

# Intelligent Methods for Resource Allocation in Grid Computing

Harsh Bansal

Department of Computer  
Science and Engineering  
Lovely Professional University  
Phagwara, India

Babita Pandey, PhD.

Department of Computer  
Science and Engineering  
Lovely Professional University  
Phagwara, India

Kewal Krishan

Department of Computer  
Science and Engineering  
Lovely Professional University  
Phagwara, India

## ABSTRACT

In the era of grid computing, resource allocation plays a vital role for assigning the available resources. 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 clustering and artificial neural networks. The efficacy is achieved through K-Means clustering algorithm which is used to cluster the similar type of resources on the basis of their configuration as high, medium or low thereby decreasing the search time by searching only into the cluster of high availability instead of searching for the best from all of the available resources. Thereafter artificial neural network trained with feed forward propagation is deployed to automatically assign the newly added resources to appropriate cluster. This approach significantly reduces the computational time of resource allocation.

## Keywords

Grid Computing, Resource Allocation, K-Means Clustering, Artificial Neural Networks.

## 1. INTRODUCTION

Grid computing[1,2] lays the foundation of seamless next generation distributed computing environment which offers services to the ubiquitous end user for solving large-scale problems thereby enabling the sharing of the geographically distributed heterogeneous resources in a widely connected network such as internet. Though there has been a substantial increase in the network bandwidth and the processing capabilities of the hardware through more refined software at low costs, but there are still problems in the field of science, engineering and business which cannot be solved effectively with supercomputers though. 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 [3].

In the grid computing environment, the resources are not managed centrally and are autonomous [4] that is they can enter and leave the grid environment at any time. Similar is the case of jobs which arrives the grid environment at any time and leave after their completion. This adds extreme vagueness because of complex changeable conditions. 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,

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 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.

Lorpunmanee et al [8] proposed an ant colony optimization for dynamic scheduling in grid environment to minimize the tardiness time. It not only improved the overall performance of the system but also adapts to the dynamic grid system. The initial pheromone value of each resource is based on expected and actual execution time of the job submitted. The pheromone value on each resource is updates using the local and global update functions as in ACS. It compared the performance of various job schedulers and dispatching rules for grid environment and as a result ACO algorithm performed the best with respect to First Come First Serve, Maximum Time Earliest Due Date and Minimum Time Earliest Release Date.

A Bio-inspired adaptive job scheduling mechanism on a computational grid was proposed by [9] which used various ant agents with simple functionalities. It aimed to minimize

the execution time of the computational jobs by making use of the distributed resources. Resources assigned pheromone values whose value depends upon the execution time. Higher the execution time larger is the pheromone value. It compared bio inspired scheduling with heuristic and random mechanisms, which concluded that bio inspired adaptive job scheduling has good adaptability and robustness in a dynamic computational grid

Ruay-Shiung Chang et al [10] Balanced Job Assignment Based on Ant Algorithm (BACO) for Computing Grids focused on load balancing of each resource and on minimizing the computational time of the jobs in the Taiwan Unigrid environment. The algorithm chooses the best available resources to process the submitted jobs. Local pheromone and global pheromone update functions modifies the pheromone value after each allocation. Local pheromone update function updates the status of the selected resource after job has been assigned. Global pheromone update function updates the status of each resource after the completion of jobs. This ensures the latest information of all the resources available for the next job submission.

Mahamud et al [11] proposed ant colony algorithm for job scheduling in the grid environment which combined the techniques from Ant Colony System and Min-Max Ant System. The algorithm focuses on local pheromone trial update and the trial limit values. A matrix is used to record the status of the available resources. It makes use of the agent concept to update the grid resource table.

In this paper we have described methods for resource allocation using clustering and to effectively manage the addition and removal of the resources using artificial neural network. 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.

Section 2 describes Intelligent Computing Methods for Resource Allocation; Section 3 deals with results and the conclusions are drawn section 4.

## 2. INTELLIGENT COMPUTING METHODS FOR RESOURCE ALLOCATION

This section describes the proposed methods using K-Means clustering algorithm and artificial neural network for resource allocation.

### 2.1 K-Means Clustering Algorithm

The paper deploys K-Means clustering algorithm which is used to cluster the similar type of resources on the basis of their configuration as high, medium or low. The K-Means algorithm has been executed on the basis of the following two input parameters:

- 'k' – the number of clusters you want to find in the data which are set to three as we are clustering the resources into high, medium and low depending upon their configuration.
- The training set let it be of 'n' nodes (N); which are the resources in terms of CPU processing speed, RAM, storage and the associated bandwidth.

