Survey on Scheduling Algorithms for Grid Computing

Anuj Kumar

Research Scholar, Gurukul Kangri Vishwavidyalaya, Haridwar (UK), India

Heman Pathak

Associate Professor , Gurukul Kangri Vishwavidyalaya, Haridwar (UK), India

ABSTRACT-Fast expansion in wide area network facilitating the availability of low cost computational resources, resource sharing, scalability etc. have taken communication to the era of Grid Computing wherein the desktop computers may participate in a Global network activity while being idle thereby enabling large software systems to utilize extra hardware resources. Scheduling and Load Balancing, two complementary processes, play a vital role in Grid Computing. A number of issues restrict the traditional schemes, meant for devising a load balanced scheduling in distributed systems, to be applicable on Grids. Over the years, although the research community has contributed a number of approaches for efficient scheduling in Grids yet there is a need of more and more efficient fault-tolerant approaches that may ensure effective utilization of resources along with ensuring the load balancing in grids. This paper presents a survey of the various scheduling approaches proposed for Grids so far with a special emphasis over computational grids. Further, on the basis of the central idea and the working principles of the various approaches witnessed in the literature, taxonomy of scheduling approaches for grids is proposed along with ascertaining the chronological research trends so as to facilitate the researchers in the domain of Grid Computing.

KEYWORDS- *Grid Computing, Computational Grid, Taxonomy, Scheduling Algorithms, Research Trends, Grid Scheduling, Review, Survey*.

1.INTRODUCTION

Grid computing is an extension of Cluster Computing that incorporates coordinating and sharing of computational power, data storage and network resources across dynamic and geographically dispersed organizations. Several factors like resource sharing, scalability, etc. have taken communication to the era of Grid computing. This permits desktop computers to take part in a global network activity when they are idle, and it enables large software systems to utilize extra hardware resources. When all the resources of inactive computer systems are gathered as an all-in-one computer system, a highly effective system arises in the form of Grid Computing System [1].

Grid Computing System use computers which are part of the grid only when idle and operators can perform tasks unrelated to the grid at any time. Security must be considered when using computer grids as controls on member nodes are usually very loose. Redundancy should also be built in as many computers may disconnect or fail during processing [2].

Computational Grids denote systems that have a higher aggregate computational capacity available for single applications than the capacity of any constituent machine in the system. A Computational Grid is a hardware and software infrastructure that provides a dependable, consistent, pervasive and inexpensive access to high end computational capabilities. It works on multi-user environment that offers discontinuous demands of huge information processing [3]. Computational Grids can be further subdivided into distributed supercomputing and high throughput categories depending on how the aggregate capacity is utilized. A distributed supercomputing Grid executes the application in parallel on multiple machines to reduce the completion time of a job.

Scheduling and Load Balancing in distributed systems are closely related to each other. While the prior one is responsible for deciding the execution order of the tasks, the later one is responsible for ensuring that all the processing elements of distributed system are fairly loaded. Load balanced task scheduling is very important problem in complex grid environment. So task scheduling which is one of the NP-Complete problems becomes a focus of research in grid computing area. Scheduling Algorithms are implemented at Grid Task Scheduler and is responsible for allocating tasks to resources under grid environment for execution [4]. In computational grid, scheduling problem is enhanced by minimizing makespan, maximizing system utilization, balancing the loads, and fulfilling economical system demand under user specific constraints [5].

The schematic for scheduling in grid systems is shown in Figure 1. As shown in the figure, the jobs that arrive from grid users are first placed in a job queue. Thereafter, as per the requirement of the load balancing schemes the job scheduler schedules the jobs from job queue to the dispatch queues of the appropriate computing node. The computing node executes the jobs and sends the computational results back to the associated grid user. Several parameters are used to measure the performance of the scheduling and load balancing algorithms in computational grids such as resource utilization, response time, throughput, waiting time, and reliability, communication overheads, processing cost etc. [6].

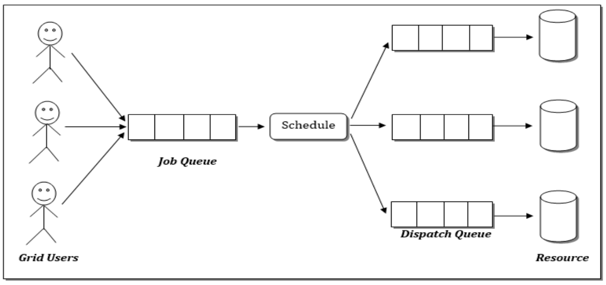

Figure 1 – Schematic for Scheduling in Grid Systems

Rest of the paper is organized as follows: Section 2 briefly presents the major issues and challenges relevant for resource management in grid computing system. Section 3 presents a review on various types of scheduling policies in grids. Section 4 covers a survey on the various scheduling approaches proposed by the researchers for grid computing systems with a special emphasis over computational grids and propose taxonomy for the same followed by an analysis of the research trends in this domain. Section 5 concludes the paper along with future directions for research.

2.MAJOR ISSUES & CHALLENGES

A distributed system adopts various policies for the resources and their usage viz. load balancing, scheduling, and fault tolerance. Although a Grid belongs to the class of distributed systems yet the traditional policies of the distributed systems can not directly apply as such into a Grid. Computational grids have the potential for solving large-scale scientific problems using heterogeneous and geographically distributed resources. However, a number of major technical hurdles must be overcome before this goal can be fully realized. Further, due to the distribution of a large number of resources in a Grid environment and the size of the data to be moved among them, the traditional distributed approaches do not provide accurate results in a Grid.

Effective resource management ensures optimal resource utilization and system throughput and thereby enhances the performance of the grid system. Hence, the scheduling and load balancing techniques should be “fair” in distributing the load across the grid nodes. “Fair” means that the difference between the “heaviest loaded” node and “lightest loaded” node should be the minimum [7].

With the rapid advancement in technology, the numbers of users are also increasing simultaneously to access resources in the heterogeneous and dynamic environment supported by grids. In a real world scenario, the job arrival patterns are volatile and the computing capabilities are unpredictable and asymmetrical. The few nodes in a particular grid site may become overloaded while other nodes on the grid sites may be under-loaded. Therefore, the heterogeneous and the dynamic environment of grids require effective scheduling and load balancing in order to make the best usage of the performance of the grid nodes [8 - 9].

The following issues make the resource management more difficult and challenging task in a Grid Computing System [10 - 18].

2.1 Heterogeneity

The heterogeneity exists in both the computational and networks resources. Firstly, the networks used in Grids may differ significantly in terms of their bandwidth and communication protocols. Secondly, computational resources are usually heterogeneous. Because resources may have different hardware such as instruction set, processors, CPU speed, memory size and different software like operating systems, file systems and so on.

