A Study on Scheduling in Grid Environment

[1]Dr.K.Vivekanandan, [2]D.Ramyachitra

[1] Professor, BSMED, Bharathiar University, Coimbatore, Tamil Nadu, India
[2] Assistant Professor, School of CSE, Bharathiar University, Coimbatore, Tamil Nadu, India

Abstract

Grid Computing is a high performance computing environment that allows sharing of geographically distributed resources across multiple administrative domains and used to solve large scale computational demands. In the grid environment, users can access the resources transparently without knowing where they are physically located. To achieve the promising potentials of computational grids, job scheduling is an important issue to be considered. Scheduling is very complica
ted due to the unique characteristics of the grids. This paper gives a classification of scheduling algorithms in distributed computing and the algorithms that are applicable to grid environment. It also studies performance of various scheduling algorithms and the softwares that support scheduling in real grid environment as well as simulated environment.

Keywords: scheduling, grid, local, global, static, dynamic, heuristic

1. Introduction

The first paper on critical path scheduling was published in March 1959 by Kelley Walker who defined the science of scheduling using critical path analysis. The evolution of scheduling was then applied in the development of computers. The mainframes migrated to minicomputers in 1970’s and 1980’s where manual scheduling techniques were used and only very large organizations were able to use central scheduling office and the supporting computer systems. The spread of cheap and easy to use PC’s in the latter half of 1980’s initiated a lot of PC based scheduling systems. The emergence of easy to use scheduling tools with graphical interface changed the industry which resulted in scheduling migrated to the desktop [1]. Due to the decline in price of PC’s and supercomputers were expensive, a cluster of computers i.e., many small computers that were combined to make a computing structure that could provide all the processing power needed with much less money altered supercomputers. In the beginning when distributed computing came into existence, it was used only in the limited area of scientific computing for analysis of protein folding, climate situations, nuclear physics etc and it took several months to complete the jobs. Scheduling algorithms was not much important in the traditional distributed computing projects and so the effect of task scheduling on the job completion time was relatively small [2]. Grid computing a form of distributed computing originated in the early 1990’s for making computer power as easy to access as a power grid [3]. It is a super virtual computer that consists of many loosely coupled networked computers that is used to execute very large applications such as DNA analysis, simulation of atmospheric circulation or ocean circulation, structural and stress analysis, water resistance of ships, earthquake analysis, chemical spill from a factory, volcano analysis, flood, wildfire, landslide etc. These types of applications which contain many tasks may take several days or weeks to complete whose completion time is affected by task scheduling. The delay in a single task can affect the completion time of the entire application. The main goal of scheduling is to minimize the job completion time and wastage of CPU cycles [2] but scheduling jobs in a heterogeneous grid environment is different compared to parallel architectures.

Before scheduling the tasks in the grid environment, the characteristics of the grid should be taken into account. Some of the characteristics of the grid includes (i) geographical distribution where the resources of grid may be located at distant places (ii) heterogeneity, a grid consists of hardware as well as software resources that may be files, software components, sensor programs, scientific instruments, display devices, computers, supercomputers networks etc (iii) resource sharing, different organizations may own the resources of the grid (iv) multiple administrations, each organization may establish different security and administrative policies to access their resources (v) resource coordination, to get combined computing capabilities, grid resources must be coordinated [4]. Scheduling is highly complicated by the distributed ownership of the grid resources as consumers and providers of the grid resources have their own access policy, scheduling strategy and optimization objectives [5]. Grid schedulers should also support advanced features such as (i) user requested job priority (ii) advanced reservation of resources (iii) resource usage limits enforced by administrators (iv) user specifiable resource requirements etc.

There are a number of grid scheduling architectures available. For small set of machines, a centralized architecture with a single scheduler is enough, but it wouldn’t scale and not fault tolerant in a geographically distributed systems. User level grid schedulers are used to select the local schedulers to submit the applications

[6]. Another approach is one where grid schedulers are organized into a tree structure [7]. Like the above said architectures, several architectures are available to reduce the complexity of the problem for particular application scenarios. Generic features of enterprise grids, high performance computing grids and global grids have been identified to develop a scheduling instance for the scheduling solutions [8]. Even though different grid architectures exist, there are also some common features for all the grid schedulers. The grid schedulers deal with organizing the information providers in such a way that the users can have an easy access to the data. They can also recognize the file system or whether any type of resource is cached or which resource is rapidly available.

In the grid system, an end user submits the job that has to be executed with some constraints like job execution deadline, cost for the execution and the time required for the execution. Grid resource manager estimates the resource requirements and provides the functionality for discovery and publishing of resources as well as scheduling, submission and monitoring of jobs [9]. Thus different performance goals also play great impacts on the design of scheduling systems. Desirable performance goals of grid scheduling includes: maximizing system throughput [10], maximizing resource utilization, minimizing execution time [11], minimizing cost on the user side and fulfilling economic constraints [12]. Thus this survey focuses on scheduling algorithms in grid environment based on the above said characteristics and challenges of the grid. The remaining section of the paper is organized as follows. Section 2 deals with taxonomy of scheduling in grid environment. Section 3 deals with taxonomy of workflow scheduling in grid. Schedulers for grid environment are presented in Section 4. Section 5 explains Grid Simulation Tools. Applications are dealt in Section 6 and finally Section 7 gives the conclusion.

2. Taxonomy of Scheduling in Grid

Casavant et al's hierarchical taxonomy [13] for scheduling algorithms in general purpose parallel and distributed computing systems will be suitable for grid computing as grid is also a form of parallel and distributed system. The taxonomy is given in the figure 1.

Local Vs Global Scheduling

In general, at the highest level, scheduling algorithms can be classified into local and global scheduling algorithms. Local scheduling algorithms deal with allocation and execution of the processes that are resident on a single CPU. On the other hand, information about the system or resources is used to allocate the processes to multiple processors in global scheduling algorithms. As grid computing deals with coordination of multiple processors that are geographically distributed, grid scheduling falls under global scheduling

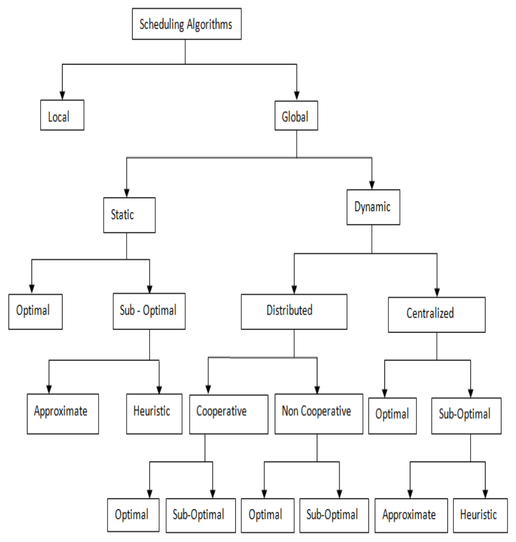

Fig. 1 Hierarchical Taxonomy of Scheduling Algorithms [13]

