Fuzzy-Based Simple And Proficient Resource Allocation Technique For Dynamic Grid Resources

M. Poonguzhali
Department of ECE, Narasus Sarathy Institute of Technology
and
Dr.S.Shanmugavel
Department of ECE, Anna University

In the grid environment, not only the submitted jobs are dynamic but also the resources are found altering dynamically. Because of the dynamic nature, efficient utilization of the grid resources remains as a challenging issue. A proficient resource allocation technique is a pre-requisite to face the aforesaid grid issue. In this paper, we propose a simple and proficient fuzzy-based resource allocation technique, which effectively allocates the dynamic grid resources to the submitted jobs. The proposed technique consists of three different stages, namely, Classification of grid resources, Generation of fuzzy rules and Resource allocation based on the fuzzy rules. In the first stage, grid resources are classified into three categories based on their dwelling time. In the second stage, fuzzy rules are generated so as to decide whether a particular resource can be allocated to the demanded job or not. In the third and final stage of the technique, the resources are allocated to the submitted jobs based on the generated fuzzy rules. Eventually, the proposed technique is evaluated by means of three performance measures, namely, 1) Utilization, 2) Failure rate and 3) Makespan. Finally, the performance of the proposed technique is evaluated by calculating the measurements for the allocated resource to the submitted jobs.

Keywords: Fuzzy Rules, Resource Allocation, Permanent, Semi-Permanent, Sporadic, Classification, Grid Resources

1. INTRODUCTION

Grid computing makes huge collection of data distributed all over the world, which provides great comfort to engineers and scientists, as they could use that from anywhere around the world so it will create the situation as if they are in their own office or laboratory [Naqaash et al. 2010]."Grid" computing has emerged as an important new field, distinguished from conventional distributed computing by its focus on large-scale resource sharing, innovative applications and, in some cases, high-performance orientation [Muruganantham et al. 2010] . In general, Grid Computing can be defined as a type of distributed computing that facilitates the construction of a computational infrastructure by utilizing wide area distributed resources, such as, databases, storage servers, high-speed networks and clusters to resolve complex issues [Vijayakumar et al. 2008]. There are justified reasons which have made Grid computing a reliable technology. One of the reasons is the need to exploit the available resources in an economic way and another one is the need to allocate the resources to the respective commercial applications in an intelligent and secure way [Kaur et al. 2007]. In setting up a grid system, two important factors to be considered are scalability and adaptability. Grid systems are applied in the computation of complicated scientific simulations and in solving issues related to ecommerce, engineering, weather forecasting, defense calculations, mobile users and so on [Manvi et al.2005].

So far, majority of the works have used Grid computing to develop middleware that offers basic functionality, for example, the capability to query for information regarding the resources

Author's address: M. Poonguzhali,Narasus Sarathy Institute of Technology,Department of Electronics and Communication Engineering, Poosaripatty, Salem-636305, India.
Dr.S.Shanmugavel, Department of Electronics and Communication Engineering,Anna University, Chennai-600 025, India.

and the capability to schedule jobs onto the resources [Galstyan et al. 2004]. The environment is basically a multi-agent system which contains the two important agent types: resource agent and user agent. In most cases, users can be considered as individual agents that create jobs and attempt to access resources for their execution, or as external resource brokers that map jobs in support of other individual users. Generally, resources are described by the number and speed of the processors available, system memory, as well as storage space. Users have resource-sharing jobs that must be mapped to respective resource providers by means of a resource allocation system. Then the resource allocation system decides amid other mappings with the aim of optimizing metrics which are bounded by the virtual organization's policy environment [Blythe et al. 2003; Krawczyk et al. 2008; Li et al. 2008; Wankar 2008].

Usually, a static partition of the resources falls short to provide an agreeable performance guarantee. Conversely, over-provisioning, to be exact, providing an adequate amount of resources to meet the peak load, can result in the underutilization of the resources. A thoughtful option would be to dynamically regulate the resource allocation [Plaxton et al. 2006]. Management of resources should be handled in an efficient manner and moreover some other suitable pricing strategy has to be integrated in order to serve the users in a better way [Manvi et al.2005; Gomoluch et al. 2003]. In fact, they are very significant in providing easy access of resources and allow the users to utilize most of the grid capabilities [Kurowski et al. 2007].

Even though significant interest has been shown on the resource allocation problem of Grid computing and especially in making predictions for resource management [Wolski et al. 1999; Dinda 2001; Wolski et al. 1999; Foster et al. 1986], a small number of researchers have dealt with the problem from the viewpoint of learning and adaptation. In the intervening time, the multi-agent systems (MAS) and distributed AI communities have proved that groups of autonomous learning agents can effectively solve the resource allocation problems. Some of the resource allocation problems comprise of the following: 1) discovering a suitable service and the resources, 2) assigning the resources on the basis of particular conditions such as pricing or priority [Gomoluch et al. 2003], and 3) dynamically assigning and updating the status of the resources [Kaur et al. 2007; Li et al. 2008]. Hence, it is well known that in the grid environment, not only the submitted jobs are dynamic but also the resources are found altering dynamically. Because of the dynamic nature, efficient utilization of the grid resources remains a challenging issue.

Here, we propose a simple and proficient fuzzy-based resource allocation technique, for effectively allocating the dynamic grid resources to the submitted jobs. The proposed technique consists of three different stages, namely, Classification of grid resources, Generation of fuzzy rules and Resource allocation based on the fuzzy rules. In the first stage, grid resources are classified into three categories based on their dwelling time namely, permanent, semi-permanent and sporadic. In the second stage, fuzzy rules are generated so as to decide whether a particular resource can be allocated to the demanded job or not. In the third and final stage of the technique, the resources are allocated to the submitted jobs based on the generated fuzzy rules. Finally, the proposed technique is evaluated by means of three performance measures, namely, 1) Utilization, 2) Failure rate and 3) Makespan. The rest of the paper is organized as follows: Section 2 presents a brief review about the recent related works in the literature, Section 3 describes the proposed resource allocation technique with pseudo code and required illustrations, Section 4 discusses about the implementation results and Section 5 concludes the paper.

2. RELATED WORKS

[Kiran et al. 2007] have proposed a makespan algorithm by employing the novel idea of process replication. In their algorithm they have amply exploited the available resources and mathematically acquired a TPCC (Total Processor Cycle Consumption) which is just higher than the TPCC of other ideal parallel and distributed computing systems. The acquired TPCC value has been established to be slightly higher than the ideal value of n.L, where, n is the number of

tasks submitted and L is the number of instructions in a given task. The given algorithm has been found to lessen both the TPCC and the makespan.

[Chapman et al. 2007] have come up with an architecture and implementation for a predictive grid scheduling framework which is based on Kalman filter theory [Kalman 1960] to forecast CPU resource utilization. From the experimental results, it has been shown that prediction accomplishes a precision within 15-20% of the utilization which is observed later and could considerably enhance scheduling quality, compared to the other approaches that consider the current load indicators.