2.2 Autonomy

Typically a Grid may comprise of multiple administrative domains. Each domain shares a common security and management policy. Each domain usually authorizes a group of users to use the resources in the domain. Thus, the application from non-authorized users should not eligible to run on the resources in some specific domains. Because, the multiple administrative domains share Grid resources, a site is viewed as an autonomous computational entity. It usually has its own scheduling policy, which complicates the task allocation problem. A single overall performance goal is not feasible for a Grid system since each site has its own performance goal and the scheduling decision is made independently of other sites according to its own performances.

2.3 Scalability

A Grid may grow from a few resources to millions. This raises the problem of potential performance degradation as the size of a Grid increases.

2.4 Efficient Resource Management

It refers to efficiently manage the different types of resources like bandwidth, processing power, etc. so that they can be efficiently utilized and satisfy the need of the users. It is one of the fundamental requirements in grid computing where resources are quite limited and need to be utilized properly.

2.5 Stagnation

Stagnation is one of the complicated issues in Grid computing systems which occur when a large number of submitted tasks are assigned to a specific resource and make it overflow thereby causing a load imbalance in the grid system.

2.6 Fault Tolerance & Adaptability

The Grids are characterised by geographically dispersed scalable architecture with multiple administrative domain which is prone to failures either by the failure of nodes or the failure of links between them. In either of these cases, the efficacy of the system is determined by the resource management equipped with efficient mechanisms for failure detection and recovery so the resource managers has to be adaptive and must tailor their behaviour dynamically so that they can extract the maximum performance from the available resources and services.

2.7 Dynamic Behaviour

The pool of resources can be assumed to be fixed or stable in the traditional parallel and distributed computing environments while in Grids both the networks and computational resources exhibit dynamicity. First, a network shared by many execution domains like internet may not provide guaranteed bandwidth. Second, both the availability and capability of computational resources may exhibit dynamic behaviour as sometimes new resources may join the Grid and sometimes some of the existing resources may become unavailable due to problems like network failure. The resource managers must tailor their behaviours dynamically so that they can extract the maximum performance from the available resources and services.

2.8 Application Diversity

The Grid applications involve a wide range of users, each having its own special requirements. For example, some applications may require sequential executions, some may consist of a set of independent jobs and other may consist of a set of dependent jobs. In this context, building a general purpose load balancing system seems extremely difficult.

2.9 Resource Non-Dedication

The resource usage contention appears as a major issue due to the non-dedication of resources resulting into inconsistency of behaviour and performance and posing a challenge for designing an accurate load balancing model. The resources that are united in grid are geographically distributed and different individuals or organizations own each of them. Additionally they have their own access policies, processing cost, and mechanism. The resource administrators are responsible to manage and control using their desired management and scheduling system.

2.10 Resource Selection and Computation-Data Separation

Unlike the traditional systems, where the executable codes of applications and input/output data are either on the same site or the input sources and output destinations are determined before the submission of an application, in a Grid, the computation sites of an application are selected by the Grid scheduler according to resource status and some performance criterion. Further, the communication bandwidth of the underlying network is limited and is shared by a host of background loads, so the communication costs may not be neglected. This situation brings about the computation-data separation problem and thereby compels to select a computational resource that can provide the low computational cost by neutralizing its high access cost to the storage site.

2.11 Job & Data Migration

Computational grids have the potential for solving large-scale scientific problems using heterogeneous and geographically distributed resources. In view of the issues caused by computation – data separation discussed above, one problem critical to the effective utilization of computational grids is efficient job scheduling. This challenge is addressed by defining grid scheduling architecture and several job / data migration strategies.

2.12 Processing Cost

The processing cost also poses a challenge in grids. As the resources belong to different individuals / organizations and are geographically dispersed there is always a variation in costs for various framework clients for their asset utilization which varies from time to time.

The aforesaid challenges put significant obstacles to the problem of designing an efficient and effective load balancing system for the Grid environments. Some of the problems caused by these factors have not yet been solved successfully and still remains as an open research issue. Thus, it is challenging to design a load balancing system for the Grid environment [19].

3. TYPES OF SCHEDULING

Different scheduling approaches for distributed systems, reported in the literature, can be classified as follows [20 - 22], although there exist an overlapping and they are not clearly distinct of each other:

3.1 Static vs. Dynamic Scheduling

In static scheduling, jobs are pre-scheduled and all the information about available resources and tasks in application must be known and a task is assigned once to a resource, so that it's easier to adapt based on scheduler's perspective. On the other hand, dynamic scheduling is more flexible than static scheduling where jobs are dynamically available for scheduling over time by the scheduler.

3.2 Centralized Vs. Decentralized Scheduling

A centralized scheduler is responsible for making global decision. It offers a number of benefits like ease of implementation, efficiency, more control and monitoring on resources. Its drawbacks include lack of scalability, fault tolerance and efficient performance so it is not recommended for large-scale grids. On the other hand, Decentralized Scheduling or Distributed Scheduling is more realistic for real grids despite of being inefficient as compared to centralized scheduling.

3.3 Co-operative vs. Non-Co-operative Scheduling

In cooperative scheduling, system have schedulers, each one of which is responsible for performing certain activity in scheduling process towards common system wide range based on the cooperation of procedures, given rules and current system users.

3.4 Preemptive vs. Non Preemptive Scheduling

Preemptive scheduling allows each job to be interrupted during execution and a job can be migrated to another resource leaving its originally allocated resource unused to be available for other jobs. It is more helpful if there are constraints as priority to be considered. On the other hand, in Non Preemptive Scheduling resources aren't allowed to be re-allocated until the running and scheduled job finished its execution.

3.5 Immediate vs. Batch Scheduling

In Immediate Scheduling, scheduler schedules any recently arriving job as soon as it arrives with no waiting for next time interval on available resources at that moment. It is also called Online Mode Scheduling. On the other hand, in Batch Scheduling the scheduler holds arriving jobs as group of problems to be solved over successive time intervals, so that it is better to map a job for suitable resources depending on its characteristics. It is also called Offline Mode Scheduling.

4. TAXONOMY OF SCHEDULING APPROACHES

Taxonomy of the various scheduling approaches available in literature for grid environment with special emphasis over computational grids on the basis of the central idea and working principle behind the approach is proposed as shown in the Table 1. An overview to the various approaches summarized therein is as follows:

4.1 ABSTRACT APPROACHES

This class comprise of the most basic approaches on scheduling as follows:

4.1.1 First Come First Serve (FCFS)

It is an abstracted way of organizing and allocating of resources to jobs over time, it serves as a principle of a queue processing or demands' servicing by ordering that means what comes in first is allocated first, what comes in next waits until the first is finished [23]. FCFS is implemented through parallel processing aiming at tuning resource allocation time with the selected task from the incoming tasks in [24]. A variant of FCFS entitled Opportunistic Load Balancing (OLB) or myopic algorithm in proposed in which works on assigning each task in a queue, in arbitrary order, to the next expected resource to be available, irrespective of the task's expected execution time using resources [25-29]. It is obviously clear that (OLB) works on keeping all machines as busy as possible. FCFS is generally recommended for Space Shared scheduling mechanism in a distributed multiprocessor environment.