Static Vs Dynamic Scheduling

In the case of static scheduling, all the information regarding the tasks and resources such as execution time of the tasks, speed of the processor are available by the time the application is scheduled. In this type of scheduling, it is easy to program from the scheduler's point of view. But in the case of dynamic scheduling, the execution time of the tasks may not be known due to the direction of branches, number of iterations in the loop etc. So, the task has to be allocated on the fly as the application executes. Both static and dynamic scheduling are widely adopted in the grid. Here, system need not be aware of the run time behavior of the application before execution [14]. Static scheduling algorithms are presented in [15], [16], [17], [18] and dynamic scheduling is presented in [19], [20], [21], [22], [23] and [24]. Static scheduling will be useful for predictive analyzes, impact studies, postmortem analyzes etc. Here it is assumed that each machine executes a single task at a time in the order in which the tasks are assigned and the size of the meta task and the number of machines in the heterogeneous computing environment are static and are known priori [15]. Static scheduling algorithm presented in [16] uses a cluster of processors for critical path tasks which is useful where communication costs are not arbitrarily heterogeneous. Fault tolerant static scheduler that uses task duplication for grid applications has been presented in [17]. Application of swarm intelligence such as Ant Colony Optimization to grid scheduling is given in [18]. Dynamic scheduling algorithms should be able to solve job failure, unexpected arrival of high priority jobs, resource failure, activation of new resources, varying workload on resources, changing resource properties, changing priority or deadline of the job etc [19]. Dynamic scheduling algorithm for grid has been developed in [20] by incorporating the prediction strategy into a static scheduling algorithm. Wasp colony’s interaction with environment to dynamic job shop scheduling has been applied and a wasp based scheduling decision mechanism has been constructed in [21]. Scheduling policies for desktop grid systems involving different levels of heterogeneity has been developed in [22] that utilize the solution to a linear programming problem. Parallel applications consisting of independent tasks have been considered here. Aline P. Nascimento et al [23] has focused on scheduling policies of an application management system which is embedded into MPI applications. There are applications with large number of light weight jobs. The overall processing of these types of applications involves high overhead time and cost in terms of job transmission to and from grid resources and job processing at the grid resources. To overcome this difficulty, a dynamic job grouping mechanism has been presented in [24] based on the processing requirements of each application, grid resources availability and their processing capability.

Optimal Vs Sub-Optimal Scheduling

Static scheduling can be further classified into optimal and sub optimal scheduling. As already seen, in the case of static scheduling, all the information regarding the tasks and resources should be known before executing the application. In this case, an optimal assignment could be made based on some criterion function, such as minimum makespan or maximum resource utilization. But scheduling algorithms are NP-Complete as well as it is difficult to make reasonable assumptions in Grid scenarios, current research tries to find sub optimal solutions that can be further divided into two general categories i.e., approximate and heuristic [14].

Approximate Vs Heuristic Scheduling

Approximation algorithms are used to find approximate solutions to optimization problems. These algorithms are used for problems when exact polynomial time algorithms are known. All the approximation algorithms are not suitable for practical applications. Some approximation algorithms have impractical running times. Also, it applies to only optimization problems and not to pure decision problems. Approximation algorithms are used for global routing [25], scheduling in line networks [26], sensor networks [27], VLSI design etc. RR approximation algorithm is the first one used for grid scheduling [28]. Polynomial time approximation algorithms has been presented in [29] for multiprocessor scheduling under uncertainty. An approximation algorithm has been used in scheduling independent tasks into a parallel machine and extended to the grid environment [30]. Thus only a very few works are done for scheduling the tasks in the grid environment using approximation algorithms.

Heuristic scheduling algorithms which are also sub optimal are faster than cost and time intensive algorithms. Heuristic scheduling is widely used in grid environment due its advantage over time and cost. Metaheuristic approaches are the de facto approach as dealing with many constraints and optimization criteria in a dynamic environment is very complex and computationally hard [31]. The reason for the strength of the metaheuristic approaches include (i) meta heuristics are clear and well understood (ii) no need for optimal solutions (iii) efficient solutions in short time (iv) dealing with multiobjective nature (v) appropriateness for periodic and batch scheduling (vi) appropriateness for decentralized approaches (vii) hybridization with other approaches (viii) designing robust grid schedulers (ix) libraries and frameworks for meta heuristics [31]. Heuristic approaches in general may be local search based, population based or hybrid. Local search based heuristics consists of Simulated Annealing (SA), Hill Climbing, Tabu Search (TS) etc.

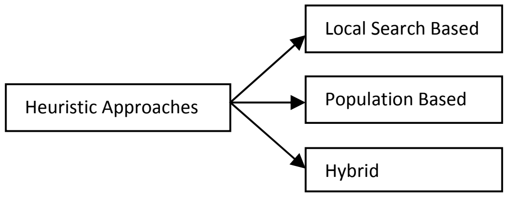

Fig. 2 Heuristic Based Scheduling

Genetic Algorithm (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) etc comes under Population based heuristics. The local search based heuristics and population based heuristics are combined to form hybrid heuristics.

In [32], heuristic algorithms based on list scheduling algorithms have been used to support co allocation and advance resource reservation. Opportunistic Load Balancing (OLB), Minimum Execution Time (MET), Minimum Completion Time (MCT), Min Min , Max Min are some of the heuristics dealt in [15]. In [33], SA has been used for data intensive grid applications where neighborhoods have been created using three perturbation schemes such as pair wise exchange, insertion and random insertion. SA is combined with Genetic Algorithm for scheduling in grids [34]. In [35], SA is used in the application scheduling in grid environment. Hill climbing has been used to determine migration route in decentralized job scheduling that involves many job migrations between neighboring grid nodes [36]. Many hill climbing versions such as steepest descent and steepest ascent are obtained by defining appropriate neighborhood relationships [31]. TS have been used to solve scientific workflow scheduling problem in grid which avoids entertainments in cycles by forbidding or penalizing moves, to take the solution in the next iteration [37]. In [38], TS has been used to create schedules in fuzzy as well as crisp modes. As normal genetic algorithm is very slow for any application due to large number of iterations, Urumchi et al in [39] has proposed a scheduling algorithm for grid using genetic algorithm with limited number of iterations. GA with multiple objectives has been presented for grid scheduling in [40] to minimize waiting time and makespan. Yang Gao et al in [41] have developed two algorithms with genetic algorithm using predictive models to schedule the jobs both at the system and application levels. V. Di Martino et al in [42] have suggested the use of local search strategy to improve the convergence quickly even when there are large number of problems as in real environment. A number of work using ant colony optimization technique has been done and in [43] Marilena et al has proposed an ant algorithm in terms of load balancing and waiting time in the queue based on different interpretation of pheromone trails. Fuzzy reputation has been aggregated to ACO to improve the decision of scheduling in grid as well as improves the velocity of execution by taking the historical and nodes information [44]. Wei Neng Chen et al in [45] have presented an ACO algorithm to schedule large scale workflows with various QoS parameters such as makespan, cost, deadline etc. Seven heuristics has been designed for the ACO approach and artificial ants were allowed to select heuristics based on pheromone values. Particle Swarm Optimization shows better results than the above said techniques and a number of works for grid scheduling has been done using this technique due to its simplicity, low computational burden and few control parameters. It has been found that the algorithm was stable and also presented low variability [46]. D.Y. Sha et al in [47] has used hybrid particle swarm optimization for job shop problem and modified the particle position based on preference list based representation, particle movement based on swap operator, and particle velocity based on tabu list to suit the algorithm for the job shop problem. The position and velocity of the particles in the conventional PSO is extended from real vectors to fuzzy matrices in [9]. PSO when applied to job shop scheduling problems, it results in quicker convergence and obtains faster solutions.

