allocation of path (Agent 4)

Step7. End.

Flow Chart



The flow chart is showing the whole working after the submission of the request of the resources in the grid environment, based on the decision 'YES' the next step for delay calculation will start. Similarly the others steps for calculating the delay will be followed.

After calculating the delay for each node, the path having the less delay will be allocated. On the other side if the requested resources are not available then this will exit and 'END', will wait in waiting queue upto when there is any available resources.

In this work, we proposed an algorithm to allocate resources in grid computing. During the searching it will find out the resources which are available according the resource request. Then after that the number of node available which matches the request will be searched by this algorithm in the grid.

Based on the searching result, it will analyse the best suitable node which will fulfill the requested resource in optimal way. After that the best feasible node will be allocated to the request.

For example, we have a node network structure as shown in Figure 1.

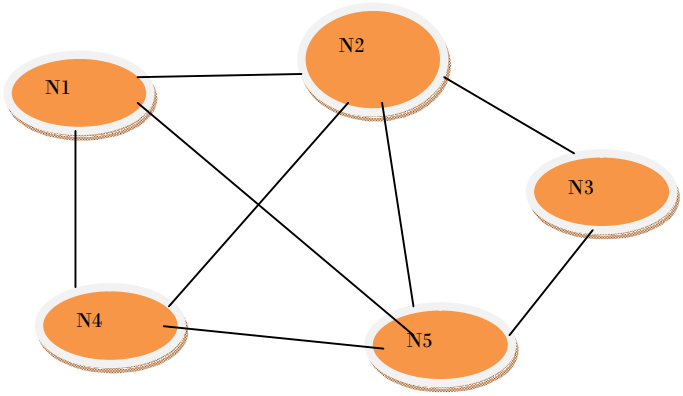


Fig. 1: Node network structure.

The Algorithm is implemented in such way that first of all it will search the available node in the grid. After that as per the requirement or the request of resources demanded, the available node will be searched.

3. RESULT AND DISCUSSION

In case base reasoning, after searching available nodes with resource will be displayed. On the basis of available nodes the case will be designed. The Table 1 is showing the contents like node id (NID), bandwidth (BD), random access memory (RAM), central processing unit (CPU), Hard disk drive (HDD), delay (DL), Congestion (CG), node distance (NL) and delay length (DL). The Final selection will be done on the basis of less delay value calculated at each node. The Table 1 is basically showing the details of resources available in grid. The node with least value of Delay will be taken as final allocation. Each new request is a new case for study.

In the Table 1 the congestion and node length will be calculated for each best path available for three delay congestion cases 1, 2 and 3 as shown in Figure 2, 3 and 4

respectively. The calculated result and delay value of each node will be compared, after that the node which is having less delay will be eligible for allocation.

Table 1: Case Base Reasoning.

NID	BD	RAM	CPU	HDD	CG	NL	DL
19	110	2	200	100	12	12	1200
15	100	3	200	52	7	7	365
30	150	3	200	50	8	8	400

For above mentioned example the delay value for case 2 i.e. Delay Congestion case 2, as per the calculated result the node id 15 is having less value for delay and this is our final allocation.

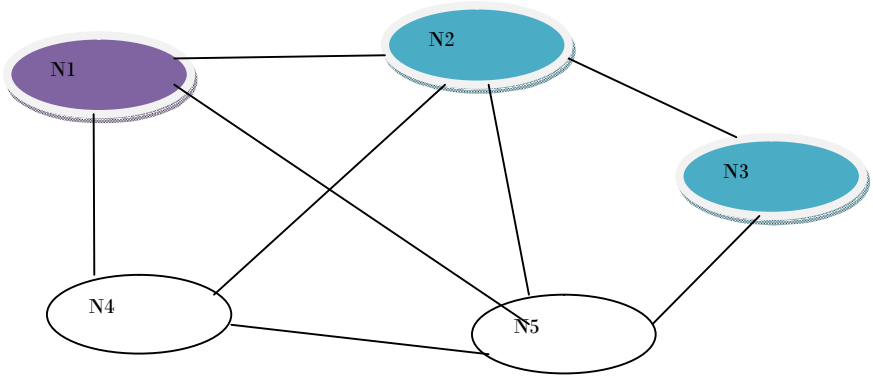


Fig. 2: Delay Congestion Case 1.