4.1.2 Round Robin (RR)

Round-robin (RR) is a simple scheduling algorithm, based on time sharing among jobs in equal slice / quantum and in circular queue without priority so it is simple and easy to implement, but it has a starvation problem so it focuses on fairness between jobs [30-31]. The advantage of RR over FCFS and other similar mechanisms is that no job has to wait for another one to be completed. However, this algorithm is not a good choice for jobs characterized by large variation in size and requirements as it leads to a situation where a job is never been satisfied and in turn leads to starvation or indefinite blocking. Round Robin is generally recommended for Time Shared Scheduling mechanisms in distributed uni-processor environment.

4.1.3 Join the Shortest Queue (JSQ)

JSQ algorithm does not require the task length for its scheduling decision and assigns each task to the resource with the minimum number of waiting tasks in its queue. It only needs the real time number of tasks in each resource queue which is collected from all of the resources at the time of each scheduling decision [32].

4.2 HEURISTIC BASED APPROACHES

In view of the fact that problem of computing the optimal schedule belongs to the class of NP-Complete problems, various heuristics to devise the near – optimal schedule are commonly used by the research community. These are discussed as follows:

4.2.1 Minimum Execution Time (MET)

In contrast to OLB, Minimum Execution Time (MET) arbitrarily assigns each task to the machine with the best expected execution time for that task, regardless of that machine's availability [33-34]. The motivation behind MET is to give each task to its best machine. This can cause a severe load imbalance across machines. In general, this heuristic is obviously not applicable to heterogeneous computing environments characterized by consistent ETC matrices.

4.2.2 Minimum Completion Time (MCT)

Minimum Completion Time (MCT) arbitrarily assigns each task to the machine with the minimum expected completion time for that task [33]. This causes some of the tasks to be assigned to the machines that are not recommended in view of minimum execution time for them. The motive of MCT is to combine the positives of OLB and MET, while avoiding the circumstances in which OLB and MET perform poorly.

4.2.3 K-Percent Best (KPB)

KPB uses the same approach as the MCT algorithm with the only difference that it only examines a subset of resources instead of searching for the minimum completion time among all resources and hence the communication costs are considerably reduced. This subset consists of a percentage (KM/100) of all the resources with the smallest execution times for the incoming task, where $(100/M) \leq K \leq 100$, choice of K determines the effectiveness. The drawback of this algorithm is that it considers the same number of resources for all types of tasks which may not be desirable in real time scenario [35].

4.2.4 Linear Programming Based Affinity Scheduling (LPAS)

LPAS is a mapping heuristic for heterogeneous computing systems [36]. LPAS_DG adapts this algorithm for Desktop

Grid systems [37]. It uses an optimization approach to find the best set of candidate resources. It aims to combine the advantages of the MCT and MET algorithms in the spirit of the KPB algorithm. The LPAS algorithm simultaneously reduces the state information and average completion time of tasks, but it requires the arrival rates and mean execution times for each class of tasks on each resource.

4.2.5 Min – Min Heuristic Based Algorithms

Min-Min algorithm comprises of two phases and begins with the set MT (Meta Task) of all unassigned tasks. In the first phase, the set of minimum expected completion time for each task in MT is found. In the second phase, the task with the overall minimum expected completion time from MT is chosen, assigned to the corresponding machine and removed from MT. The same iterations continue until all tasks in the MT are mapped [38].

Table 1–Taxonomy of Scheduling Approaches in Grids

S. No.	Basis of Classification	Sub Classes (if any)	References
1	Abstract Approaches	First Come First Serve (FCFS)	[23 - 29]
		Round Robin (RR)	[30 - 31]
		Join Shortest Queue (JSQ)	[32]
2	Heuristic Based Approaches	Minimum Execution Time (MET)	[33 - 34]
		Minimum Completion Time (MCT)	[33]
		K-Percent Best (KPB)	[35]
		Linear Programming Based Affinity Scheduling (LPAS)	[36 - 37]
		Min – Min Heuristic	[38 – 41]
		Max-Min Heuristic	[42 - 44]
		RASA	[45]
		Switcher Algorithm	[46]
		Suffrage Algorithm	[47]
		Min – Mean Algorithm	[48]
		Most Fit Task First (MFTF)	[49]
3	Artificial Intelligence Based Approaches	A* Algorithm	[50]
		Simulated Annealing (SA)	[51 - 55]
		Genetic Simulated Annealing (GSA)	[56 - 57]
		Tabu Search	[58 - 59]
4	Directed Acyclic Graphs & Petri Net Based Approaches	***	[60 - 69]
5	Constraint Based / Resource Aware Approaches	***	[45] [70 - 74]
6	Local vs. Global Scheduling Based Approaches	***	[75 - 76]
7	Rescheduling Based Approaches	***	[39] [72] [77]
8	Soft Computing Based Approaches	Fuzzy Logic Based Approaches	[78]
		Genetic - Fuzzy Based Approaches	[79]
		Genetic Algorithm Based Approaches	[80]
9	Job-Grouping Based Approaches	***	[81]
10	QoS Based Approaches	***	[82 - 85]

Min-Min algorithm minimizes makespan than the other heuristics but it fails to produce a load balanced schedule. A Load Balanced Min-Min (LBMM) algorithm that reduces the makespan and increases the resource utilization is proposed in [39]. In the first phase the traditional Min-Min algorithm is executed and in the second phase the tasks are rescheduled to use the unutilized resources effectively. Further, another scheduling algorithm based on Min–Min heuristic is proposed in [40] which first estimates the completion time of the tasks on each of resources and then selects the appropriate resource for scheduling. A heuristic based on Min-Min heuristic is also proposed in [41].

4.2.6 Max-Min Heuristic Based Algorithms

Max-Min algorithm differs from Min-Min in second phase, where tasks with overall maximum expected completion time from MT is chosen and assigned to corresponding machine. Thus, Min-Min gives priority to the task that has the shortest earliest completion time, but Max-Min tends to schedule the longer tasks first [42].

A static batch mode heuristic for efficient task scheduling is proposed in [43]. Further, a skewness based Min-Min Max-Min heuristic is proposed in [44] for efficient task scheduling while ensuring effective utilization of resources.

4.2.7 RASA

This algorithm alternatively applies Max-Min and Min-Min over scheduling process iterations. For example, if the first task is assigned to a resource by Max-Min strategy, in the next round the task will be assigned by Min-Min and so on. Experiments reveal that if the number of available resources in grid system is odd then it is highly preferred to start by Min-Min algorithm in first round otherwise starting by Max-Min algorithm is recommended. For next rounds just resources are just assigned to tasks using a strategy different from last round ignoring waiting time of the small tasks in Max-Min algorithm and the waiting time of the large tasks in Min-Min algorithm. The time complexity of RASA is like Max-Min and Min-Min, O (mn^2) where m is the total number of resources and n is the number of tasks [45].