Distributed Vs Centralized Scheduling

In dynamic scheduling scenarios, the scheduling decisions may be made by one centralized scheduler or shared by multiple distributed schedulers. The centralized scheduler has the advantage of ease of implementation, but suffers from the lack of scalability and fault tolerance [14]. Shahram Amin et al in [48] have used distributed scheduler using fuzzy logic that considers the advantages of local clusters for executing the jobs. Xiangchun Han et al in [49] have proposed a two level centralized scheduling model. Qingjiang Wang et al in [50] have presented a de–centralized scheduling by composing a subgrid that consists of a node and its neighbors using distributed backfilling. Here, subgrids are overlapped with others so that the waiting jobs may be migrated around the grid. Manish Arora et al in [51] has given a de-centralized and scalable algorithm that overlaps the node coordination time with actual processing of ready jobs to save clock cycles used for making decisions.

Cooperative Vs Non Cooperative Scheduling

One process controls multiple cooperative threads in cooperative scheduling. Each grid scheduler in this case, has the responsibility to carry out its own portion of task scheduling. Cooperative scheduling architecture supports high scalability, ability to make decisions based on global state, free movement of jobs between sites based on local scheduler decision, easy integration of different gateways, independence on remote services and local summit, direct inclusion of virtualized resources [52]. Mustafizur Rahman et al in [53] have proposed a model for decentralized and cooperative workflow scheduling for grid environment that has been derived from distributed hash table. In non– cooperative scheduling, individual schedulers act as independent entities and arrive at their own decision regarding their objects.

3. Taxonomy of Workflow Scheduling in Grid

Scientists and engineers are building complex applications to manage and process large data sets through the advent of grid and application technologies. These types of applications require composition and execution of complex workflows. Scheduling workflows is a kind of global task scheduling as it focuses on mapping and managing the execution of inter dependent tasks on shared resources. Workflow scheduling has been classified based on architecture, decision making, planning, strategies and performance estimation [54].

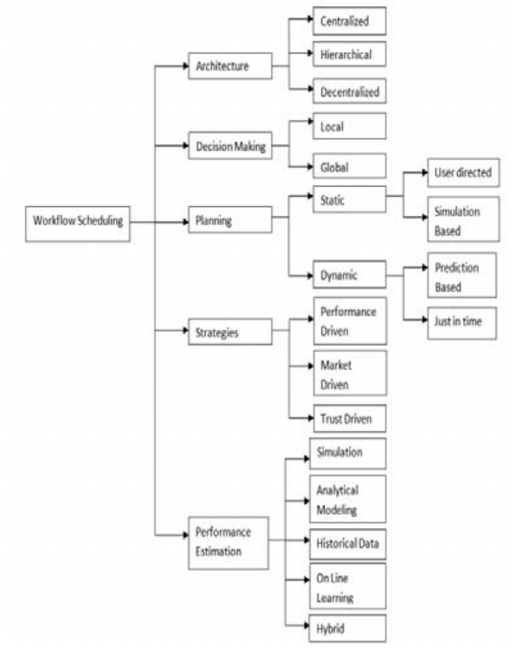

Fig. 3 Taxonomy of Workflow Scheduling [54]

In [54], Jia Yu et al has classified workflow scheduling from the view of scheduling architecture, decision making, planning scheme, scheduling strategy and performance estimation as shown in figure 3. In terms of architecture, workflow scheduling can be classified into centralized, hierarchical and decentralized scheduling architectures of which centralized scheduler is the most common. In the case of centralized scheduler, one central workflow scheduler makes decisions for all the tasks in workflow as it has information on all parts of workflow and can find the best schedule. A decentralized architecture has better scalability compared to centralized one, but it cannot plan for the entire workflow. Here, providers make decisions based on local, imperfect and delayed information. In the case of hierarchical architecture, it can improve the performance and achieve a reasonable schedule even though it cannot protect against failure of root. Decision making can be of either local or global where local decision based scheduling deals with only one task at a time and thereby gives a best schedule only for that particular task and thus reduces the performance of entire workflow whereas global decision based scheduling deals with all the tasks of the workflow and gives better schedule for the entire workflow. Abstract workflows are translated to concrete workflows through planning scheme which is further classified into static and dynamic. Scheduling decisions are taken before execution starts in static scheme whereas in dynamic scheme scheduling decisions are taken at run time. Static scheme includes user directed and simulation based scheduling. In user directed scheduling scheme, the user specifies the allocation of resources to the processes and in the case of simulation based scheduling scheme, the allocation of the resources to the processes is automated which can be centralized, mediated or peer to peer. Dynamic based scheduling includes prediction based and just in time scheduling. Prediction based dynamic scheduling uses dynamic information with previous results based on prediction. Here the scheduling is made before the start of the execution. But in the case of just in time scheduling, decision regarding scheduling of tasks is taken at the time of execution. Likewise, scheduling strategies are classified into performance driven, market driven and trust driven. Based on performance estimation, workflow scheduling is categorized into simulation, analytical modeling, historical data, online learning and hybrid.