[Zhang and Phillips 2009] have proposed a new job scheduling algorithm. In order to provide good job allocation decisions, their scheduling algorithm has been based on the concept of a resource availability prediction technique, which predicts the existence of resources. The objective of their algorithm has been to ensure that jobs function efficiently in unreliable grid computing environments. Their algorithm has functioned far better than Production Prediction score (PPS) scheduler.

[Chen et al. 2009] have introduced a grid resource scheduling algorithm which is based on utility function. Their algorithm has been proposed to solve the issue of heterogeneity of user requirements in grid resource allocation, by examining the relationship between the execution time, cost and the user utility function. The algorithm has accomplished superior performance in terms of cost than the time based optimization algorithm and also in terms of time than the cost based optimization algorithm, when it consumed equal amount of time and cost respectively.

[Hao et al. 2008] have proposed Grid job schedule arithmetic. The proposed grid job schedule arithmetic has combined the trust mechanism and job schedule mechanism. The expectation trust benefit function value is the prediction benefit of a particular job that is executed in the grid. The expectation trust benefit driven arithmetic has been comparatively better than the conversional min-min arithmetic in terms of benefit and it has exhibited better performance in trust benefit also.

[Chen and Lu 2008] have presented a new grid resource scheduling algorithm that augments the consumption of resources and system throughput, and in addition, accomplishes the load balancing and utility optimization between resource providers and users inside the grid systems. They have described the diverse functional parts of the proposed algorithm and conducted two types of simulative experiments about completion time and cost of tasks. It is experimentally shown that the proposed algorithm is efficient.

[Chunlin 2008] has utilized the interlayer coupling of across-layer design concept in grid computing. They have proposed a joint application and fabric layer resource scheduling algorithm that merges the benefits of both the application-centric and the system-centric scheduling. They have devised an integrated design of resource scheduling and user QoS satisfaction control into a constrained optimization issue. The application layer has altered the user's resource demand which is based on the current resource settings, and the fabric layer has assigned CPU, storage and bandwidth as needed by the upper layer in an adaptive way.

[Jiang et al. 2009] have simulated and analyzed a new dynamic resource allocation method based on fuzzy modeling with heterogeneous QoS requirements and found out that it could distribute most suitable resource among the different services quickly and sensitively as the service QoS demand varies under the constraint of achieving maximum utilization of grid resources.

[Roblitz et al. 2006] have presented a scheme for reserving job resources with imprecise requests for handling fuzzy reservation requests. The evaluated algorithms with real workload traces from a large supercomputer site have indicated that the scheme greatly improves the flexibility of the solution process without significantly affecting the overall workload of a site. From a user's perspective, only about 10% of the non reservation jobs have a longer response time, and from a site administrator's view, the makespan of the original workload has been extended by only 8% in the worst case.

From the review, it can be seen that most of the literary works available in the literature intend to predict grid resources and to develop resource allocation algorithms. Resource allocation techniques would be effective and efficient if (i) better analysis and conclusions are provided by the techniques based on the predicted output and (ii) resource allocation is made as per the decisions made based on the predictions. However, the aforesaid factors are not properly considered in the reviewed works and hence they are ineffective and inefficient. So, to avoid these drawbacks we propose a new classification approach for predicting grid resources. It categorizes the resources based on the prediction parameters and performs grid resource allocation with the aid of fuzzy intelligence using the categorized results.

3. PROPOSED FUZZY-BASED RESOURCE ALLOCATION TECHNIQUE

Fuzzy logic system is very unique and it can deal with numerical data and linguistic knowledge. The basic concept of fuzzy systems is a fuzzy (sub) set [PourMohammadBagher 2008]. In the fuzzy logic, the Boolean logic is extended to process the partial truth concept which means that the truth acquires a value between a complete true value and a complete false value.

Fuzzy logic is also considered to be much more than a logical system because it has other facets also. The main facets are logical, fuzzy-set-theoretic, epistemic and relational. Majority of the practical applications of fuzzy logic are related with its relational facet [Zadeh 2008]. There are many positive aspects in this fuzzy logic system. It is conceptually easy to understand and is based on natural language [Siler and Buckley, 2005]. It is a best tool for dealing with the problems of uncertainties and imprecise information. The capabilities of modeling imprecise and qualitative knowledge and to handle uncertainty are the other notable features of fuzzy sets. It is also competent in addressing approximate or vague ideas that exists in many information retrieval (IR) tasks [Tseng 2007]. Fuzzy logic is employed in a variety of applications in several domains like estimation, prediction, control, approximate reasoning, pattern recognition, medical computing, robotics, optimization, industrial engineering, etc [Siler and Buckley, 2005; Ramirez-Gonzalez and Malik 2009; Xia et al. 2007].

We propose a fuzzy-based simple and efficient resource allocation technique for allocating dynamic grid resources to the submitted jobs. The technique can by used in the local scheduler to manage the resources available in a site based on the requirement data that are submitted by a meta-scheduler. The overall architecture of the proposed resource scheduling is represented in Fig 1. Here, the submitted jobs are considered to be static and they are prioritized. Let, the submitted jobs be $J_i; 0 \leqslant i \leqslant N_J - 1$, where,$N_J$ is the number of jobs submitted to the grid. Each job has its own priority level,P_i ; $P_i \epsilon$ (P_{min}, P_{max}) and it requires the resources , $R_{kj}; 0 \leqslant k \leqslant N_{ri} - 1$ where,N_{ri} is the number of resources required by the i^{th} job. Here, the job with priority level P_{min} is considered as the high priority job, while, the job with priority level P_{max} is considered as the low priority job. Let,$R_j^{(t)}; j\epsilon(0, N_R - 1)$, be the resources available in the grid at the t^{th} duration and each duration consists of p time periods.

From the architecture, it can be visualized that the proposed scheduling model consists of a resource classifier that classifies the grid resources according to certain resource features. The prioritizing unit provides information to the fuzzy scheduler about the priority of the jobs that are given to the grid environment. The prioritizing unit performs the process with the aid of the database, which has the details about the nature of jobs. The fuzzy scheduler allocates the possible resources to the submitted jobs based on the generated fuzzy rules.$R_0, R_1 ... R_N$ that are given in Fig. 1 are the resources that could be allocated to the submitted job according to the job nature as well as the resource nature. Hence in terms of operation, the proposed resource allocation technique is comprised of three stages, namely, Classification of resources, Generation of fuzzy rules and Resource allocation based on fuzzy rules. The flow diagram of the operation of the proposed resource scheduling technique is depicted in Fig. 2.

The three operations involved in the proposed resource scheduling technique are detailed below.