4.2.8 Switcher Algorithm

Switcher, as the name depicts, switches between the Max-Min and Min-Min algorithm to select the best on the basis of Standard Deviation (SD) of minimum completion time of unassigned jobs. A position in Meta Task (MT) where the difference in completion time between the two successive jobs is more than the value of SD is searched. If it lies in first half of the list, then Min-Min algorithm is evaluated as the number of longer jobs is more, otherwise Max-Min is evaluated by taking the last job from the list. If this position does not exist, then SD is compared with a threshold value and job allocation is done using Min-Min strategy, if SD is smaller than threshold value. Otherwise, Max-Min is selected for assigning the next job [46].

4.2.9 Suffrage Algorithm

In this two step algorithm, the suffrage value (calculated as the difference between the minimum and second minimum completion time) for each job are found in first step followed by the assignment of the task with maximum suffrage value to corresponding machine with minimum completion time [47].

4.2.10 Min – Mean Algorithm

Min-Mean heuristic scheduling is proposed in [48] for static meta-tasks. The proposed algorithm applies original Min-Min then estimates the mean makespan of all the resources and finally reschedules tasks. In case of heterogeneity among the submitted tasks, the algorithm performs better than the Min-Min.

4.2.11 Most Fit Task First (MFTF)

As some existing static scheduling methods may not accomplish well in the case of dynamic task arrivals, an adaptive and dynamic scheduling method called Most Fit Task First (MFTF) is proposed in [49] for computational grids supporting heterogeneous computing nodes and dynamic task arrivals.

4.3 ARTIFICIAL INTELLIGENCE BASED APPROACHES

In an effort for devising the adaptive solutions to scheduling problem, the use of AI techniques have also been reported in the literature. The few selected algorithms under this class are as follows:

4.3.1 A* Algorithm

A* is a tree search technique based on a m-array tree, beginning at a root node that is a null solution. As the tree grows, intermediate nodes represent partial mappings and leaf nodes represent final mappings. Each node has a cost function, and the node with the minimum cost function is replaced by its child node. Whenever a node is added, to reduce the height of the tree, the tree is pruned by deleting the node with the largest cost function. This process is repeated until a complete mapping (a leaf node) is reached [50].

4.3.2 Simulated Annealing (SA)

It is an iterative technique that considers only one possible mapping for each meta-task at a time. Simulated annealing uses a procedure that probabilistically allows poor solutions to be accepted to attempt to obtain a better search of the solution space based on a system temperature [51-55].

4.3.3 Genetic Simulated Annealing (GSA)

It is a combination of the GA and SA techniques [56-57]. GSA follows the procedures similar to the GA. For the selection process, GSA uses the SA cooling schedule and system temperature.

4.3.4 Tabu Search

It is a solution space search that keeps track of the regions of the solution space to avoid repeating a search near the areas that have already been searched [58-59]. A mapping of meta-tasks uses the same representation as a chromosome in the GA approach. Its implementation begins with a random mapping, generated from a uniform distribution.

4.4 DIRECTED ACYCLIC GRAPHS & PETRI NET BASED APPROACHES

A general framework to facilitate directed acyclic graph (DAG) scheduling in grid systems is proposed in [60] wherein the consideration for dynamic changes in grid computing resources is missing. Several Petri net models, such as extended time Petri nets, colored Petri nets (CPNs), and stochastic Petri nets (SPNs), have been developed to address such dynamic changes, and are considered to be effective tools for scheduling and resource allocation in grid computing systems [61-63].

Petri nets assign tasks to grid resources either using a distributed scheme, or a hierarchical scheme. In a distributed scheme, a broker considers resources to be distributed states for each request, sends the request to the sites that contain the distributed resources, and receives results from the distributed sites' coordinators. In a hierarchical scheme, requests for resources from a site are arranged hierarchically and the resources are classified based on their geographical location in relation to the site making the request, and sites have coordinators at different levels. In a hierarchical scheme, scheduling occurs in three layers, and each site has one broker and scheduler, to control and manage allocation of its resources. Tasks are sent through the hierarchical brokers, to sites that have the needed resources. A distributed scheme is better than a hierarchical scheme, when all tasks are locally requested on sites as a request is sent directly to the brokers on sites where the resource is available [64-65].

4.4.1 HTPN

A three-layer model based on a hierarchical time Petri net (HTPN) is presented in [61], with different Petri net models constructed for each level. HTPN focuses on independent tasks, and does not consider dependent tasks. Dependent task scheduling is presented in [62], using an extended time Petri net.

4.4.2 HSPN

The grid resource scheduling model based on Petri nets is extended in [63] wherein a four-level scheduling algorithm that considers independent tasks is proposed. Hierarchical Stochastic Petri Net (HSPN) [66] is a hierarchical scheme that uses SPNs to schedule and allocate resources, based on the hierarchical and distributed schemes presented in [67].

4.4.3 ICPDP

An algorithm for handling DAG type of tasks named Improved Critical Path using Descendant Prediction (ICPDP) is proposed in [68]. The proposed algorithm is integrated in the DIOGENES project and experimental results reveal the improvement that ICPDP brings to DAG scheduling.

4.4.4 Reliability Aware Scheduling using DAG

A reliability-aware scheduling algorithm based on directed acyclic graph (DAG) for precedence constrained tasks is proposed in [69] which ensure high quality of reliability for applications.

4.5 CONSTRAINT BASED / RESOURCE AWARE APPROACHES

This class of approaches keep the various constraints and dependencies, associated with resources and tasks, at the focal point during the process of solving scheduling problems in the domain of grid computing. The need of monitoring the dynamic states of the resources becomes essential to derive the scheduling decisions thereby classifying such approaches as resource aware approaches. The various approaches of this class are as follows:

4.5.1 CBJRS

Constraint-Based Job and Resource scheduling (CBJRS) algorithm is proposed in [70] to reduce the processing time, processing cost and enhance the resource utilization in comparison to other algorithms.

4.5.2 NIMROD

It is a constraint based scheduling model guided by the deadline and grid economy model proposed in [71].

4.5.3 Resource Aware Scheduling Algorithm

A Resource Aware Scheduling Algorithm which leverages two existing task scheduling algorithms, Min–Min and Max–Min, is described in [45]. This algorithms use an estimation of tasks completion time and resource execution time. The presented algorithm alternates the two algorithms depending on input tasks.