M.Wieczorek et al in [55] has given taxonomy on multi criteria grid workflow scheduling and it is shown in the following figure 4. They have classified scheduling process based on criteria multiplicity, workflow multiplicity, and dynamism and advance reservation. Criteria multiplicity is categorized into single

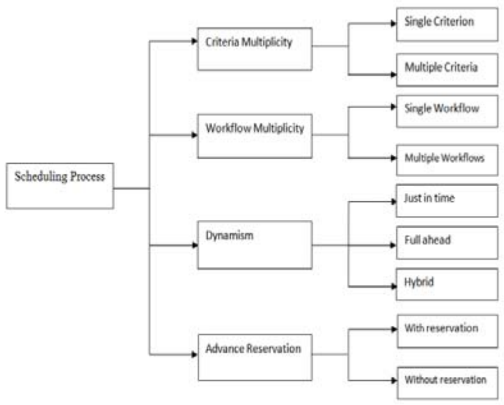

Fig. 4 A Taxonomy of Multi criteria Workflow Scheduling [55]

criterion and multiple criteria. Optimization of the scheduling process is based on single criterion such as makespan in single criterion scheme whereas in multiple criteria scheme, optimization is based on multiple criteria such as makespan, cost, deadline, reliability etc. M.Wieczorek et al in [56] have presented bi criteria scheduling for scientific workflows on the grid. Saurabh Kumar Garg et al in [57] have presented heuristics for scheduling parallel applications on utility grids to manage and optimize the trade off between time and cost constraints. Wei Neng Cheng et al in [45] have used various criteria for scheduling such as makespan, cost, deadline, reliability etc. Rizos Sakellariou et al have considered a basic model for workflow applications modeled as DAGs and investigated heuristics to schedule the tasks to the resources in a way that satisfies a budget constraint with optimization for overall time [58]. Considering workflow multiplicity, it is categorized into single workflow and multiple workflows. Optimization of execution of a single workflow and multiple workflows within a single scheduling process is considered for single and multiple workflow categories. Most of the papers dealt with optimization of single workflow and a few concentrated on optimization of multiple workflows. Some research efforts have been made to merge multiple workflows into single workflow before execution without considering the submission time of the workflows by users. A planner guided dynamic scheduling strategy for multiple workflows has been presented in [59] by leveraging job dependence information and execution time estimation. Based on dynamism, scheduling process is classified as just in time, full ahead and hybrid scheduling. Just in time scheduling is a dynamic based scheduling where the scheduling decision for an individual task is postponed to the maximum extent, and performed before the execution starts. Full ahead scheduling is a static scheduling and the entire workflow is scheduled before the execution starts. Both these scheduling are combined to perform hybrid scheduling. Just in time service is utilized to reduce the interaction between the users and the system's resources by adjustment of resources to request as well as prepare services prior to the users' requests [60]. In [61], just in time scheduling has been used to schedule tasks in the workflow. Failure of tasks has been handled by resubmitting the tasks to another resource , which did not have a failure history for those tasks. Dietmar Sommerfeld et al in [62] have used list scheduling heuristic to perform full ahead planning of workflow tasks based on prediction of execution times and also distributed jobs in the grid just in time. Advance reservations of tasks to the resources have been considered in many papers. Junzhou Luo et al in [63] have proposed architecture for advance reservation to adapt to dynamic behavior of grid and to solve the imprecise denial of reservation request problem. Marek Wieczorek et al in [64] have devised and implemented advance reservation based on fair share principle as part of the scheduling and resource management services of grid application development. Philipp Wieder et al in [65] have used WS-Agreement to negotiate advance reservation of resources to execute workflow components. A dynamic priority scheduler for advance reservation has been presented in a space shared environment to coordinate resource sharing in distributed grid computing environments [66].

4. Schedulers for Grid Environment

A number of tools have been developed for scheduling in grid computing systems. Some of the tools include Nimrod-G, Condor-G, GRaDS, Legion, NetSolve, Sun Grid Engine etc.

Nimrod-G

Nimrod was the first tool to use heterogeneous resources in a grid which was created as a research project funded by the Distributed Systems Technology Centre. Nimrod-G is a grid aware version of Nimrod which takes advantage of the features such as automatic discovery of resources in globus toolkit. It is used for automated modeling and execution of parameter sweep applications in grid. It uses resource management and scheduling algorithms based on economic principles and also supports user defined deadline and budget constraints. The components of Nimrod-G include client or user station, parametric engine, scheduler, dispatcher and job-wrapper [12]. Client or User Station acts as an user interface for controlling an experiment under consideration. It is also possible to run multiple instances of the same client at different locations. Parametric engine is a central component that controls the entire experiment. It also maintains the state of the whole experiment and it is recorded in persistent storage so that it allows the experiment to be restarted if the node running nimrod goes down. Next comes the scheduler which is responsible for resource discovery, resource selection and job assignment. According to scheduler's instruction, the dispatcher initiates the execution of a task on the selected resource. Job wrapper starts execution of the task on the assigned resource and sends the result back to the parametric engine via dispatcher. David Abramson et al in [67] has used Nimrod-G to manage all operations associated with remote execution including resource discovery, trading, scheduling etc.

Condor-G

Condor is a specialized workload management system for compute intensive jobs that provides a job queuing mechanism, scheduling policy, priority scheme, and resource monitoring and resource management. It is used to manage a cluster of dedicated compute nodes and also exploits wasted CPU power from idle desktop workstations. It can be used to seamlessly combine all of an organization's computational power into one resource. It is the product of Condor Research Project at the University of Wisconsin-Madison and was first installed as a production system in the UW-Madison Department of Computer Sciences. Condor incorporates many of the emerging grid-based computing methodologies and protocols. Condor-G is fully interoperable with resources managed by Globus [68]. When a user submits a job to the Condor, it is executed on a remote machine within the pool of machines available to Condor where security of the remote machines is preserved by Condor through remote system calls. Condor can be useful on a range of small to large network sizes. On a single machine, it pauses the job when the user uses the machine for other purposes, and it restarts the job if the machine reboots. On a small dedicated cluster, it functions as a cluster submission tool. James Frey et al in [69] has used Condor G for handling job management, resource selection, security and fault tolerance.

GRaDS

Grid Application Development Software (GRaDS) Project with support from NSF Next Generation Software Program has been developing tools for construction of applications on the grid easier. This led to the development of a prototype software infrastructure called GrADSoft that runs on top of Globus and facilitates scheduling, launching and performance monitoring of tightly coupled Grid applications. In GrADS, the end user just submits their parallel application to the framework for execution. The framework schedules the application to appropriate set of resources, launching and monitoring the execution and also rescheduling the applications on different set of resources if necessary. Anirban Mandal et al has launched and executed EMAN, a Bio imaging workflow application onto the grid [70]. F.Berman et al in [71] has presented an extension to GrADS software framework for scheduling workflow computations that has been applied to a 3-D image reconstruction application etc.