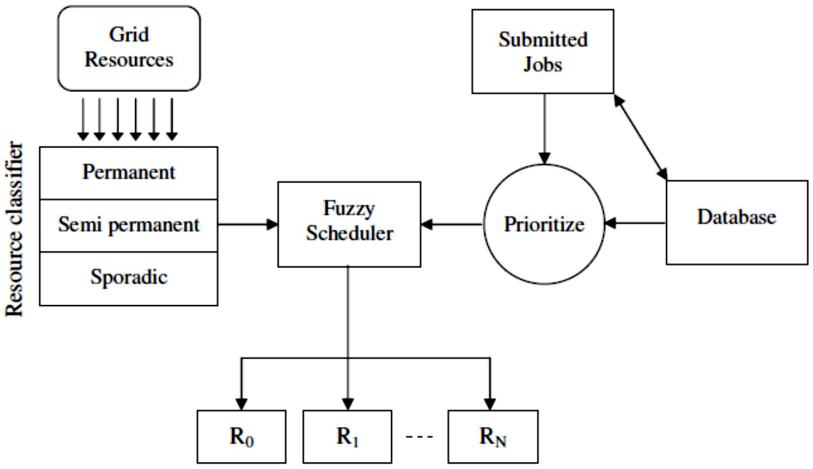


Figure.1 Architecture of the proposed scheduling model

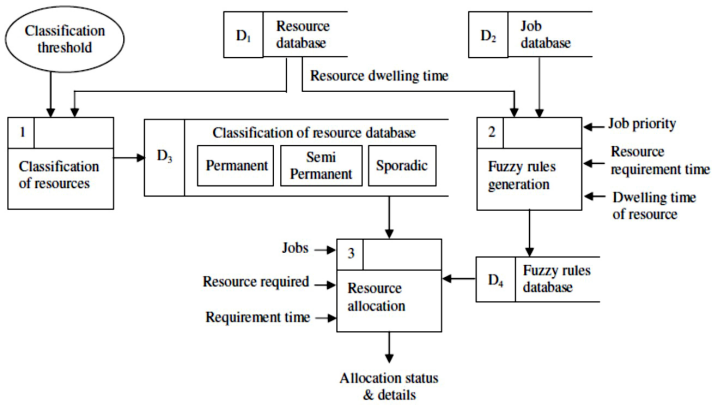


Figure. 2. Flow diagram representing the operation of the proposed resource scheduling technique

3.1 Classification of Resources

In the first stage of the proposed technique, the grid resources are classified based on the dwelling time of the grid resources $R_j^{(t)}$ that can be obtained from the history. Here, the dwelling time is chosen for classification because of the fact that a resource has to be allocated to a high priority job if and only if the reliability of the resource is high i.e. the existence of the resource should be for a long as well as defined period. The classification mainly because of the dwelling time can guide the technique in allocating the resources in an effective way. As a result,$R_j^{(t)}$ is classified into three categories, namely, permanent, semi-permanent and sporadic. The permanent resources are the resources that are available in the grid throughout the period, whereas, the semi-permanent resources are those available in the grid for more than a specific period (i.e. $>p/2$). The sporadic resources are distinct type of resources whose arrival and dwelling time are undefined. However, for simplicity, we utilize the predicted dwelling time for the sporadic resources. The classified resources can be specified as ,$R_{I-j}^{(t)}$, $R_{II-j}^{(t)}$ and $R_{III-j}^{(t)}$ and their dwelling time as , $d_{I-j}^{(t)}$, $d_{II-j}^{(t)}$ and $d_{III-j}^{(t)}$ respectively. The classes,*I*,*II* and *III* , and represents permanent, semi-permanent and sporadic respectively. As discussed earlier, in class *I*, the dwelling time of all the resources remains the same, say p periods (i.e. $d_{I-j}^{(t)} = p$), in class *II*, the dwelling time of the resources are greater than $p/2$ periods but less than p periods i.e. $p/2 < d_{II-j}^{(t)} < p$, and in class *III*, the dwelling time of the resources are within the p i.e. $d_{III-j}^{(t)} \epsilon (0, p)$. Thus, classified resources are maintained in the

database for every p period. The database assists in the allocation of resource.

3.2 Generation of Fuzzy Rule Set

Generating fuzzy rules, the most vital task in the proposed technique decides the type of resources that has to be allocated for the job. The fuzzy rules are generated with three input variables,*x1,x2* , and *x3*, and an output variable y ; and the rules take the form of *if-then* statements. The *x1,x2* , and *x3* are meant for the status of the priority of the jobs, requirement time of a particular resource, which is demanded by the jobs and the dwelling time of the resource, respectively. The output y refers to the score for allocating the resource to the demanding job. Each input variable x_a ; *a=1,2 and 3* , carries three values, which are termed as min, mid and max. So, the priorities, requirement time and dwelling time are classified into min, mid and max. For different combination of the input variable, the fuzzy rules are generated and it is given as follows

S.No	Fuzzy Rules
1	*if* $x_1 = \min$ *and* $x_2 = \min$ *and* $x_3 = \min$, *then* $y = \min$
2	*if* $x_1 = \min$ *and* $x_2 = \min$ *and* $x_3 = mid$, *then* $y = mid$
3	*if* $x_1 = \min$ *and* $x_2 = \min$ *and* $x_3 = \max$, *then* $y = \max$
4	*if* $x_1 = \min$ *and* $x_2 = mid$ *and* $x_3 = \min$, *then* $y = \min$
5	*if* $x_1 = \min$ *and* $x_2 = mid$ *and* $x_3 = mid$, *then* $y = \min$
6	*if* $x_1 = \min$ *and* $x_2 = mid$ *and* $x_3 = \max$, *then* $y = mid$
7	*if* $x_1 = \min$ *and* $x_2 = \max$ *and* $x_3 = \min$, *then* $y = \min$
8	*if* $x_1 = \min$ *and* $x_2 = \max$ *and* $x_3 = mid$, *then* $y = \min$
9	*if* $x_1 = \min$ *and* $x_2 = \max$ *and* $x_3 = \max$, *then* $y = \min$
10	*if* $x_1 = mid$ *and* $x_2 = \min$ *and* $x_3 = \min$, *then* $y = mid$
11	*if* $x_1 = mid$ *and* $x_2 = \min$ *and* $x_3 = mid$, *then* $y = \max$
12	*if* $x_1 = mid$ *and* $x_2 = \min$ *and* $x_3 = \max$, *then* $y = \max$
13	*if* $x_1 = mid$ *and* $x_2 = mid$ *and* $x_3 = \min$, *then* $y = mid$
14	*if* $x_1 = mid$ *and* $x_2 = mid$ *and* $x_3 = mid$, *then* $y = mid$
15	*if* $x_1 = mid$ *and* $x_2 = mid$ *and* $x_3 = \max$, *then* $y = \max$
16	*if* $x_1 = mid$ *and* $x_2 = \max$ *and* $x_3 = \min$, *then* $y = \min$
17	*if* $x_1 = mid$ *and* $x_2 = \max$ *and* $x_3 = mid$, *then* $y = \min$
18	*if* $x_1 = mid$ *and* $x_2 = \max$ *and* $x_3 = \max$, *then* $y = mid$
19	*if* $x_1 = \max$ *and* $x_2 = \min$ *and* $x_3 = \min$, *then* $y = \max$
20	*if* $x_1 = \max$ *and* $x_2 = \min$ *and* $x_3 = mid$, *then* $y = \max$
21	*if* $x_1 = \max$ *and* $x_2 = \min$ *and* $x_3 = \max$, *then* $y = \max$
22	*if* $x_1 = \max$ *and* $x_2 = mid$ *and* $x_3 = \min$, *then* $y = \min$
23	*if* $x_1 = \max$ *and* $x_2 = mid$ *and* $x_3 = mid$, *then* $y = \max$
24	*if* $x_1 = \max$ *and* $x_2 = mid$ *and* $x_3 = \max$, *then* $y = \max$
25	*if* $x_1 = \max$ *and* $x_2 = \max$ *and* $x_3 = \min$, *then* $y = \min$
26	*if* $x_1 = \max$ *and* $x_2 = \max$ *and* $x_3 = mid$, *then* $y = \min$
27	*if* $x_1 = \max$ *and* $x_2 = \max$ *and* $x_3 = \max$, *then* $y = \max$