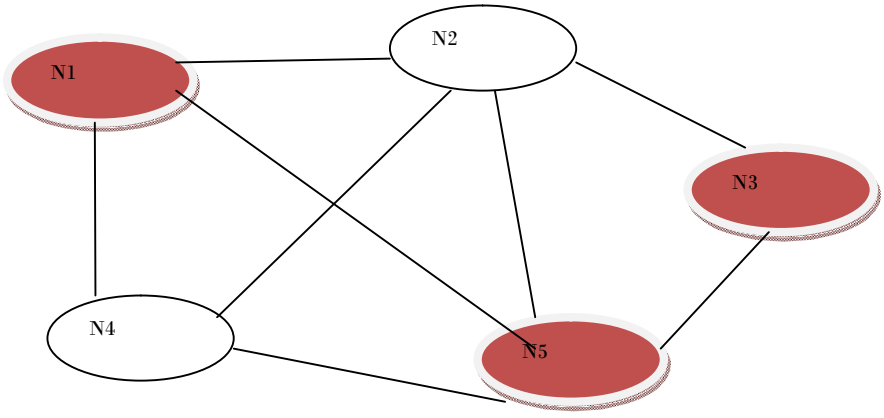


Fig. 3: Delay Congestion Case 2.

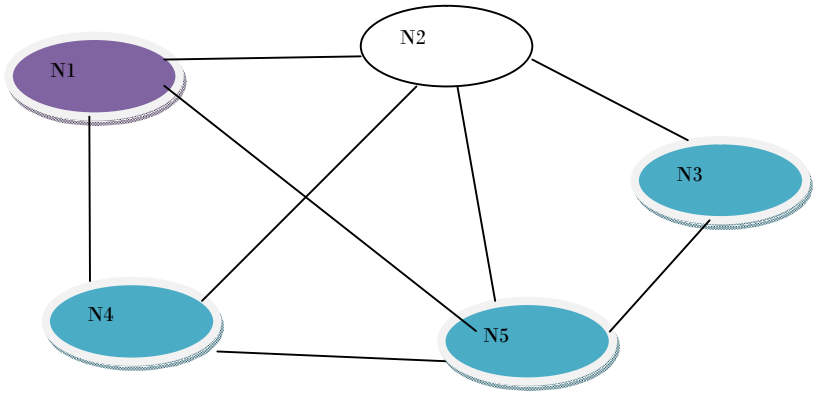


Fig. 4: Delay Congestion Case 3.

We have implemented this Graphical User Interface with the help of development tool Java, PHP and MySql. The front end is designed with PHP coding and backend is implemented in MySql. Agent programming has been done using java. The system is design and testing with dummy data for analyzing the results. The Agent is communicating with each other after getting some input from other agents. An intelligent approach has been introduced to make the resource availability faster and best allocation of node. Each agent is independent of taking decision of the basis of input received from the other agents.