4.5.4 Fault Tolerant Scheduling for Dependent and Dynamic Tasks

An algorithm supporting dynamic behavior of tasks is proposed in [72]. It is suitable for arbitrary constraints tasks whose dependencies are organized as a graph, having the tasks as nodes and the constraints as edges. It consists of two phases: an initial scheduling phase and a rescheduling phase, in which tasks are separated in entry tasks and inner tasks, based on dependency of failing tasks. Depending on the type of the node that fails, there may be used different scheduling algorithms: Highest Level First with Estimated Times, Modified Critical Path or Earliest Time First.

4.5.5 Comparison of Scheduling Approaches

Typical scheduling structures for computational grids are discussed in [73] along with an introduction and classification of various scheduling algorithms and selection strategies applicable to these structures. Simulations were used to evaluate these aspects considering combinations of different Job and Machine Models. For hierarchical scheduling the simulation results reveal the benefit of Backfill. Unexpected results are obtained as FCFS proves to perform better than Backfill when using a central job-pool.

4.5.6 HRN

Highest Response Next (HRN) Algorithm allocates the jobs to processors based on priority and processor competence. Scheduling is based on priority, time, memory and CPU requirements. HRN effectively utilize the resources than First Come First Served scheduling algorithm. The main drawbacks of this algorithm are difficulties in finding priority of job and higher turnaround time [74].

4.6 LOCAL VS. GLOBAL SCHEDULING BASED APPROACHES

The design and implementation of GangSim simulator is proposed in [75] to support studies of scheduling strategies in grid environment, with a special focus on investigating the interactions between local and community resource allocation policies.

Impact of data migration under a variety of demanding grid conditions is explored in [76] along with the evaluation of the proposed scheduling algorithm on different groupings of servers into locales, and between server systems, utilizing genuine workloads got from driving supercomputing focuses. Hybrid Scheduling based approaches are also witnessed in the literature so as to combine the positives of local and global scheduling.

4.7 RESCHEDULING BASED APPROACHES

A load balancing algorithm for fair scheduling is proposed in [77] for a computational grid. It addresses the fairness issues by using mean waiting time. It schedules the task by using fair completion time and then reschedules by using mean waiting time of each task to obtain load balance. The performance metrics emphasized are execution time and execution cost. Rescheduling is also involved in a number of approaches discussed so far like [39] and [72].

4.8 SOFT COMPUTING BASED APPROACHES

The demand of the user community for high speed and accuracy directs for devising adaptive and robust solutions to scheduling problems that can handle the uncertainties associated with load estimation and length of jobs along with handling dynamic pattern of task arrival. This fact has inspired the researchers to apply soft computing techniques viz. fuzzy logic, genetic algorithms, other hybrid approaches etc. for scheduling problems in the domain of grid computing. The various approaches of this class are as follows:

4.8.1 Fuzzy Logic Based Approaches

The main challenges faced by grid environment are integration, interoperability of security framework and the trust relationship between participants. A fuzzy-logic-based self-adaptive job replication scheduling (FSARS) algorithm which considers the trust relationships between the participants is proposed in [78]. FSARS uses the security demand of the task and Trust level of the resources as the main parameters. The proposed method gives a vigorous performance against resource failures and improved scheduling achievement rate.

4.8.2 Genetic Algorithm & Genetic – Fuzzy Based Approaches

Scheduling problem of independent tasks in the market-based grid environment is addressed in [79] wherein NSGA-II is used to optimize task scheduling problem in grid. For reducing computation, authors considered load balancing problem and enhanced it in task scheduling ultimately using fuzzy system without implementing third objective function. For the first time, authors proposed Variance based Fuzzy Crossover operator for this purpose and more variety in Pareto-optimal solutions. Two functions are defined to produce two inputs for fuzzy system. Variance of Costs and occurrence of resources in scheduling are used to stipulate probability of crossover logically. Second fuzzy function with cooperation of Makespan objective satisfies load balancing objective indirectly. Further, a genetic algorithm based efficient task scheduling approach is proposed in [80].

4.9 Job-Grouping Based Approaches

There is a need for an efficient job grouping-based scheduling system to dynamically assemble the lightweight jobs of an application into a group of jobs, and send these grouped jobs to the grid resources because the lightweight jobs which are generally large in number cause high overhead time and cost when executed on computational grids. The dynamic grouping of such fine grained jobs based on the computational requirements of each job with respect to availability, processing capability and network bandwidth of computational grid resources is proposed in [81].

4.10 QUALITY OF SERVICE BASED APPROACHES

The users connected to the grid system have varying QoS requirements and their satisfaction is a major determinant for the effectiveness of the grid resource management. This class of scheduling approaches keeps the users' QoS requirements at the focus. This class comprise of the following approaches:

4.10.1 QWMTM & QWMTS

Two algorithms called QoS Guided Weighted Mean Time-Min (QWMTM) and QoS Guided Weighted Mean Time Min-Min Max-Min Selective (QWMTS) are proposed in [82]. These algorithms schedule independent batch tasks while considering network bandwidth as QoS parameter.

4.10.2 Predictive Max-Min Min-Min Switcher

QoS based predictive Max-Min, Min-Min Switcher for scheduling jobs is proposed in [83]. Based on heuristics, appropriate selection among QoS based Min-Min and QoS based Max-Min is made. The algorithm uses historical information about the execution time of jobs to predict the performance.

4.10.3 QoS Guided Min-Min Heuristic

QoS guided Min-Min based on original Min-Min is proposed in [84] which schedules the tasks with high bandwidth requirements before the others. Simulation results reveal that QoS guided Min-Min is better than original Min-Min when submitted tasks varies highly in their bandwidth.

Further, anaother QoS based efficient task scheduling approach proposed in [85] uses ETC matrix.

5. CONCLUSION & FUTURE DIRECTIONS

On the basis of the aforesaid literature on scheduling algorithms, chronological research trends in scheduling for grid computing systems with a special emphasis on computational grids is shown through a bar chart in Figure 2.

As shown in the graph, AI based approaches were the major focus of researchers till the year 2000. Thereafter, DAG & Petri net based approaches were commonly used over a period of 10 years. It is quite evident from the figure that heuristics based approaches remain the focus of researchers in all the time periods. It is because of the fact that being an NP-Complete problem, an optimal schedule can easily be devised through heuristics. Further, soft computing and rescheduling based approaches are widely used in the present era. The reason for the same is the capability of soft computing techniques to deal with uncertainties associated with the dynamic job arrival patterns and workload estimation of nodes. Further, the rescheduling enables to combine two approaches as none of the approaches guarantee the optimal solution to the scheduling problem in the domain of grid computing systems.

In view of the research trends presented in the graph, the research community must explore the reasons for the considerable shift of researchers' focus from Artificial

Intelligence & Constraint based approaches to other options. Similarly, the shift being noticed in present time from DAG & Petri net based approaches must critically be analyzed. Further, increasing inclination towards Soft Computing based approaches must explore the scope for other Soft Computing techniques to devise an effective solution to the scheduling problem in Grids. These future directions for research will surely guide the researchers and enable them to make an informed choice of the scheduling approach in view of major characteristics of the grid application.