Figure. 3. Generated fuzzy rules for different combinations of the three input variables

In Fig. 3, the first rule shows that when a job requires a resource for a minimum time (x_2 is min) and the job has high priority (x_1 is min) and the resource has a minimum dwelling time (x_3 is min), then the output says that the possibility for successful completion of the job with that resource is minimum. So, resource allocation is performed next based on the fuzzy output.

3.3 Resource Allocation Based on Fuzzy Rules

After generating fuzzy rules, the resources are allocated based on the fuzzy rules by selecting the type of the resources that are more suitable to the job that demanded the resources. A database is maintained which consists of the submitted jobs J_i, their priority P_i , the required resources R_{kj} and their time of requirement T_{kj} . The database structure that holds the information relevant to the submitted jobs is given in the Fig. 4.

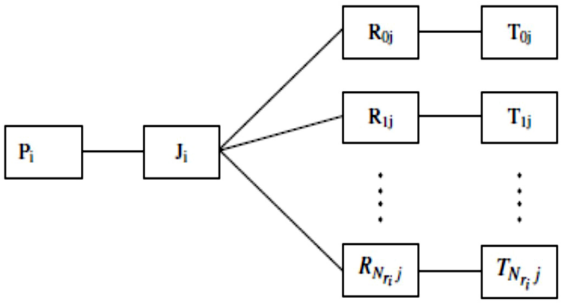


Figure. 4. The structure of the dataset which is comprised of the jobs, its priority, demanded resource and the time of requirement of the resource

As discussed earlier, the resources are classified for every p period and maintained in the database. The database generated at the t^{th} duration is checked to find whether any resources required by the submitted jobs are available in the database or not. In order to check this, a resource set, which is comprised of the resource IDs that are demanded by the submitted jobs are generated as

$$\{R_C\} = \{R_C\} \cup \{R_{kj}\}i \quad (1)$$

Then, each element (i.e. resource) of the $\{R_c\}$ is checked for its availability in the database of classified resources. Once the resource is available, it is allocated based on the fuzzy rules. The pseudo code for allocating the grid resources, which are available at every p period, based on fuzzy logic is presented in Fig. 5.

As given in the pseudo code, every resource in $\{R_c\}$ is taken and the jobs, which demand the resource are identified. Then the resource allocation steps are performed as given in the flow chart that is depicted in Fig. 6. When the i^{th} job is identified for the m^{th} resource (i.e. i^{th} job requires the m^{th} resource), then the class III (i.e. sporadic type) is checked to find whether the m^{th} resource is present in it or not. Once the m^{th} resource of sporadic type is available in the period, the dwelling time of the m^{th} resource is identified from $d^{(t)}_{III-j}$. Then, the priority of the i^{th} job, resource requirement time demanded by the job and the dwelling time of the resource are given as inputs,$x1$,$x2$ and $x3$, respectively to fuzzy logic. From the fuzzy rules, the output y , which is a score to allocate the resource, is obtained. If the obtained score y meets the sporadic class threshold (i.e. $y > S^{th}_{III}$), the resource is allocated to the job and then the following steps are performed.

- Store the resource allocation details, J_i , resource type, resource index R_j , time of requirement and the time t at which the resource is allocated.
- Subtract the time of requirement from the resource dwelling time
- Delete the allocated resource from R_{kj} to avoid re-allocation

In order to store the resource allocation details, ctr is maintained so that ctr can hold the details about the number of resource allocations. In other words, the ctr inclusively indicates the duration of the resource allocation. Once the resource is allocated to the i^{th} job, the same process is repeated by checking the resource for all the N_j jobs. In the mean time, (as given in pseudo code) the index of the job i.e. i is stored in I_{ctr} , resource type is stored in TP_{ctr} , resource index in

```
ctr ← 0
for every m in {R_c}
    for every i in {J}
        if R_c(m) ∈ {R_kj}_i
            x1 ← P_i
            x2 ← T_kj : k ∈ (0, N_ri − 1), j = R_c(m)
            if R_c(m) ∈ R_III−j
                x3 ← d_III−j
                GET y from fuzzy rules
                if y > S_th−III
                    J_ctr ← i
                    R_ctr ← R_c(m)
                    TP_ctr ← III
                    d_III−j ← d_III−j − x2
                    {R_kj}_i ← {R_kj}_i − {R_c(m)}
                    ctr ← ctr + 1
                else
                    if R_c(m) ∈ R_II−j
                        x3 ← d_II−j
                        GET y from fuzzy rules
                        if y > S_th−II
                            J_ctr ← i
                            R_ctr ← R_c(m)
                            TP_ctr ← II
                            d_II−j ← d_II−j − x2
                            {R_kj}_i ← {R_kj}_i − {R_c(m)}
                            ctr ← ctr + 1
                        else
                            if R_c(m) ∈ R_I−j
                                if {T_kj}_i < d_I−j
                                    J_ctr ← i
                                    R_ctr ← R_c(m)
                                    TP_ctr ← I
                                    d_I−j ← d_I−j − x2
                                end if
                            end if
                        end if
                    end if
                end if
            end if
        end if
    end for
end for
```

Figure. 5. Pseudo code for fuzzy rules- based resource allocation

R_{ctr} by replacing the dwelling time with the new one i.e. after allocating the resource for the job, which required for a particular period. But, if the score y does not meet S_{III}^{th} (i.e. $y \leqslant S_{III}^{th}$), then the m^{th} resource is checked for its presence in the semi-permanent type and the aforesaid steps

are followed as already done for the sporadic type. In the semi-permanent type, the fuzzy score y is compared with the threshold S_{II}^{th}. If y satisfies S_{II}^{th} , then the resource is allocated and similar steps (detailed earlier) are yfollowed. If the score doesn't satisfy S_{II}^{th} , the availability of the m^{th} resource is checked in the permanent class. If the m^{th} resource is present in the permanent class, then the dwelling time of the resource is checked to find whether the time of requirement of the resource is greater than the dwelling time of the resource or not. If the time of requirement of the resource is less than $d_{I-j}^{(t)}$, then the resource is allocated to the corresponding job and steps similar to those mentioned above are followed. This is carried out for all the resources in