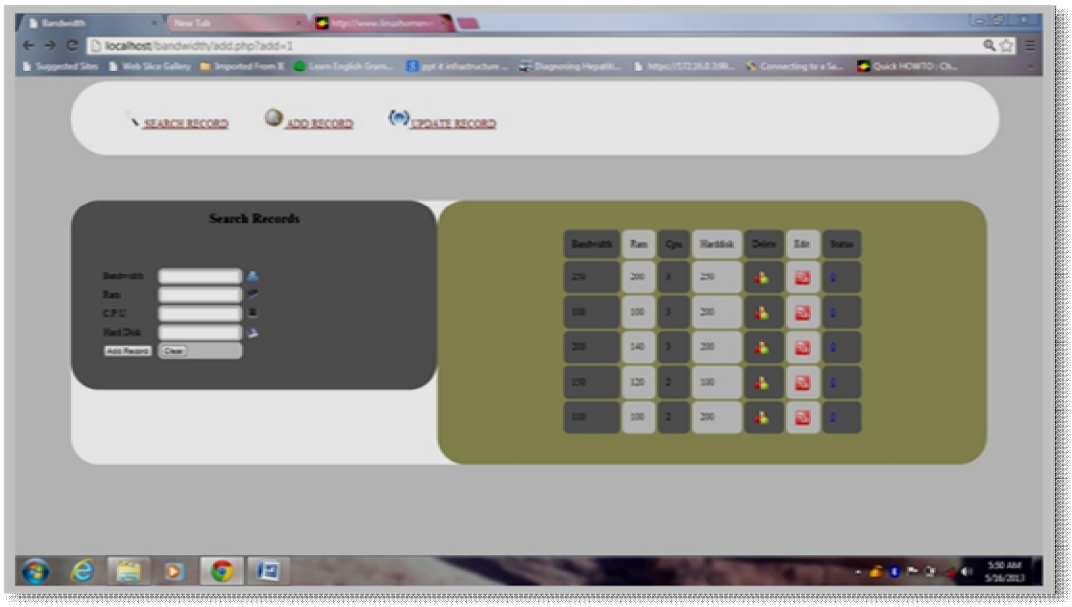


Image 1: Adding new resources.

The output screen in Image 1 (adding new resources) is shown for adding of resources i.e. bandwidth, CPU, Hard disk and RAM etc.

Image 2: Searching available nodes.

The output screen in Image 2 (Searching Available nodes) is shown for searching the all the nodes with their node id and resource capacities available for this request to accommodate.

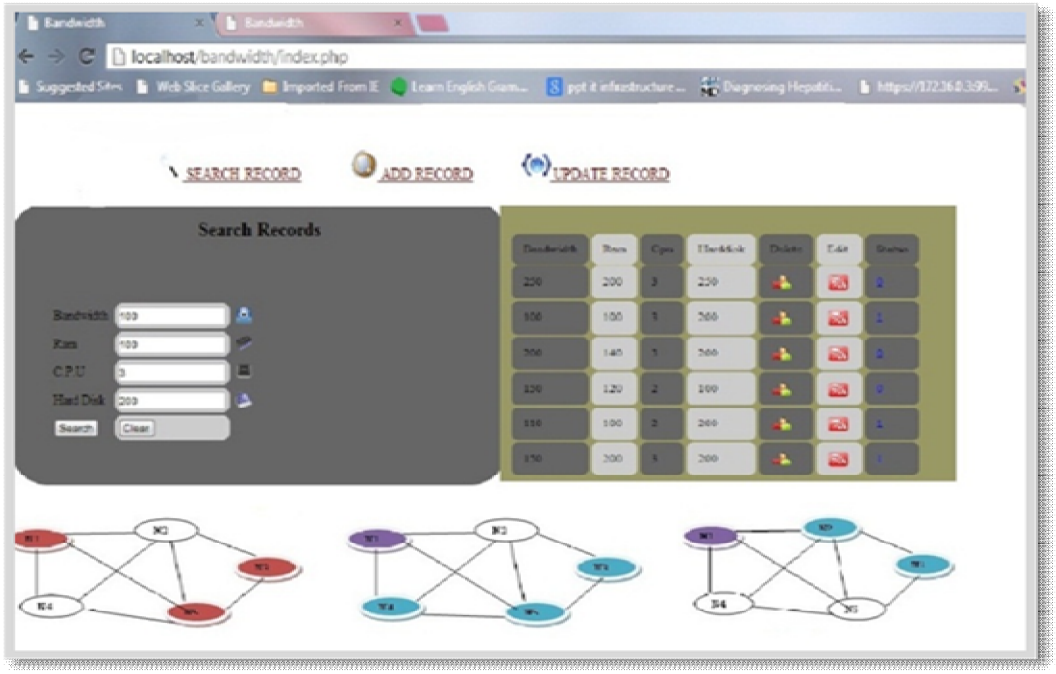


Image 3: Best path selection.

The output screen in Image 3 (Best path selection) is shown for the best suitable node having requested resources after calculation of congestion on network and distance between the each node. Based on these parameters it will calculate the delay for each node.