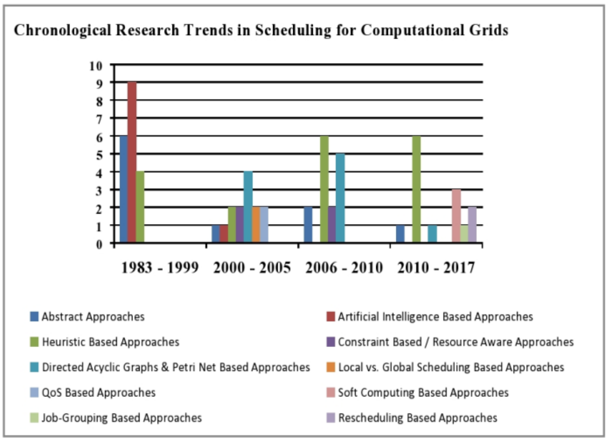

Figure 2 – Research Trends in Scheduling for Grids

6. ACKNOWLEDGEMENTS

Authors feel grateful to the research community because of whose contributions the survey could be performed, taxonomy could be proposed and chronological research trends could be analyzed. The acknowledgements are also due to all the publishing agencies for making the contributions accessible.

REFERENCES

[1] Foster I, C. Kesselman, and S. Tuecke, The anatomy of the grid: Enabling scalable virtual organizations, the International Journal of High Performance Computing Applications, Volume 15, Number 3, 2001, pp. 200-222.

[2] B. Jacob, Brown, M., Fukui, K., & Trivedi, N., “Introduction to grid computing”, IBM Redbook, 2005

[3] Frederic Magoules, Kiat-An Tan and Abhinit Kuma, “Introduction to grid computing”, CRC press, 2009

[4] Xhafa, F., “Metaheuristics for Scheduling in Distributed Computing Environments”, Springer-Verlag, 2008, pp. 2-9.

[5] Hemamalini, M., “Review of Grid Task Scheduling in Distributed Heterogeneous Environment”, International Journal of Computer Applications. Vol. 40 (2), 2012, pp. 24 – 26.

[6] Yan, Kuo-Qin, Shun-Sheng Wang, Shu-Ching Wang, and Chiu-Ping Chang, "Towards a hybrid load balancing policy in grid computing system", Expert Systems with Applications, Volume **36**, and Number **10**, **2009**, pp. 12054- 12064.

[7] Subrata, Riky, Albert Y. Zomaya, and Bjorn Landfeldt., "Artificial life techniques for load balancing in computational grids." Journal of Computer and System Sciences, Volume 73, Number 8, 2007, pp. 1176-1190.

[8] Lu, Kai, Riky Subrata, and Albert Y. Zomaya, "On the performance-driven load distribution for heterogeneous computational grids." Journal of Computer and System Sciences 73, no. 8 (2007): 1191-1206.

[9] Jasma Balasangameshwara, “Improving Fault-Tolerant Load Balancing Algorithms in Computational Grids”, I.J. Information Engineering and Electronic Business, 2015, Volume 6, pp. 53-62.

[10] D.K. Patel et al., “Survey of Load Balancing Techniques for Grid”, Journal of Network and Computer Applications 65(2016)103–119 105

[11] Yagoubi B, Slimani Y. “Dynamic load balancing strategy for grid computing.” World Academy of Science, Engineering & Technology 2006:90–5.

[12] Yagoubi B, Lilia H T, Maussa H S. Load balancing in grid computing. Asian Journal of Information Technology 2006;5(10):1095–103.

[13] Yagoubi B, Slimani Y. Task Load balancing strategy in grid environment. Journal of Computer Science 2007;3:186–94.

[14] Yagoubi B, Slimani Y. “Load balancing strategy in grid environment”, Journal of Information Technology Applications 2007; 1(4):285–96.

[15] Shan, Hongzhang, et al., "Scheduling in heterogeneous grid environments: The effects of data migration", International Conference on Advanced Computing and Communication,

[16] Buyya Rajkumar, David Abramson, and Jonathan Giddy, "Grid Resource Management, Scheduling, and Computational Economy", In International Workshop on Global and Cluster Computing, Japan, Volume 21, pp. 2002-2040. 2000.

[17] Karl Czajkowski, Ian Foster, Nick Karonis, and Steven Tuecke, "A resource management architecture for metacomputing systems", In Workshop on Job Scheduling Strategies for Parallel Processing, pp. 62-82. Springer, Berlin, Heidelberg, 1998

[18] Buyya Rajkumar, David Abramson, and Jonathan Giddy, "Grid Resource Management, Scheduling, and Computational Economy", In International Workshop on Global and Cluster Computing, Japan, Volume 21, pp. 2002-2040. 2000.

[19] Hao Y, Liu G, Wen N., An enhanced load balancing mechanism based on deadline control on GridSim. Future Generation Computer Systems 2012;28:657–65.

[20] T. Casavant and J. Kuhl, "A Taxonomy of Scheduling in General Pupose Distributed Computing Systems", "IEEE Trans. on Software Engineering", vol. 14, no. 2, February 1988, pp. 141-154.

[21] M. Arora, S. K. Das, R. Biswas, "A Decentralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments", "Proc. Of International Conference on Parallel Processing Workshops (ICPPW'02)", Vancouver, British Columbia Canada, August 2002, pp. 499-505.

[22] Fatos Xhafa, Ajith Abraham, "Computational models and heuristic methods for Grid scheduling problems", "Future Generation Computer Systems 26", 2010, pp. 608-621.

[23] El-Rewini, H., Ali, H.H., Lewis, T. Task scheduling in multiprocessing systems, IEEE Journal, December 1995, vol. 28, pp. 27-37.

[24] U. Schwiegelshohn, R. Yahyapour, "Analysis of First-Come-First-Serve parallel job scheduling", " Proceedings of the 9th SIAM Symposium on Discrete Algorithms", 1998, pp. 629-638.

[25] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R. F. Freund, "Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems", "J. of Parallel and Distributed Computing", vol. 59, no. 2, November 1999, pp.107-131.

[26] Dr.D.I.Georage Amalarethinam, P.Muthulakshmi, "An Overview of the Scheduling Policies and Algorithms in Grid Computing", " International Journal of Research and Review in Computer Science (IJRRCS)", vol. 2,no. 2, April 2011, pp. 280-294.

[27] R. Armstrong, D. Hensgen, and T. Kidd, "The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions", "7th IEEE Heterogeneous Computing Workshop (HCW '98)", 1998, pp. 79-87.