Legion

Legion is an object based, meta systems software project at the University of Virginia that began in late 1993. It has been created to address key issues such as scalability, programming ease, fault tolerance, site autonomy, security etc. It was also designed to support large degrees of parallelism in application code and manage the complexities of the physical system for the user. It also allows applications developers to select and define system-level responsibility. Anand Natrajan et al in [72] has presented a grid resource management of legion for scheduling all compute objects as well as data objects on machines whose capabilities match the requirements, while preserving site autonomy as well as recognizing usage policies.

NetSolve

The NetSolve system from the University of Tennessee's Innovative Computing Laboratory was to address the ease of use, portability and availability of optimized software libraries for high performance computing. It enables users to solve complex scientific problems remotely, by managing networked computational resources and using scheduling heuristics to allocate resources to satisfy the requests [73].
The Primary goal of NetSolve was to make an easy access to grid resources. NetSolve, which is a client-agent-server system provides remote access to hardware as well as software resources. The agent maintains a list of all available severs and performs resource selection for client requests and also ensures load balancing of the servers. Eventhough locating appropriate resources to the request is a challenge in grid computing, the NetSolve agent uses knowledge of the requested service, information about the parameters of the service request from the client, and the current state of the resources to get possible servers and return the servers in sorted order [74].

Sun Grid Engine

Sun Grid Engine (SGE) is the foundation of Sun Grid Utility Computing system, made available over Internet in the United States in 2006, later available in many other countries. It is used on high performance computing cluster is used for accepting, scheduling, dispatching and managing remote and distributed execution of large numbers of standalone or parallel user jobs. It also schedules the allocation of distributed resources such as processors, memory, disk space etc. Some of the features of SGE include advance reservation of resources, multi clustering, job submission verifier on both client and server sides, topology aware scheduling, job and scheduler fault tolerance etc. Goncalo Borges et al in [75] has presented a work developed to integrate SGE with the EGEE (Enabling Grids for E-Science) middleware. EGEE is the world's largest operating grid infrastructure serving thousands of multi science users with robust, reliable and secure grid services worldwide.

5. Grid Simulation Tools

As grid computing is more loosely coupled, heterogeneous and geographically dispersed, getting all these resources working together for a single scientific or business problem may be difficult for research purpose which requires repeated evaluation of some strategies. Here comes the simulators which provide users with practical feedback when developing real world systems. This allows the developer to determine correctness and efficiency of the proposed system before it is actually constructed and also overall cost of developing the real system also diminishes. There are many simulation tools for grid computing such as simgrid, gridsim, optorsim, bricks etc available for evaluating applications and network services for grid systems.

SimGrid

The SimGrid project was started in 1999 which provides core functionalities for simulation of distributed applications in heterogeneous distributed environments. The main aim of the project was to facilitate research in the area of distributed and parallel application scheduling on distributed computing platforms that were ranging from simple network of workstations to computational grids. SimGrid v1 prototyped scheduling heuristics and SimGrid v2 extended the capabilities of its predecessor by transitioning from a wormhole model to analytical one. Application Programming Interface was also added to study non-centralized scheduling and other kinds of concurrent sequential processes. GRAS(Grid Reality and Simulation) API was added in v3.0 where distributed applications can be developed within the simulator. Henri Casanova in [76] has used Simgrid for the study of scheduling algorithms for distributed application.

GridSim

Since in grid environment, resources abd users are distributed across multiple organizations with their own policies, it is impossible to perform scheduler performance evaluation in a repeatable and controllable manner, Rajkumar Buyya et al has developed java based discrete-event grid simulation toolkit [77]. The toolkit allows modeling and simulation of entities in parallel and distributed computing systems users, applications, resources and resource broker for design as well as evaluation of scheduling algorithms. Some of the functionalities of gridsim include incorporating failures of grid resources during runtime, suppoting advance reservation of a grid system, incorporating auction model, incorporating extension of datagrid into GridSim, incorporating network extension into GridSim etc.

OptorSim

A grid simulator designed to test dynamic replication strategies and appropriate scheduling of jobs was developed as a part of European Data Grid project. OptorSim which has the structure of EDG, includes the following elements to achieve a realistic simulated environment. These include storage resources where data can be kept, computing resources to which jobs can be sent, scheduler to decide to which resource the job has to be sent, the network which connects the sites and finally replica management. It also incorporates peer to peer messaging system which is used by some of the optimization algorithms for conducting auctions [78].

Bricks

It is a java based performance evaluation system for scheduling algorithms and frameworks of high performance global computing systems. It consists of a scheduling unit that allows simulation of various behaviors of resource scheduling algorithms, programming modules for scheduling, processing schemes for networks and servers etc. Users can also construct and alter the script using building bricks within the script for testing and evaluating simulations [79].

6. Applications

There are many scientific problems that require grid environment to get solved. It provides an environment to solve problems in physics, chemistry, nuclear fusion, earth science, space, human health, agriculture, medicine, education, research etc. In medical and biomedical fields, grid computing is useful in digital x-ray image analysis, radiation therapy simulation and protein folding. In chemistry, problems related to quantum chemistry, organic chemistry and polymer modeling makes use of grid computing. In physics, high energy physics, theoretical physics, lattice calculations, combustion and neutrino physics use grid environment.

7. Conclusion

This paper has given a detailed study on scheduling with its evolution, application and importance. Also, in this paper, a study on taxonomy of scheduling has been given in different perspectives and the performance of scheduling algorithms has been discussed. The softwares that support scheduling in real grid environment as well as in simulated environment are also given. Thus this paper gives a detailed survey on scheduling with its applications.

It is observed that heuristic based algorithms and in particular, population based heuristics are most suitable for scheduling the tasks in the grid environment. But there are population based heuristics which are complex in nature and takes a long execution time. For instance, ant colony optimization, when run in a normal PC, it takes hours to execute an algorithm to schedule more than 1000 processes. This algorithm even though gives better results compared to other population based heuristics such as genetic algorithm, due to its longer execution time of the algorithm, some other algorithm which executes faster has to be considered. Particle Swarm Optimization, Frog Leap Algorithm are some of the population based heuristics which can be used for scheduling in grid.

References

[1] Patrick Weaver, "A Brief History of Scheduling – Back to the Future," *myPrimevera06*, Hyatt, Canberra, 4-6 April 2006.

[2] Harumasa Tada, Makoto Imase, Masayuki Murata, "On the Robustness of the Soft State for Task Scheduling in Large-scale Distributed Computing Environment," *Proceedings of the International Multiconference on Computer Science and Information Technology*, pp. 475 -480, 2008.

[3] Ian Foster, Carl Kesselman, *The Grid: Blueprint for a new computing infrastructure*, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2003.