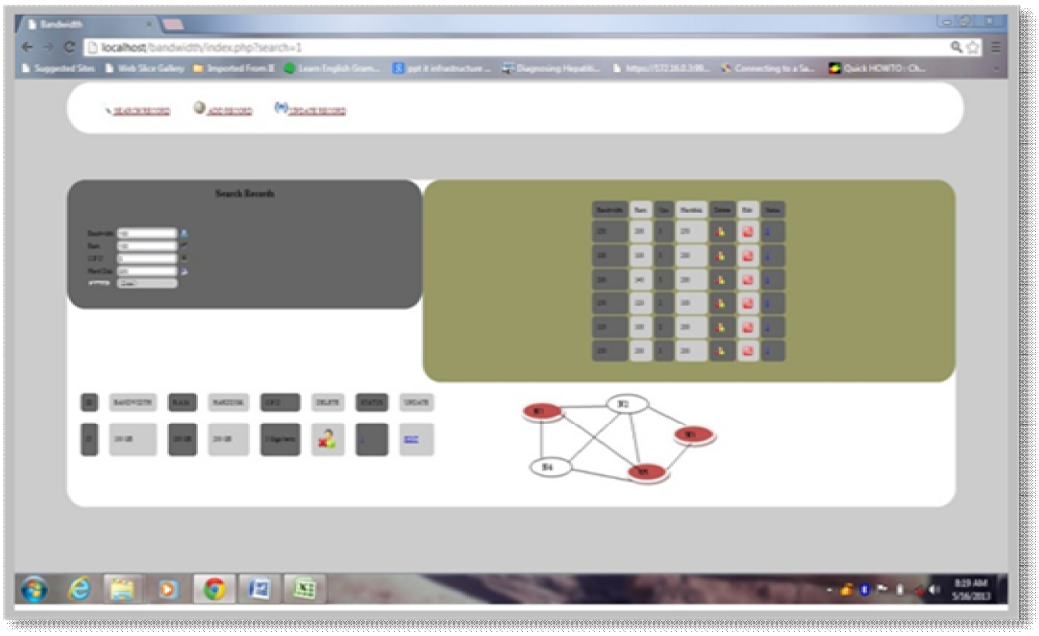


Image 4: Final best path allocation.

The output screen in Image 4 (Final best path allocation**)** is shown for comparing the total delay of each node respective to each candidate node. Final path will be selected and allocated for request to be entertained.

We have assumes that the best node is as per case base reasoning having less delay.

$$DL = NL \times CG$$

Where NL=Length of node and CG=Congestion.

When the value of the delay is less that particular path will be allocated.

4. CONCLUSION

The problem of allocating resources in Grid scheduling requires a method that allows local and external schedulers to communicate to achieve an efficient management of the resources themselves. In this paper, we have presented an algorithm for Grid resource allocation, showing the interactions among the involved agents. The each agent is communicating with other agents to get as input and the fast response is generated. Each agent named has played his role and communicated the output of one agent as

input for other agents. In first step the matching and selection will be done and in second step the best path will be selected. In the third step the final allocation on the basis of congestion, length (distance between the candidate's nodes) and the delay will be calculated for each candidate node. The best paths having less delay value will be allocated. This method will be reducing the overhead of assigning and managing and ensures the instant provisioning of the resources to the user. This method significantly reduces the computational time for resource allocation.

REFERENCES

[1] I. Foster and C. Kesselman (Eds.); "The Grid 2: Blueprint for a new computing infrastructure", Morgan Kaufmann, 2nd edition, 2004.

[2] H. Stockinger; "Defining the grid: a snapshot on the current view", J Supercomput., Vol. 42, pp. 3-17, 2007

[3] H. Bansal, B. Pandey and K. Krishan; "Intelligent Methods for Resource Allocation in Grid Computing", International Journal of Computer Applications, Vol. 47(6), pp. 1-5, June 2012.

[4] Raj kumar Buyya; "Economic-based distributed resource management and scheduling for Grid computing", Ph.D. Thesis, School of Computer Science and Software Engineering, Monash University, Melbourne, Australia.

[5] M. Dorigo and C. Blum; "Ant colony optimization theory: A survey" Theoretical Computer Science, Vol. 344, pp. 243-278, 2005.

[6] T. Stutzle; "MAX-MIN Ant System for Quadratic Assignment Problems", Technical Report AIDA-97-04, Intellectics Group, Department of Compute Science, Darmstadt University of Technology,Germany, July 1997.

[7] Y. Hui, Xue-Qin, L. Xing and W. Ming-Hui; "An Improved Ant Algorithm For Job Scheduling In Grid Computing", Proceedings of 2005 International Conference on Machine Learning and Cybernetics, Vol. 5, pp. 2957-2961,18-21 August 2005.