[28] R. F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith,T.Kidd, M. Kussow, J. D. Lima, F. Mirabile, L. Moore, B. Rust, and H. J. Siegel, "Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet", "7th IEEE Heterogeneous Computing Workshop (HCW '98)", 1998, pp. 184-199.

[29] R. F. Freund and H. J. Siegel, "Heterogeneous processing", IEEE Comput. 26, June 1993.

[30] Rasmus V. Rasmussen, Michael A. Trick . Round Robin scheduling – a survey, European Journal of Operational Research, vol. 188, Issue 3, August 2008, pp. 617–636.

[31] Ruay-Shiung Chang, Jih-Sheng Chang, Po-Sheng Lin, "An ant algorithm for balanced job scheduling in grids", "Future Generation Computer Systems 25", 2009, pp. 20–27.

[32] Tracy D. Braun, Howard Jay Siegel, Noah Beck, Lasislau L B • ol • oni, Muthucumara Maheswaran, and Albert I Reuther. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, 61(6):810{837, 2001. ISSN 0743-7315. doi: 10.1006/jpdc.2000.1714

[33] R. Armstrong, D. Hensgen, and T. Kidd, "The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions", in 7th IEEE Heterogeneous Computing Workshop (HCW '98), pp. 79_87, 1998.

[34] R. F. Freund, M. Gherrity, S. Ambrosius, et al., "Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet", in 7th IEEE Heterogeneous Computing Workshop (HCW '98), pp. 184_199, 1998.

[35] Muthucumaru Maheswaran, Shoukat Ali, Howard Jay Siegel, Debra Hensgen, and Richard F. Freund, "Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems", In Proceedings of the 8th Heterogeneous Computing Workshop (HCW' 99), page 30, Los Alamitos, CA, USA, 1999. IEEE Computer Society. ISBN 0-7695-0107-9. doi: http://doi.ieeecomputersociety.org/10.1109/HCW.1999.765094.

[36] Issam Al-Azzoni and Douglas G. Down. Linear Programming-Based A_nity Scheduling of Independent Tasks on Heterogeneous Computing Systems. IEEE Transactions on Parallel and Distributed Systems, 19(12):1671{1682, 2008. doi: 10.1109/TPDS.2008.59.

[37] Issam Al-Azzoni and Douglas G. Down. Dynamic Scheduling for Heterogeneous Desktop Grids. In Proceedings of the 9th IEEE/ACM International Conference on Grid Computing (GRID '08), pages 136{143, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-1-4244-2578- 5. doi: 10.1109/GRID.2008.4662792.

[38] M. Wu, W. Shu and H. Zhang, "Segmented Min-Min: A Static Mapping Algorithm for Meta-Tasks on Heterogeneous Computing System", in Proc. 9th Heterogeneous Workshop (HCW'00), 2000, pp. 375-385.

[39] T. Kokilavani and D.I. George Amalarethinam, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing", International Journal of Computer Applications (IJCA) Volume 20, Number 2, April 2011

[40] Anousha S, Ahmadi M., "An improved Min–Min task scheduling algorithm in grid computing", In Proceedings of the international conference on grid and pervasive computing (GPC'13). Lecture Notes in Computer Science; 2013, vol.7861. p. 103–13.

[41] Anousha, S., Shoeib, A., and Ahmadi, M., "A New Heuristic Algorithm for Improving Total Completion

Time in Grid Computing", Springer Verlag Berlin Heidelberg, 2014, pp. 17-26.

[42] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R. F. Freund, "Dynamic Matching and Scheduling of a Class of Independent tasks onto Heterogeneous Computing Systems," Journal of Parallel and Distributed Computing, Vol. 59, No. 2, pp. 107-131,1999.

[43] Panda, S., Agrawal, P., Khilar, P., & Mohapatra, D., "Skewness-Based Min-Min Max-Min Heuristic for Grid Task Scheduling". In proceedings of 4th IEEE International Conference on Advanced Computing and Communication Technologies, 2014, pp. 282-289.

[44] Vijayalakshmi, R., and Vasudevan, V., "Static Batch Mode Heuristic Algorithm for Mapping Independent Tasks in Computational Grid", Journal of Computer Science, 2015, Vol. 11(1), pp. 224.

[45] Saeed Parsa, Reza Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm", "International Journal of Digital Content Technology and its Applications", vol. 3, no. 4, December 2009, pp. 91-99.

[46] M. Singh andP.K.Suri, "QPSMax-Min<>Min-Min : A QoS Based Predictive Max-Min, Min-Min Switcher Algorithm for Job Scheduling in a Grid", *Information Technology Journa,l* Vol. 7, No. 8, pp. 1176-1181, 2008.

[47] E. U. Munir, J. Li and S. Shi, "QoS Sufferage Heuristic for Independent Task Scheduling in Grid," Information Technology Journal, Vol. 6, No. 8, pp. 1166-1170, 2007.

[48] Kamalam.G.K and Muralibhaskaran.V, "A New Heuristic Approach:Min-Mean Algorithm For Scheduling Meta-Tasks On Heterogenous Computing Systems", "International Journal of Computer Science and Network Security", vol.10, no.1, January 2010.

[49] Sheng-De Wang, I-Tar Hsu and Zheng-Yi Huang, "Dynamic scheduling methods for computational grid environments", Proceedings 11th IEEE International Conference onParallel and Distributed Systems, 2005. Fukuoka, Japan, 20-22 July 2005.

[50] K. Chow and B. Liu, "On mapping signal processing algorithms to a heterogeneous multiprocessor system", in 1991 International Conference on Acoustics, Speech, and Signal Processing (ICASSP'91),Vol. 3, pp. 1585_1588, 1991.

[51] Z. Michalewicz and D. B. Fogel, "How to Solve It: Modern Heuristics," Springer-Verlag, New York,2000.

[52] M. Coli and P. Palazzari, "Real time pipelined system design through simulated annealing", J. Systems Architecture 42, 6_7 (Dec. 1996), 465_475.

[53] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Optimization by simulated annealing, Science 220, 4598 (May 1983), 671_680..

[54] S. J. Russell and P. Norvig, "Artificial Intelligence: A Modern Approach"', Prentice_Hall, Englewood Cliffs, NJ, 1995.

[55] A. Y. Zomaya and R. Kazman, "Simulated annealing techniques, in Algorithms and Theory of Computation Handbook" (M. J. Atallah, Ed.), pp. 37-1_37-19, CRC Press, Boca Raton, FL, 1999.

[56] H. Chen, N. S. Flann, and D. W. Watson, "Parallel genetic simulated annealing: A massively parallel SIMD approach", IEEE Trans. Parallel Distrib. Comput. 9, 2 (Feb. 1998), 126_136.

[57] P. Shroff, D. Watson, N. Flann, and R. Freund, "Genetic simulated annealing for scheduling data-dependent tasks in heterogeneous environments", in ``5th IEEE Heterogeneous Computing Workshop (HCW '96)," pp. 98_104, 1996.