[4] Miguel L. Bote-Lorenzo, Yannis A. Dimitriadis, and Eduardo Gomez-Sanchez, "Grid Characteristics and Uses: a Grid Definition," *In the Postproc. Of the first European Across Grid Conference (ACG'03),* Springer-Verlag LNCS 2970, pp.219-298, Santago de Compostela, Spain, Feb. 2004.

[5] Yanmin Zhu, Lijuan Xiao, Lionel M. Ni, Zhiwei Xu, "Incentive Based P2P Scheduling in Grid Computing," *Grid and Cooperative Computing, LNCS 3251,Springer Verlag,* pp. 209 – 216, 2004.

[6] Fran Berman, Rich Wolski, Silvia Figueira, Jennifer Schopf, and Gary Shao, "Application-Level Scheduling on Distributed Heterogeneous Networks," *In Supercomputing '96*, 1996.

[7] V. Hamscher, U.Schwiegelshohn, A.Streit, and R.Yahyapour, "Evaluation of job-scheduling strategies for Grid Computing," *1st International Workshop on Grid Computing*, vol. LNCS 1971, pages 191-202, 2000.

[8] N.Tonellotto, R.Yahyapour, Ph.Wieder, "A Proposal for a Generic Grid Scheduling Architecture," *CoreGRID Technical Report, Number TR-0015,* Jan.11, 2006.

[9] Ajith Abraham, Hongbo Liu,Weishi Zhang , and Tae-Gyu Chang, "Scheduling Jobs on Computational Grids Using Particle Swarm Algorithm," *10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, UK,2006, pp.500-507.

[10] Michael Litzkow, Miron Livny, and Matt Mutka, "Condor-A Hunter of Idle Workstations," *In Proceedings of Eigth International Conference of Distributed Computing Systems*, California, June, 1988, pp.204-111.

[11] David Abramson, Rok Sosic, J. Giddy, B. Hall, "Nimrod: A tool for performing parameterised simulations using distributed workstations," *In HPDC*, pages 112–121, 1995

[12] Rajkumar Buyya, David Abramson, Jonathan Giddy, "Nimrod/G: An Architecure for a Resource Management and Scheduling System in a Global Computational Grid," *The 4th International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2000),* May 2000, Beijing, China. IEEE Computer Society Press, USA

[13] T.Casavant, and Kuhl, "A Taxonomy of Scheduling in General – purpose Distributed Computing Systems," *IEEE Trans. on Software Engineering*, vol. 14, no. 2, pp. 141 – 154, Feb. 1988.

[14] Fangpeng Dong and Selim G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems," *Technical Report No.2006 – 504, School of Computing, Queen's University, Kingston, Ontario,*Jan. 2006.

[15] R. Braun, H.Siegel, N.Beck. L. Boloni, M.Maheswaran, A.Reuther, J.Robertson, M.Theys, B. Yao, D.Hensgen and R.Fruend, "A Comparison of Eleven Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems," *International Journal of Parallel and Distributed Computing,* vol. 61, no. 6, pp. [1] 810 – 837, 2001.

[16] Junghwan Kim, Jungkyu Rho, Jeong-Ook Lee and Myeong-Cheol Ko, "CPOC: Effective Static Scheduling for Grid Computing," *LNCS: High Performance Computing and Communiations*, Springer Berlin/Heidelberg, vol. 3726/2005, Oct. 2005.

[17] Fechner, B. Honig, U. Keller, J. Schiffmann, W. "Fault Tolerant Static Scheduling for Grids," *IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2008*, 2008.

[18] John Levine and Graham Ritchie, "Swarm Intelligence and Static Heterogeneous Multi-Processor Scheduling," *Swarm Intelligence and Grid Computing: Applications in Grid Scheduling and e-Science,* Centre for Intelligent Systems and their Applications, School of Informatics, University of Edinburgh.

[19] Maria Chtepen, "Dynamic Scheduling in Grid Systems," *Sixth FirW PhD Symposium,* Faculty of Engineering, Ghent Univeristy, 30th November 2005, paper nr.110

[20] Nguyen The Loc, Said Elnaffar, "A Dynamic Scheduling Algorithm for Divisible Loads in Grid Environments," *Journal of Communications,* vol. 2, no. 4, June 2007.

[21] Bao Zhen-qiang, Li Xiang-Qing, Zhang Dan, Wang Peng, Gao Kai-zhou, "Wasp Algorithm for Dynamic Scheduling Based on Manufacturing Grid," isip, pp.83-87, 2008 International Symposiums on Information Processing, 2008

[22] I.AI-Azzoni, D.G.Down, "Dynamic Scheduling for heterogeneous desktop grids," *9th IEEE/ACM International Conference on Grid Comptuing,* pages 136 – 143, 2008.

[23] Aline P. Nascimento, Alexandre C.Sena, Cristina Boeres, Vinod E.F. Rebello, "Distributed and Dynamic Self Scheduling of Parallel MPI Grid Applications," *Concurrency and Computation:Practice and Experience*, vol. 19, no. 14, 2007.

[24] N.Muthuvelu, J.Liu, N.L. Soe, S.R.Venugopal, A.Sulistio and R.Buyya, "A Dynamic Job Grouping Based Scheduling for Deploying Applications with Fine Grained Tasks on Global Grids," *Proceedings of the 3rd Australian Workshop on Grid Computing and e-Research (AusGrid 2005),* Newcastle, Australia, Jan. 30 –Feb.4, 2005.

[25] Christoph Albrecht, "Global Routing by New Approximation Algorithms for Multicommodity Flow," *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems,* vol.20, no. 5, May 2001.

[26] Harald Racke, Adi Rosen, "Approximation Algorithms for Time Constrained Scheduling on Line Networks," *WOODSTOCK'09,* EI Paso, Texas, USA, 2009.

[27] Patrik Floreen, Petteri Kaski, Topi Musto, Jukka Suomela, "Local Approximation Algorithms for Scheduling Problems in Sensor Networks," *LNCS:Algorithmic Aspects of Wireless Sensor Networks,* Springer Berlin/Heidelberg, vol. 4837/2008, 2008.

[28] Noriyuki Fujimoto, Kenichi Hagihara, "A Comparison among Grid Scheduling Algorithms for Independent Coarse-Grained Tasks," saint-w, pp.674, 2004 Symposium on Applications and the Internet-Workshops (SAINT 2004 Workshops), 2004

[29] Guolong Lin, Rajmohan Rajaraman, "Approximation Algorithms for Multiprocessor Scheduling under Uncertainty," *19th Annual ACM Symposium on Parallel Algorithms and Architectures, San Diego, California, USA,* pages 25 – 34, 2007.

[30] Fujimonto, N. Hagihara, K. , "A 2-Approximation Algorithm for Scheduling Independent Tasks onto a Uniform Parallel Machine and its Extension to a Computational Grid," *IEEE International Conference on Cluster Computing,* Barcelona, pages 1-7, 2006.