The procedure for classifying the clusters into high, medium and low is as follows:

**Step1:** Normalize each parameter by dividing the maximum parametric value with each parametric value of each resource.

$$C_{ij} = P_{ij} / P_{imax}$$

, where i is i<sup>th</sup> parameter in the j<sup>th</sup> node and C<sub>ij</sub> is the cost of each parameter in the j<sup>th</sup> node

**Step2:** Calculate the availability of each node as,

$$\sum_{i=1}^4 C_{ij}$$

**Step3:** Calculate the availability of the each cluster as,

$$HA_k = \sum_{j=1}^n C_{ij}$$

, where k is the cluster

**Step 4:** The cluster having maximum HA<sub>i</sub> minimum HA<sub>i</sub> are classified as high and low clusters respectively whereas the cluster having medium HA<sub>i</sub> is classified as medium.

### 2.2 Artificial Neural Networks

Artificial Neural Networks has been used to predict the assignment of the resources to an appropriate cluster using the same data which is used in K- Means clustering. The sample input data consists of a total of 26 nodes with processor speed, ram, and storage space and associated bandwidth as its features. The corresponding targets for the input nodes are respective output of the K-Means Clustering, signifying which input node belongs to which cluster. We have used feed forward back propagation training algorithm to construct and test trained neural network models. The training set is presented to the neural network software repetitively until the network has satisfactorily learned about the relationship between the resource configuration and the cluster. The performance graph of the trained neural network and regression between the actual and the target output are shown in Fig. 1 and Fig.2 respectively.

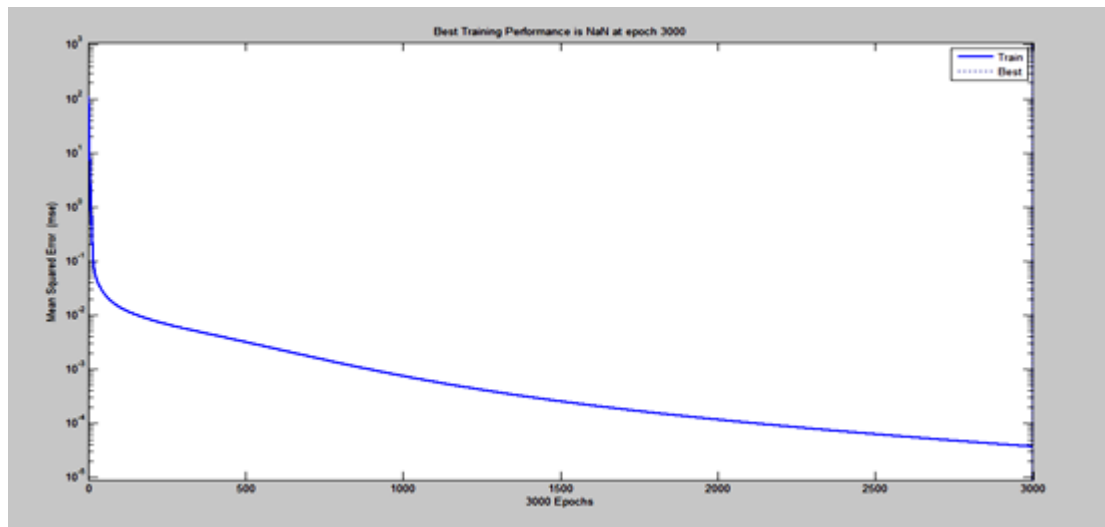


Fig 1: Performance graph of the trained neural network

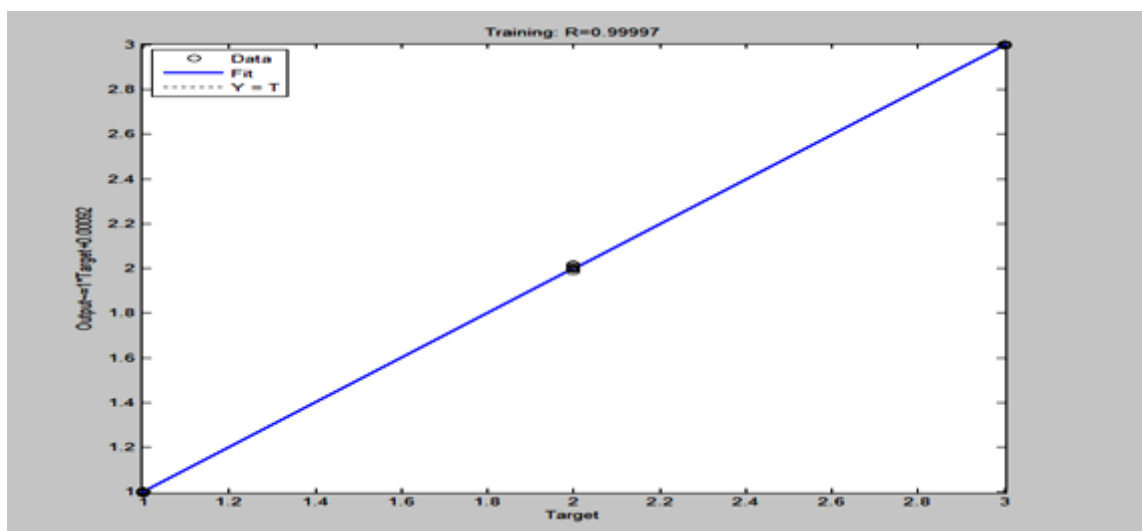


Fig 2: Regression graph between actual and target output with rate = 0.99997

The simulation of the resource assignment to the appropriate cluster is shown as below:

sim (net, [0.5384; 0.8684; 0.75; 0.5])

### 3. EXPERIMENTAL RESULTS

The assignment of the resources by the resource broker to the users has been simulated using a system designed in Visual Studio using C Sharp language. The resources are assigned on the basis of first come first served basis. The user input the resources needed for processing the job that is CPU processing speed (GHz), ram (GB) and the storage space (GB) as shown in upper part of Fig. 4. The proposed method determines the number of available cluster and starts by allocating the resources of highly available cluster on the basis of the input. It only searches first into the highly available cluster. If the highly available cluster has not

ans. = 2.9997, which signifies that the resource with provided information should be assigned to the third cluster.

sufficient free resources to process the request then it will automatically switch over to the medium cluster and so on. Status bits have been used to determine the availability of the resource features such as CPU status, RAM status, HD status. The status bit is altered when the resources are assigned or released. If the resource is unassigned the status bit is set to false and is summed into the total value of the resources available. If not, the status bit is set to true which signifies that the resource has been assigned and that much corresponding value is subtracted from the resource pool. The pseudo code for resource allocation is shown in fig. 3.

```

int clustercount, clustervalue[], cpucount, cpucount2, reqcpuvalue, addition, count, sum

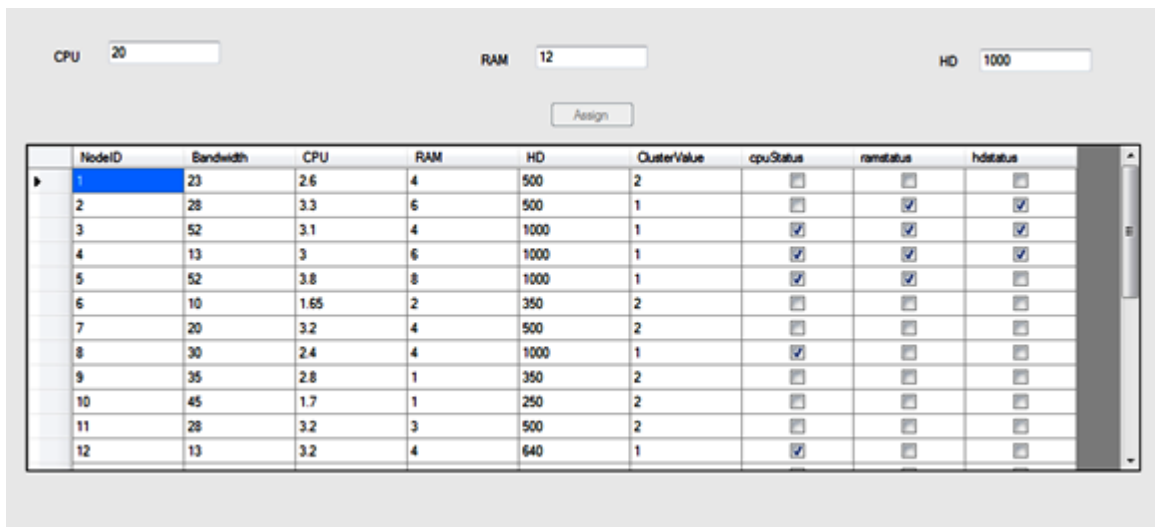
set clustercount to number of distinct clustervalue from nodes where cpustatus='false'
for i is 0, i is less than clustercount, i increments by 1
    set clustervalue to number of distinct clusters // stores the value of the distinct
clusters
ENDFOR
if textbox1.text //check the textbox if it has some value
    set reqcpuvalue to value of textbox1.text
    if addition is greater than reqcpuvalue // if the total value of cpu resources
available is greater than that of the user requested value
        for i is 0, i is less than clustercount, i increments by 1
            store the value for the number of cpuvalues in the cpucount from the database
            store the value for the number of cpuvalues in the cpucount2 from the database
            while result is not equal to 0 and count is less than cpucount and reqcpuvalue is
not less than 0
                set reqcpuvalue to textbox1.text - sum
                sum = sum + cpuvalue from the database
                set cpustatus of the assigned node == true
                increment count by 1
                cpucount2--
            ENDWHILE
            if cpucount2-- is equal to 0 and sums is not equal to reqcpuvalue
                continue; // call for next cluster
            else if cpucount2
                break;
            ENDIF
        ENDFOR
    else
        "Not Sufficient Resources"
    ENDIF
ENDIF

```

**Fig 3: Pseudo code for resource allocation**

Fig. 4 shows the allocated resources as per the needs of user. For example, the user requested for 20 GHz of CPU computing power, 10 GB of ram and 1000 GB of storage. It

first checks the resources in the highly available cluster. Thus, the nodes (2, 3, 4, 8, 12....) from this cluster are allocated.