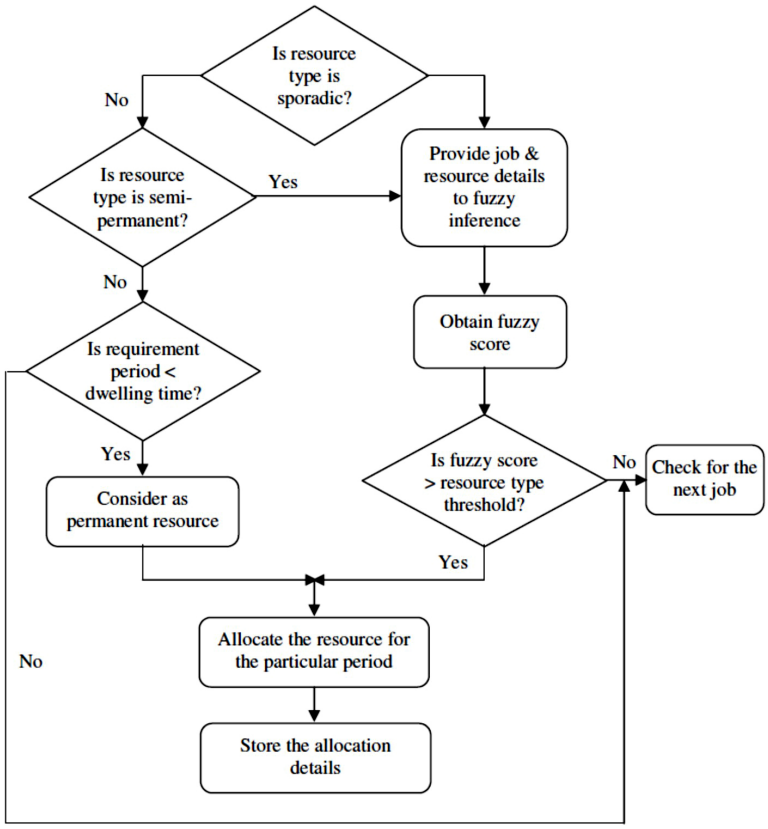


Figure. 6. Flow chart describing the Resource allocation steps carried out for a resource and a job that requires the resource

$\{R_c\}$ and so possible resources are allocated to the jobs which require those resources. The same procedure is repeated for every p period until all the demanded resources are allocated to the submitted jobs.

In this technique, the fuzzy rules are utilized only for allocating the semi-permanent and sporadic resources because of their high dynamic availability. With the aid of fuzzy rules, the risk of allocating those resources to high priority jobs is avoided in an efficient manner. The fuzzy rules are generated in such a way that the semi-permanent and sporadic resources are allocated to high priority jobs only when the dwelling time is considerably greater than the resource requirement time. The decision is based on checking the fuzzy score, which is obtained for the subjected jobs and resource details, with a threshold. Hence, the job execution will not be interrupted because of the unavailability of the allocated resources. The proposed resource allocation technique is evaluated using the performance measures, utilization, failure rate and make span [Farooq et al. 2009; Mehta et al. 2006] that can be determined as follows

$$U = \frac{1}{|t|} \sum_{t=0}^{|t|-1} \frac{N_{alloc_t}}{|R_j^{(t)}|} \tag{2}$$

$$F_R = \frac{1}{|t|} \sum_{t=0}^{|t|-1} 1 - \frac{N_{alloc_t}}{|R_j^{(t)}|} \tag{3}$$

$$M_S = (|t|-1)p + max(T_{kj}^{(|t|-1)}) \tag{4}$$

From the aforesaid measures, the performance of the proposed resource allocation technique is determined and its efficacy in allocating the resources can be understood well.

4. RESULTS AND DISCUSSION

The proposed fuzzy-based resource allocation technique has been implemented in the working platform of MATLAB (version 7.8). For convenient analysis, the proposed resource allocation technique has been evaluated by submitting the $N_j = 10$ jobs with its priority ranges from 1 to 10 (i.e. $P_{min} = 1$ and $P_{max} = 10$). The jobs with its priority levels, the resources that are demanded by the jobs and the time of requirement are given in the Table I. As we have already known, the resources are classified into three categories based on the dwelling time. For implementation simplicity, we have generated the resources of three classes arbitrarily for every t^{th} duration with $p = 15\ sec$. So, for every t^{th} duration, resources of three classes have been generated arbitrarily that makes the realistic grid environment. The generated resources of three different classes with their dwelling time are given in the Table II. The proposed resource allocation technique looks for suitable resources for the submitted jobs at an interval of t . Based on the fuzzy rules, the resources have been allocated to all the submitted jobs and it is given in Table III. To evaluate the performance of the proposed resource allocation technique, the performance metrics, utilization, failure rate and makespan have been determined for different job sets and the resultant metrics are given in Table IV.

Table I. A job set with 5 jobs, their priority, demanded resources and requirement time of the resources

Job	Priority	Demanded Resources	Time of requirement (sec)
J_1	1	R_1	5
		R_2	4
		R_3	2
J_2	2	R_3	2
		R_4	8
		R_5	9
		R_7	10
J_3	3	R_4	8
		R_2	10
		R_3	11
		R_1	13
J_4	4	R_8	5
		R_9	4
		R_3	2
J_5	5	R_2	13
		R_{11}	5
		R_{12}	14
		R_{13}	12
		R_{10}	11
		R_{19}	13

In tables I, II and III, only a sample of jobs and resources are tabulated. In table IV, the performance metrics is given for some five different job sets and represented in Fig 7. But for detail analysis, the proposed technique was implemented in three different synthetic datasets. The synthetic dataset was generated with 50 resources in each class and the job set was comprised of 30 jobs. In the job set, the required resources and the resource requirement period for every job were arbitrarily generated and the performance of the technique was evaluated. The performance of the technique was compared with the conventional Genetic algorithm (GA)-based resource scheduling technique, Particle Swarm Optimization (PSO) - based resource scheduling technique and Simulated Annealing (SA) - based resource scheduling technique. The results obtained for the four different techniques for the three different synthetic datasets are given in Table V. The comparison between the techniques is illustrated in Fig. 8.