[31] Fatos Xhafa, Ajith Abraham, "Meta-heuristics for Scheduling in Distributed Computing Environment," *SCI 146,* pp. 1-37, 2008.

[32] Joerg Decker, Joerg Schneider, "Heuristic Scheduling of Grid Workflows Supporting Co-Allocation and Advance Reservation," *Seventh IEEE International Symposium on Cluster Computing and the Grid, CCGRID,* pages 335-342, 2007.

[33] S.Benedict, V.Vasudevan, "Scheduling of Scientific Workflows Using Simulated Annealing Algorithm for Computational Grids," *International Journal of Soft Computing 2(5),* 606 – 611, 2007.

[34] Shajulin Benedict, Rejitha R.S., Vasudevan V, "An Evolutionary Hybrid Scheduling Algorithm for Computational Grids," *Journal of Advanced Computational Intelligence and Intelligent Informatics,* vol. 12, no. 5, pp. 479 – 484, 2008.

[35] Asim YarKhan, Jack Dongarra, "Experiments with Scheduling Using Simulated Annealing in a Grid Environment," *Lecture Notes in Computer Science,* vol. 2536, *Third International Workshop on Grid Computing,* pages 232 – 242, 2002.

[36] Qingjiang Wang, Yun Gao, Peishun Liu, "Hill Climbing – Based Decentralized Job Scheduling on Computational Grids," *First International Multi – Symposiums on Computer and Computational Sciences,* vol. 1, pages 705 – 708, 2006.

[37] Shajulin Benedict, V.Vasudevan, "Improving Scheduling of Scientific Workflows Using Tabu Search for Computational Grids," *Information Technology Journal 7(1),* 91 – 97, 2008.

[38] Fayad C, Garibaldi J.M., Ouelhadj D, "Fuzzy Grid Scheduling Using Tabu Search," *IEEE Internationl Fuzzy Systems Conference, FUZZ-IEEE 2007,* pages 1-6, 2007.

[39] Urumchi, Xinjiang, "An Improved Genetic Algorithm with Limited Iteration for Grid Scheduling," *Sixth International Conference on Grid and Cooperative Computing (GCC 2007),* pp. 221-227Aug. 16-18, 2007.

[40] Siriluck, Lorpunmanee, Mohd Noor Md Sap, Abdul Hanan, Abdullah, Surat Srinoy, "Genetic Algorithm in Grid Scheduling with Multiple Objectives," *5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Databases,* pages 429-435, 2006.

[41] Yang Gao, Hongqiang Rong, Joshua Zhexue Huang, "Adaptive Grid Job Scheduling with Genetic Algorithms," *Future Generation Computer Systems,* 21, pages 151 – 161, 2005.

[42] V.Di Martino, M.Mililotti, "Sub Optimal Scheduling in a Grid Using Genetic Algorithms," *Parallel and Nature Inspired Computational Paradigms and Applications,* vol. 30, issue 5-6, pages 553 – 565, May 2004.

[43] Marilena Bandieramonte, Antonella Di Stefano, Giovanni Morana, "An ACO Inspired Strategy to Improve Jobs Scheduling in a Grid Environment," *LNCS, Springer Berlin/Heidelberg, Algorithms and Architectures for Parallel Processing,* vol. 5022, pages 30 – 41, 2008.

[44] Zhurong Zhou, Wei Deng, Linrui Lu, "A Fuzzy Reputation Based Ant Algorithm for Grid Scheduling," *International Joint Conference on Computational Sciences and Optimization,* vol. 1, pages 102–104, 2009.

[45] Wei Neng Chen, Jun Zhang, "An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem With Various QoS Requirements," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews,* vol. 39, issue 1, pages 29 – 43, 2009.

[46] Yan-ping Bu, Wei Zhou, Jin-shou Yu, "An Improved PSO Algorithm and its Application to Grid Scheduling Problem," iscsct, vol. 1, pp.352-355, 2008 International Symposium on Computer Science and Computational Technology, 2008

[47] D.Y. Sha, Cheng-Yu Hsu, "A Hybrid Particle Swarm Optimization for Job Shop Scheduling Problem," *Computers and Industrial Engineering,* vol. 51, issue 4, pages 791 – 808, Dec 2006.

[48] Shahram Amin, Mohammad Ahmadi, "Distributed Resource Scheduling in Grid Computing Using Fuzzy Approach," *Recent Advances in Computer Engineering, Proceedings of the 12th WSEAS International Conference on Computers, Greece,* pages 820 – 825, 2008

[49] Xiangchun Han, Duanjun Chen, Jing Chen, "One Centralized Scheduling Pattern for Dynamic Load Balance in Grid," *International Forum on Information Technology and Applications, Chengdu,* vol. 2, pages 402-405, May 2009.

[50] Qingjiang Wang, Xiaolin Gui, Shouqi Zheng, Bing Xie, "De-Centralized Job Scheduling on Computational Grids Using Distributed Backfilling," *LNCS, Springer Berlin/Heidelberg, Grid and Cooperative Computing,* vol. 3251, pages 285 – 292, 2004.

[51] Manish Arora, Sajal K. Das, Rupak Biswas, "A De-Centralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments," *International Conference on Parallel Processing Workshops,* pp. 499, 2002.

[52] L.Matyska, M.Ruda, S.Toth, "Peer to Peer Cooperative Scheduling Architecture for National Grid Infrastructure," *ISGC 2010, CESNET,* March 2010.

[53] Mustafizur Rahman, Rajiv Ranjan, Rajkumar Buyya, "Cooperative and Decentralized Workflow Scheduling in Global Grids," *Future Generation Computer Systems,* 26, pages 753 – 768, 2010.

[54] Jia Yu, Rajkumar Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing," *Journal of Grid Computing,* vol. 3, issue 3 - 4, pages 29, 2005.

[55] M.Wieczorek, R.Proden, A.Hoheisel, "Taxonomies of the Multi – criteria Grid Workflow Scheduling Problem," *CoreGRID Technical Report, Number TR - 0106,* August 30, 2007.

[56] M.Wieczorek, S.Podlipnig, R.Prodan, T.Fahringer, "Bi-criteria Scheduling of Scientific Workflows for the Grid," *8th IEEE International Symposium on Cluster Computing and the Grid, CCGRID '08,* pages 9 -16 2008.

[57] Saurabh Kumar Garg, Rajkumar Buyya, H.J. Seigel, "Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management," *Thirty Second Australian Computer Science Conference (ACSC 2009),* Australia, *Conferences in Research and Practice in Information Technology,(CRPIT),* vol. 91, Bernard Mans, Ed.