NodeID	Bandwidth	CPU	RAM	HD	ClusterValue	cpuStatus	ramstatus	hdstatus
1	23	2.6	4	500	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	28	3.3	6	500	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	52	3.1	4	1000	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
4	13	3	6	1000	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
5	52	3.8	8	1000	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	10	1.65	2	350	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	20	3.2	4	500	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	30	2.4	4	1000	1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	35	2.8	1	350	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	45	1.7	1	250	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	28	3.2	3	500	2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12	13	3.2	4	640	1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

**Fig 4: Allocated resources as per the needs of the user**

#### 4. CONCLUSION

In this paper, intelligent computing methods for resource allocation using K-Means Clustering algorithm and Artificial Neural Networks have been proposed. The clusters generated by K-Means Clustering algorithm are classified into high, medium and low. The user request is first searched in the highly available cluster for the availability of the resources and hence decreases the search time by searching only into the cluster of high availability instead of searching for the best from all of the available resources, if there are not sufficient resources available then the other clusters are searched as well. The other benefit of this technique will be that it will help the resource broker in issuing the resources depending upon the priority for the user or jobs. Cluster of highly configured resources can be assigned to the users with high

priority, cluster of medium configured resources can be assigned to user with medium priority and cluster of low configured resources can be assigned to users with low priority. Artificial neural network is used to automatically assign the newly added resources to the appropriate cluster, hence reducing the overhead of assigning and managing and ensures the instant provisioning of the resources to the user. These methods significantly reduce the computational time for resource allocation.

#### 5. FUTURE SCOPE

Ant colony optimization (ACO) is an effective algorithm which is used for solving the resource scheduling problem in grid computing. These methods can be deployed to ACO for resource scheduling on the grid. A matrix that contains the pheromone value on each resource is used to facilitate the

selection of suitable resources to process submitted jobs. The algorithm selects the resource with the highest pheromone value for the submitted job thereby searching into the pheromone value matrix. The search process can be optimized by clustering the resources on the basis of their configuration into high, medium and low so that when there is a need for the best pheromone value to be searched it only searches into the highly configured cluster only and will save the time which is required for searching the pheromone value on resources which are assigned to cluster of medium and low availability, which will basically be not required as our purpose will be solved by the highly configured cluster.

## **6. REFERENCES**

- [1] Foster and C. Kesselman, "The grid: Blueprint for a new Computing infrastructure", 1999.
- [2] Foster, C. Kesselman, and S. Tuecke, "The anatomy of the Grid: Enabling scalable virtual organizations", Lecture Notes in Computer Science 2150, 2001.
- [3] Rajkumar Buyya, "Economic-based Distributed Resource Management and Scheduling for Grid Computing", School of Computer Science and Software Engineering Monash University, Melbourne, Australia.
- [4] Heinz Stockinger, "Defining the grid: a snapshot on the current view"
- [5] M. Dorigo, C. Blum, "Ant colony optimization theory: A survey" Theoretical Computer Science, Vol. 344, Issue 2-3, pp. 243-278, 2005.
- [6] Stutzle, T., "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] Hui Yan, Xue-Qin, Xing Li, Ming-Hui Wu, "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 Aug. 2005.
- [8] Siriluck Lorpunmanee, Mohd Noor Sap, Abdul Hanan Abdullah and Chai Chompoo-inwai, "An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment", International Journal of Computer and Information Engineering 1:8, pp. 469-476, 2007
- [9] Y. Li, "A Bio-inspired Adaptive Job Scheduling Mechanism on a Computational Grid", International Journal of Computer Science and Network Security, Vol. 6(3), pp. 1-7, 2006.
- [10] Ruay-Shiung Chang, Jih-Sheng Chang and Po-Sheng Lin, "Balanced Job Assignment Based on Ant Algorithm for Computing Grids", 2007 IEEE Asia-Pacific Services Computing Conference, pp. 291-295, 2007
- [11] Ku Ruhana Ku-Mahamud, Husna Jamal Abdul Nasir, "Ant Colony algorithm for Job Scheduling in Grid Computing", 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation, pp.40-44, 2010.