When tested with different datasets, GA-based resource allocation algorithm had a very little improved performance over the other allocation techniques; however this happened only for

Table II. A sample of resources under the three classes, permanent, semi-permanent and sporadic and the dwelling time of the resources

S.No	Permanent		Semi permanent		Sporadic	
	Resource	Dwelling time (sec)	Resource	Dwelling time (sec)	Resource	Dwelling time (sec)
1	R_{15}	15	R_{16}	11	R_6	5
2	R_7	15	R_{20}	10	R_{18}	6
3	R_2	15	R_{17}	12	R_{16}	10
4	R_{17}	15	R_{10}	11	R_{12}	15
5	R_8	15	R_{18}	15	R_8	13
6	R_{14}	15	R_7	15	R_7	14
7	R_{18}	15	R_5	10	R_{13}	15
8	R_6	15	R_2	13	R_3	9
9	R_{20}	15	R_6	10	R_5	8
10	R_{16}	15	R_{14}	12	R_{15}	13
11	R_{13}	15	R_{13}	13	R_{14}	7
12	R_{11}	15	R_{19}	14	R_{17}	15
13	R_{12}	15	R_{12}	13	R_{11}	14

Table III. The jobs, demanded resource, the class of the resource which is allocated to the job and the duration at which the resource is allocated

Job	Demanded Resource	Class of allocated resource	Duration of allocation (t)
J_1	R_2	Semi-permanent	2
	R_3	Sporadic	2
J_2	R_1	Semi-permanent	2
	R_3	Sporadic	2
	R_4	Semi-permanent	1
	R_7	Permanent	2
	R_5	Semi-permanent	2
J_3	R_1	Permanent	2
	R_2	Permanent	2
	R_4	Semi-permanent	2
	R_3	Semi-permanent	2
J_4	R_1	Permanent	5
	R_3	Sporadic	1
	R_8	Sporadic	2
	R_9	Semi-permanent	1
	R_{11}	Semi-permanent	1
J_5	R_{14}	Permanent	2
	R_{10}	Permanent	2
	R_{11}	Semi-permanent	2
	R_{12}	Semi-permanent	2

Table IV. The performance metrics, utilization, failure rate and makespan for four different job sets

Job set	Utilization U (in %)	Failure rate F_R (in %)	Make span M_S
1	80.68	19.31	75
2	74.19	25.08	73
3	79.64	20.35	115
4	77.70	22.22	88
5	73.42	26.58	94

the dataset 3 and not for the remaining two. The proposed technique almost competes with the other resource allocation techniques in utilization as well as in failure rate. Though, the proposed technique achieves higher makespan rather than GA-based technique for the datasets

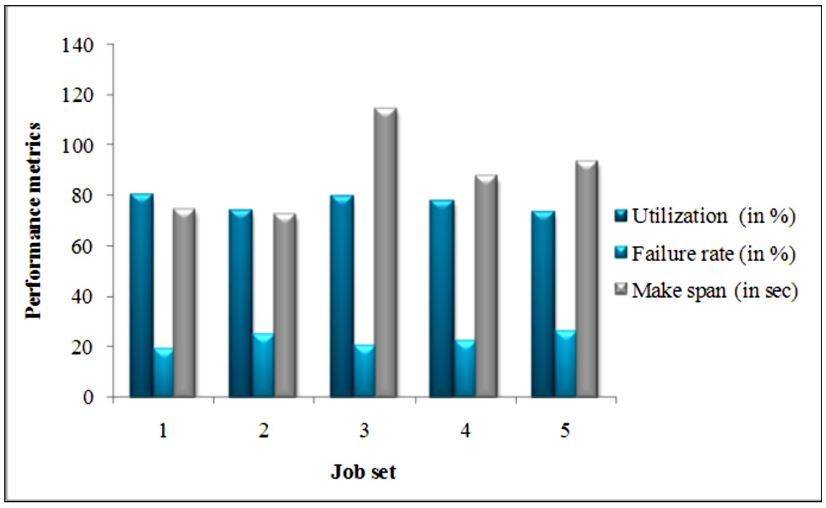


Figure. 7. Representation of performance metrics for a sample dataset

Table V. Performance comparison of the proposed resource allocation technique with GA-based resource allocation, PSO-based resource allocation and SA-based resource allocation techniques for (i) Synthetic dataset 1, (ii) Synthetic dataset 2 and (iii) Synthetic dataset 3

(i)

Performance Metrics	Proposed Resource Allocation Technique	GA-based Resource Allocation Technique	PSO-based Resource Allocation Technique	SA-based Resource Allocation Technique
Minimum Utilization (in%)	61.13	37.45	25.31	34.45
Minimum Failure rate (in%)	22.43	51.65	62.62	56.21
Minimum Makespan (in sec)	79	70	85	74
Maximum Utilization (in %)	77.56	48.34	37.34	43.79
Maximum Failure rate (in %)	38.84	62.51	74.69	65.5
Maximum Makespan (in sec)	120	98	157	156
Mean Utilization (in %)	65.33	44.65	33.21	39.65
Mean Failure rate (in %)	34.65	55.34	66.76	60.31
Mean Makespan (in sec)	109	87	148	127

(ii)

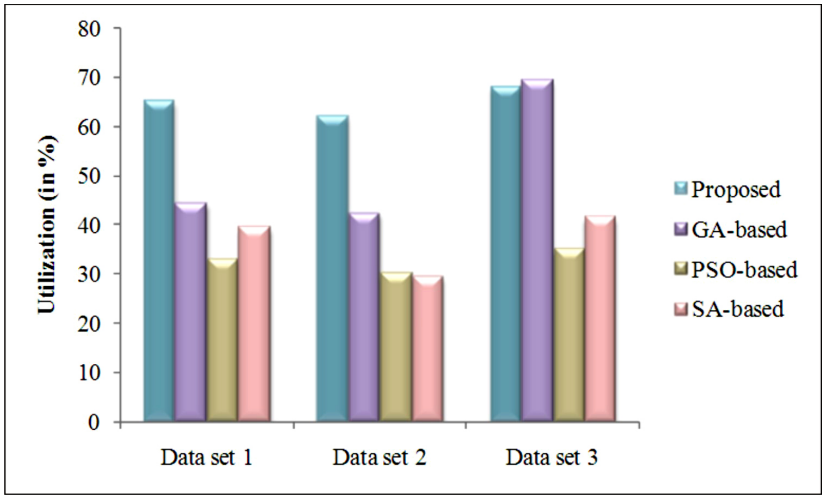
Performance Metrics	Proposed Resource Allocation Technique	GA-based Resource Allocation Technique	PSO-based Resource Allocation Technique	SA-based Resource Allocation Technique
Minimum Utilization (in%)	51.11	26.42	13.32	22.42
Minimum Failure rate (in%)	32.42	59.65	71.64	68.20
Minimum Makespan (in sec)	70	60	72	61
Maximum Utilization (in %)	67.56	40.34	28.34	31.79
Maximum Failure rate (in %)	48.81	73.49	86.62	77.53
Maximum Makespan (in sec)	109	90	137	143
Mean Utilization (in %)	62.3	42.65	30.21	29.65
Mean Failure rate (in %)	31.65	52.34	62.76	70.31
Mean Makespan (in sec)	106	85	126	124