[58] Rizos Sakellariou, Henan Zhao, Eleni Tsiakkouri, Marios D. Dikaiakos. "Scheduling Workflows with Budget Constraints," In Sergei Gorlatch, Marco Danelutto (Eds.), *Integrated Research in GRID Computing* (CoreGRID Integration Workshop 2005, Selected Papers), Springer-Verlag, CoreGRID Series, 2007, pp. 189-202 (ISBN 978-0-387-47656-8).

[59] Zhifeng Yu, Weisong Shi, "A Planner – Guided Scheduling Strategy for Multiple Workflow Applications," *International Conference on Parallel Processing - Workshops,* pp. 1 – 8, September 2008.

[60] Wail M. Omar, A.Taleb –Bendiab, Yasir Karam, "Autonomic Middleware Services for Just-In-Time Grid Services Provisioning," *Journal of Computer Science 2 (6),* pages 521 – 527, 2006.

[61] Suraj Pandey, William Voorsluys, Mustafizur Rahman, Rajkumar Buyya, James Dobson, Kenneth Chiu, "A Grid Workflow Environment for Brain Imaging Analysis on Distributed Systems," *Concurrency and Computation: Practice and Experience,* 2009.

[62] Dietmar Sommerfeld, Harald Richter, "Problems and Approaches of Workflow Scheduling in MediGRID," *e-science,* pp. 223-230, 2009, *Fifth IEEE Conference on e-science,* 2009.

[63] Junzhou Luo, Zhiang Wu, Jiuxin Cao, Tian Tian, "Dynamic Multi-Resource Advance Reservation in Grid Environment," *The Journal of SuperComputing,* Springer Netherlands, 2008.

[64] Marek Wieczorek, Mumtaz Siddiqui, Alex Villazon, Radu Prodan, Thomas Fahringer, "Applying Advance Reservation to Increase Predictability of Workflow Execution on the Grid," *Second IEEE International Conference on e-Science and Grid Computing(e-Science '06)*, Amsterdam, Netherlands, pp. 82, Dec. 04 – 06, 2006.
[65] Philipp Wieder, Ramin Yahyapour, Oliver Waldrich, Wolfgang Ziegler, "Improving Workflow Execution through SLA-based Advance Reservation," *CoreGRID TR-0053*, December 29, 2006.
[66] Ravin Ahuja, G.Gabrani, Asok De, "A Dynamic Priority Scheduler for Advance Reservation in Grid Computing," *International Journal of Soft Computing*, vol. 4, issue 2, pages 60 – 67, 2009.
[67] David Abramson, Rajkumar Buyya, and Jonathan Giddy, "A Computational Economy for Grid Computing and its Implementation in the Nimrod-G Resource Broker," *Future Generation Computer Systems (FGCS)*, 2002
[68] http://www.cs.wisc.edu/condor
[69] James Frey, Todd Tannenbaum, Miron Livny, Ian Foster, Steven Tuecke, " Condor-G: A Computation Management Agent for Multi Institutional Grids," *Cluster Computing, Springer Netherlands*, vol. 5, no. 3, July 2002, pages 237 – 246.
[70] Anirban Mandal, Anshuman Dasgupta, Ken Kennedy, Mark Mazina, Charles Koelbel, Gabriel Marin, Keith Cooper, John Mellor-Crummey, Bo Liu, Lennart Johnsson, "Scheduling Workflow Applications in GrADS," *2004, IEEE International Symposium on Cluster Computing and the Grid*, 2004.
[71] F.Berman, H.Casanova, A Chien, K.Cooper, H.Dail, A.Dasgupta, W.Deng, J.Dongarra, L.Johnsson, K.Kennedy, C.Koelbel, B.Liu, X.Liu, A.Mandal, G.Marin, M.Mazina, J.Mellor-Crummey, C.Mendes, A.Olugbile, M.Patel, D.Reed, Z.Shi, O.Sievert, H.Xia, A.Yarkhan, "New Grid Scheduling and Rescheduling Methods in the GRaDS Project," *International Journal of Parallel Programming, Springer Netherlands*, vol. 33, nos. 2-3, June 2005, pages 209 – 229.
[72] Anand Natrajan, Marty A.Humphery, Andrew S.Grimshaw, "Grid Resource Management in Legion," *Grid Resource Management: State of the Art and Future Trends*, 2004, pages 145 – 160.
[73] Dorian C.Arnold, Henri Casanova, Jack Dongarra, "Innovations of the NetSolve Grid Computing System," *Concurrency and Computation: Practice and Experience*, 2002, pages 1-23.
[74] Keith Seymour, Asim Yarkhan, Sudesh Agrawal, Jack Dongarra, "NetSolve: Grid Enabling Scientific Computing Environments," *In Grid Computing and New Frontiers of High Performance Processing*, vol. 14 of *Advances in Parallel Computing*, 2005.
[75] Goncalo Borges, Mario David, J Homes, Carlos Fernandez, Javier Lopez Cacheiro, Pablo Rey Mayo, Alvaro Simon Garcia, Dave Kant, Keith Sephton, "Sun Grid Engine, a New Scheduler for EGEE Middleware," Iberian Grid Infrastructure Conference – IBERGRID, May 2007.
[76] Henri Casanova, "Simgrid: a Toolkit for the Simulation of Application Scheduling," *Proceedings of the first IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, 2001.
[77] Rajkumar Buyya, Manzur Murshed, "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing," *Concurrency and Computation :Practice and Experience*, 14, 2002, pages 1175 – 1220.
[78] David G.Cameron, Ruben Carvajal-Schaffino, A.Paul Millar, Caitriana Nicholson, Kurt Stockinger, Floriano Zini, "Evaluating Scheduling and Replica Optimisation Strategies in OptorSim," *International Conference on Grid Computing, Proceedings of the 4th International Workshop on Grid Computing*, 2003, page 52.
[79] Atsuko Takefusa, Satoshi Matsuoka, Kento Aida, Hidemoto Nakada, Umpei Nagashima, "Overview of a Performance Evaluation System for Global Computing Scheduling Algorithms," *Proceedings of the 8th IEEE International Symposium on High Performance Distributed Comnputing*, 1999, page 11.

Author Profile

Dr.K.Vivekanandan obtained Ph.D in Computer Science in 1996. He has produced 10 Ph.D candidates and 7 M.Phil candidates. He has published articles in 16 national and international journals. His area of interest includes data mining and information systems.

D.Ramyachitra completed MCA from Madras University in 2000 and M.Phil in Computer Science in 2004 from Bharathiar University. She is currently pursuing Ph.D in Computer Science in Bharathiar University. She has published papers in 7 national and international Conferences and in 5 national and international journals. Her area of interest includes grid computing and image processing.