[58] I. De Falco, R. Del Balio, E. Tarantino, and R. Vaccaro, "Improving search by incorporating evolution principles in parallel tabu search", in 1994 IEEE Conference on Evolutionary Computation, Vol. 2, pp. 823_828, 1994.

[59] F. Glover and M. Laguna, "Tabu Search", Kluwer Academic, Boston, MA, 1997.

[60] J. Yu, and R. Buyya, "A budget constrained scheduling of workflow applications on utility grids using genetic algorithms," in Proceedings of the Workshop on Workflows in Surpport of Large-Scale Science, Paris, 2006, pp. 1-10.

[61] Y. Han, C. J. Jiang, and S. Luo, "Resource scheduling model for grid computing based on sharing synthesis of Petri net," in Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design, Coventry, UK, 2005, pp. 367-372.

[62] Y. Han and X. Luo, "Modelling and performance analysis of grid task scheduling based on composition and reduction of Petri nets," in Proceedings of the 5th International Conference on Grid and Cooperative Computing, Changsha, China, 2006, pp. 331-334.

[63] X. Zhao, B. Wang, and L. Xu, "Grid application scheduling model based on Petri net with changeable structure," in Proceeding of 6th International Conference on Grid and Cooperative Computing, Los Alamitos, CA, 2007, pp. 733-736.

[64] Y. Han, C. Jiang, and X. Luo, "Resource scheduling scheme for grid computing and its Petri net model and analysis," Parallel and Distributed Processing and Applications, Lecture Notes in Computer Science vol. 3759, G. Chen et al., editors, Heidelberg: Springer, pp. 530-539, 2005.

[65] Z. Hu, R. Hu, W. Gui, J. Chen, and S. Chen, "General scheduling framework in computational grid based on Petri net," Journal of Central South University of Technology, vol. 12, no. 1, pp. 232-237, 2005.

[66] Mohammad Shojafar , Zahra Pooranian, Jemal H. Abawajy, Mohammad Reza Meybodi , "An Efficient Scheduling Method for Grid Systems Based on a Hierarchical Stochastic Petri Net", Journal of Computing

Science and Engineering, *Vol. 7, No. 1, March 2013, pp. 44-52*

[67] V. Subramani, R. Kettimuthu, S. Srinivasan, and P. Sadayappan, "Distributed job scheduling on computational grids using multiple simultaneous requests," in Proceedings of 11th IEEE International Symposium on High Performance Distributed Computing, Edinburgh, UK, 2002, pp. 359-366.

[68] B. Simion, C. Leordeanu, F. Pop, V. Cristea, A hybrid algorithm for scheduling workflow applications in grid environments (ICPDP), in: Proc. of the 2007 OTM Confederated Int. Conf. on On the Move to Meaningful Internet Systems: CoopIS, DOA, ODBASE, GADA, and IS—Volume Part II, OTM'07, Springer-Verlag, Berlin, Heidelberg, ISBN: 3-540-76835-1, 2007, pp. 1331–1348. 978-3-540-76835-7. URL: http://dl.acm.org/citation.cfm?id=1784707.1784728.

[69] X. Tang, K. Li, R. Li, B. Veeravalli, Reliability-aware scheduling strategy for heterogeneous distributed computing systems, J. Parallel Distrib. Comput. (ISSN: 0743-7315) 70 (9) (2010) 941–952. http://dx.doi.org/10.1016/j.jpdc.2010.05.002.

[70] Soni, V. K., Sharma, R., Mishra, M. K., & Das, S. Constraint-based job and resource scheduling in grid computing, 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 2010, Vol. 4, 2010, July, pp. 334-337

[71] Buyya R, Murshed M, Abramson D. A deadline and budget constrained cost-time optimization algorithm for scheduling task farming applications on global Grids, In *2002 International Conferenceon Parallel and Distributed Processing Techniques and Applications (PDPTA'02),* Las Vegas, Nevada, USA, June 2002.

[72] A. Olteanu, F. Pop, C. Dobre, V. Cristea, A dynamic rescheduling algorithm for resource management in large scale dependable distributed systems, Comput. Math. Appl. 69 (9) (2012) 1409–1423.

[73] Hamscher, Volker, et al. "Evaluation of job-scheduling strategies for grid computing." Grid Computing—GRID pp. 191-202, 2000

[74] Raksha Sharma, Vishnu Kant Soni, Manoj Kumar Mishra, Prachet Bhuyan, "A Survey of Job Scheduling and Resource Management in Grid Computing", World Academy of Science, Engineering and Technology, 2010.64.

[75] Dumitrescu, C. L., & Foster, I., GangSim: a simulator for grid scheduling studies, IEEE International Symposium on Cluster Computing and the Grid, 2005,Volume 2, May 2005, pp. 1151-1158

[76] Hongzhang Shan, Leonid Oliker and Warren Smith, "Scheduling in heterogeneous grid environments: The effects of data migration", International Conference on Advanced Computing and Communication, Gujarat, India. 2004.

[77] U.Karthick Kumar, "A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 5, No 1, September 2011

[78] Deepa N. K. and L. M. Nithya, "Fuzzy Logic Based Job Scheduling in Computational Grid with Minimum Communication and Replication Cost", International Journal of Innovative Research in Science, Engineering and Technology, Volume 3, Special Issue 1, February 2014.

[79] Reza Salimi Navid Bazrkar and Mostafa Nemati, "Task Scheduling for Computational Grids Using NSGA II with Fuzzy Variance Based Crossover", Advances in Computing 2013, Volume 3, Number 2, pp. 22-29.

[80] K. Sunitha, P.V. Sudha, "An efficient task scheduling in distributed computing systems by improved genetic algorithm", International Journal of Communication Network Security, ISSN: 2231 – 1882, Volume-2, Issue-2, 2013

[81] Mukherjee, Arijit. "An Efficient Job-Grouping Based Scheduling Algorithm for Fine-Grained Jobs in Computational Grids." Ph.D. dissertation 2011.

[82] Sameer Singh Chauhan,R. Joshi. C, "QoS Guided Heuristic Algorithms for Grid Task Scheduling", "International Journal of Computer Applications", vol. 2, no.9, June 2010, pp 24-31.

[83] Singh. M and Suri. P.K, "QPS A QoS Based Predictive Max-Min, Min-Min Switcher Algorithm for Job Scheduling in a Grid", "Information Technology Journal", vol. 7, Issue. 8, 2008, pp. 1176-1181.

[84] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Min-min Heuristic for Grid Task Scheduling," Journal of Computer Science and Technology, vol. 18, 2003, pp. 442-451.

[85] L. Chunlin, and L. Layuan, "QoS based resource scheduling by computational economy in computational grid," Journal of Information Processing Letters, Vol. 98, pp. 119-126, 2006.