Figure. 1: Representation of performance metrics for a sample dataset

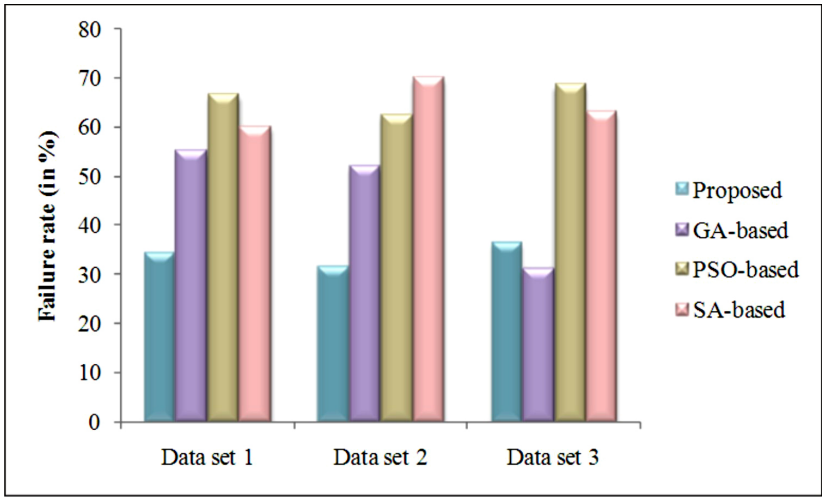
1 and 2, a little minimum makespan is achieved for the dataset 3. As already stated, when GA-based technique performs well in utilization and failure rate, its performance in makespan is poor

(iii)

Performance Metrics	Proposed Resource Allocation Technique	GA-based Resource Allocation Technique	PSO-based Resource Allocation Technique	SA-based Resource Allocation Technique
Minimum Utilization (in%)	70.13	45.45	34.31	45.45
Minimum Failure rate (in%)	24.49	48.63	54.65	49.20
Minimum Makespan (in sec)	88	87	93	81
Maximum Utilization (in %)	75.01	51.34	45.34	50.79
Maximum Failure rate (in %)	29.84	54.51	65.67	54.52
Maximum Makespan (in sec)	129	105	164	166
Mean Utilization (in %)	68.33	46.65	35.21	41.65
Mean Failure rate (in %)	36.65	58.34	68.76	63.31
Mean Makespan (in sec)	112	113	150	131

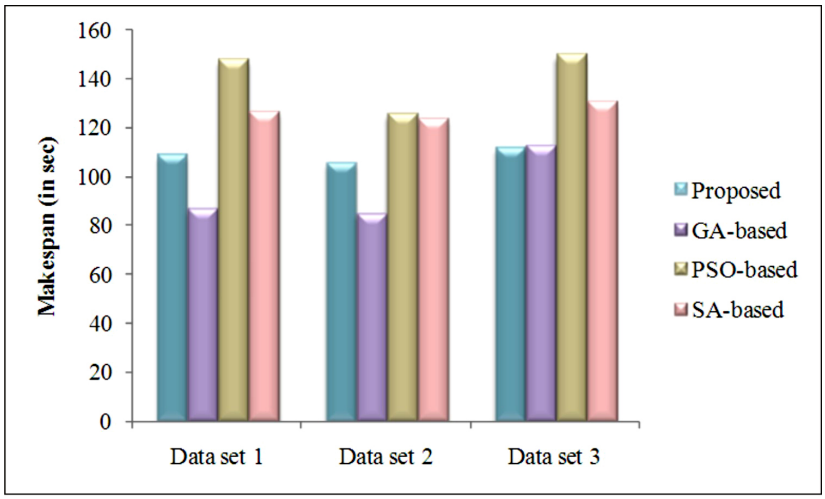


(i)



(ii)

when compared to the proposed technique. This indirectly indicates that the proposed technique achieves a satisfactory performance in all the three performance metrics when compared to all the three aforesaid conventional techniques.



(iii)

Fig. 8. Comparison graph between the proposed technique, GA-based technique, PSO-based technique and SA-based technique over the three different synthetic datasets in terms of (i) Utilization, (ii) Failure rate and (iii) Makespan

5. CONCLUSION

We have proposed a simple, but proficient fuzzy-based resource allocation technique for the grid environment. The simulation results have shown that the proposed technique effectively allocates the grid resources to the submitted jobs. It has been seen that for every period, the technique looks for a possible and suitable resource for allocation to the job. If the resource is available, then it will be allocated, otherwise the job has to wait for the next period for the arrival of a suitable resource. We have calculated the three performance measures utilization, failure rate and makespan for evaluating the performance of the proposed technique. The measurements have shown that the proposed technique can allocate the resources effectively to all the submitted jobs at the right time. Moreover, allocation of resources, which has undefined dwelling time, to high priority jobs is avoided. This leads to the prevention of the risk of interruption of the job execution because of the lack of the demanded resource. Thus, the technique analyzes the resources well to allocate it for a particular resource. This has been accomplished mainly because of the fuzzy rules and so the proposed fuzzy-based technique is more proficient in allocating the dynamic grid resources to the submitted jobs.

REFERENCES

J. BLYTHE, E. DEELMAN, Y. GIL,C. KESSELMAN ,A. AGARWAL G. MEHTA AND K. VAHI 2003. The Role of Planning in Grid Computing. *13th International Conference on Automated Planning and Scheduling (ICAPS), Trento, Italy,* June 2003.

C. CHAPMAN,M. MUSOLESI ,W. EMMERICH AND C. MASCOLO, 2007. Predictive Resource Scheduling in Computational Grids. *IEEE International Parallel and Distributed Processing Symposium, Long Beach, CA,* 2007, 1-10.

J. CHEN, J. PENG ,ANDX. CAO2009. A Grid Resource Scheduling Algorithm Based on *the Utility Optimization. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering,* 5(1), 1355-1362. CHEN, J., AND LU, B., 2008. Load Balancing Oriented Economic Grid Resource Scheduling. Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 19-20 Dec. 2008, 813-817.

AJ. CHEN ,AND, B. LU 2008. Load Balancing Oriented Economic Grid Resource Scheduling. *Pacific-Asia Workshop on Computational Intelligence and Industrial Application,* 19-20 Dec. 2008, 813-817.

L. CHUNLIN, 2008. Joint Application-Fabric Layer Optimization in Grid Computing. *11th IEEE International Conference on Computational Science and Engineering, Sao Paulo,* 16-18 July 2008, 141 - 146.

P.A, DINDA2001. Online prediction of the running time of tasks. *In Proc. 10th IEEE Symp. on High Performance Distributed Computing,* 2001

U. FAROOQ, S., MAJUMDARA AND E.W. PARSONSA 2009. chieving efficiency, quality of service and robustness in multi-organizational Grids. Journal of Systems and Software, 82(1), 23-38.

I. FOSTER AND C. KESSELMAN 1986. Computational grids. The Grid: Blueprint for a New Computing Infrastructure, pages 15-52, Morgan Kaufmann, San Francisco, California, 1986.

A. GALSTYAN , K. CZAJKOWSKI AND K. LERMAN 2004. Resource Allocation in the Grid Using Reinforcement Learning. *In Proc. Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 2004.

J. GOMOLUCH AND M. SCHROEDER 2003. Market-based Resource Allocation for Grid Computing: A Model and Simulation. Middleware Workshops, 2003, 211-218.

Y. HAO , Y. XU ,G. LIU AND Z. PAN 2008. An Expectation Trust Benefit Driven Algorithm for Resource Scheduling in Grid Computing. *3rd International Conference on Innovative Computing Information and Control, Dalian, Liaoning China*, 18-20 June 2008, 88 - 88.

W. JIANG , H. CUI, AND J. CHEN 2009. A fuzzy modeling based dynamic resource allocation strategy in service grid. KAUR, G., AND CHOPRA, I., 2007. *Grid Computing- Challenges Confronted and Opportunities Offered. Proceedings of COIT*, 2007.

R. E.,KALMAN1960. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME Journal of Basic Engineering*, 82(D), 35-45.

N. KIRAN , V. MAHESWARAN ,M. SHYAM AND P. NARAYANASAMY 2007. A novel Task Replica based Resource Scheduling Algorithm in Grid computing. *International Conference on High Performance Computing*, 2007.

S. KRAWCZYK , AND K. BUBENDORFER 2008. Grid Resource Allocation: Allocation Mechanisms and Utilisation Patterns. *Conferences in Research and Practice in Information Technology Series; In Proc. Sixth Australasian workshop on Grid computing and e-research*, 2008, 73-81.

K. KUROWSKI A. OLEKSIAK,J. NABRZYSKI,A. KWIECIE,M. WOJTKIEWICZ,M. DYCZKOWSKI,F. GUIM,J. CORBALAN AND J. LABARTA 2007. Multi-Criteria Grid Resource Management Using Performance Prediction Techniques. *Integrated Research in GRID Computing*, 215-225.

J. LI, W. ZIEGLER , O. WALDRICH, AND , D. MALLMANN 2008. Towards SLA Based Software License Management in Grid Computing.CoreGRID Technical Report, Number TR-0136, 2008.

S.S MANVI, M.N. BIRJE AND B. PRASAD 2005. An Agent-based Resource Allocation Model for computational grids. *Multiagent and Grid Systems - An International Journal 1(1)*, 17-27.

A.M. MEHTA, J. SMITH ,H.J. SIEGEL ,A.A. MACIEJEWSKI AND A. JAYASEELAN 2006. Dynamic Resource Management Heuristics for Minimizing Makespan while Maintaining an Acceptable Level of Robustness in an Uncertain Environment. *12th International Conference on Parallel and Distributed Systems, Minneapolis, Minnesota*, July 12-15, 2006.

S. MURUGANANTHAM,P.K. SRIVASTHA AND KHANAA 2010. Object Based Middleware for Grid Computing. Journal of Computer Science, 6(3), 336-340.

M. NAQAASH M.A. IQBAL,Q.PERVAIZ,O. SHAMIM AND S. A. HUSSAIN 2010. Grid Computing Used For Next Generation High Speed Processing Technology. *International Journal on Computer Science and Engineering*, 02(05), 1926-1933. PLAXTON, C.G., SUN, Y., TIWARI, M., AND VIN, H., 2006. Reconfigurable Resource Scheduling. Proceedings of the 18th Annual ACM Symposium on Parallel Algorithms and Architectures, Cambridge, Massachusetts, USA, July 30 - August 2, 2006.

C.G. PLAXTON , Y. SUN,M. TIWARI AND H. VIN 2006. Reconfigurable Resource Scheduling. In *Proceedings of the 18th Annual ACM Symposium on Parallel Algorithms and Architectures, Cambridge, Massachusetts, USA*, July 30 - August 2, 2006.

L. POURMOHAMMADBAGHER2008. Intelligent Agent System Simulation Using Fear Emotion. *World Academy of Science, Engineering and Technology*, 48, 334-338.

M. RAMIREZ-GONZALEZ AND O.P. 2009. Simplified Fuzzy Logic Controller and its Application as a Power System Stabilizer. *15th International Conference on Intelligent System Applications to Power Systems*, 8-12 Nov. 2009, 1-6.

T. ROBLITZ,F. SCHINTKE ,AND A. REINEFELD2006. Resource reservations with fuzzy requests. *Concurrency and Computation: Practice and Experience*, 18(13), 1681-1703.

W. SILER AND J.J. BUCKLEY 2005. Fuzzy expert systems and fuzzy reasoning. John Wiley and Sons, Inc., Hoboken, New Jersey, 2005.

H.C. TSENG2007. Internet Applications with Fuzzy Logic and Neural Networks: A Survey. *Journal of Engineering, Computing and Architecture*, 1(2).

V. VIJAYAKUMARAND R.S.D. WAHIDA BANU2008. Security for Resource Selection in Grid Computing Based on Trust and Reputation Responsiveness. *IJCSNS International Journal of Computer Science and Network Security*, 8(11), 107-115.

R. WANKAR2008. Grid Computing With Globus: An Overview and Research Challenges. *International Journal of Computer Science and Applications*, 5(3), 56 - 69.

R. WOLSKI,N. SPRING ,AND J. HAYES1999. Predicting the CPU availability of time-shared unix systems on the computational grid. In *proceedings of High Performance Distributed Computing, Redondo Beach, CA*, 1999,105 - 112.

R. WOLSKI,N. SPRING ,AND J. HAYES1999. The network weather service: A distributed Resource performance forecasting service for metacomputing. *Future Generation Computer Systems*, 15(5 -6), 757-768.

F. XIA,W. ZHAO ,Y. SUN AND Y.C. TIAN2007. Fuzzy Logic Control Based QoS Management in Wireless Sensor/Actuator Networks. Sensors, 7(12), 3179-3191.

L.A. ZADEH2008. Is There a Need for Fuzzy Logic?. *Information Sciences: an International Journal*, 178(13), 2751-2779. ZHANG, J., AND PHILLIPS, C., 2009. Job-Scheduling with Resource Availability Prediction for Volunteer-Based Grid Computing. London Communications Symposium, LCS 2009, University College London, September 2009.

J. ZHANG,AND C. PHILLIPS2009. Job-Scheduling with Resource Availability Prediction for Volunteer-Based Grid Computing. *London Communications Symposium, LCS 2009, University College London*, September 2009.

M.Poonguzhali acquired a B.E. degree in Electronics and Communication Engineering from Government College of Technology, Coimbatore, in the year 1997, and an M.E. degree in Applied Electronics from the same institute in 1999. She is currently pursuing the Ph.D. in the Department of Electronics and Communication Engineering, College of Engineering, Anna University, Chennai, India. Now she is working as Assistant Professor, in Narasus Sarathy Institute of Technology. Her areas of interest include Grid computing, mobile ad hoc networks, high-speed networks, and digital communication. (Email: poonguzhaliphd@gmail.com, kuzhali76@gmail.com)

Dr. S. Shanmugavel graduated from Madras Institute of Technology in electronics and communication engineering in 1978. He obtained his Ph.D. degree in the area of coded communication and spreadspectrum techniques from the Indian Institute of Technology (IIT), Kharagpur, in 1989. He joined the faculty of the Department of Electronics and Communication Engineering at IIT, Kharagpur, as a Lecturer in 1987 and became an Assistant Professor in 1991. Presently, he is a Professor in the Department of Electronics and Communication Engineering, College of Engineering, Anna University, Chennai, India. He has published more than 68 research papers in national and international conferences and 25 research papers in journals. He has been awarded the IETE-CDIL Award in September 2000 for his research paper. His areas of interest include Grid computing, mobile ad hoc networks, ATM networks, and CDMA engineering. (Email: ssvel@annauniv.edu)