ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2 Issue 8, August 2013

HEURISTICS IN GRID SCHEDULING

D. Thilagavathi[1] and Dr. Antony Selvadoss Thanamani[2]

[1]Research Scholar [2]Professor and Head,
Research Department of Computer Science,
NGM College, Pollachi, Tamilnadu.

Abstract— **Grid** ***computing aggregates heterogeneous resources for executing resource intensive applications of science and engineering. Grid scheduling is a complex problem (NP-hard) to map existing tasks to accessible storage and computational resources to utilize them in the optimistic manner. In this article, we have analyzed the grid scheduling problem, the process of job scheduling, challenges and approaches for grid scheduling. We present an experimental outcome of the investigation into the use of Heuristic algorithms to optimize the Job Scheduling Problem in Grid instead of the traditional optimization method.***

Index Terms— **Grid scheduling, NP-hard, Heuristic, Job scheduling**

I. INTRODUCTION

"A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities" [8]. A grid environment can be seen as a collection of nearly unlimited computational resources that are working under integrated and collaborative environment. Grid computing concept uses the multi-owner resources to solve large problems in the field of science, engineering and commerce. To achieve the promising potential of collective resources in the efficient manner, effective and efficient scheduling algorithms for resource sharing are required exclusively. The available algorithms which run on the client-server architecture and on the homogeneous and dedicated resources like computer clusters are not sufficient enough to run the grid scheduling environment [3].

II. OVERVIEW OF GRID SCHEDULING

A Grid scheduler (or broker) must make resource selection decisions in an environment where it has no control over the local resources, the resources are distributed, and information about the systems is often limited or dated. Here, schedulers are responsible for job management like allocating resources needed by particular job, partitioning of the job to run the job in the parallel manner in parallel processing environment, management of data, event correlation and service-level management capabilities. A Grid scheduler is different from local scheduler in that a local scheduler only manages a single site or cluster and usually owns the resource.

A. A Generic Grid Scheduler Architecture

The generic Grid scheduler architecture is described in Figure 1. A Grid Scheduler (GS) or Grid Resource Broker (GRB) receives applications from grid users, selects feasible resources for those applications according to acquired information from the Grid Information Server and finally generates application-to-resource mappings. Grid Schedulers usually cannot control grid resources directly and are not necessarily located in the same domain with the resources, which are visible to them. Figure 1 shows only one Grid Scheduler but in reality multiple such schedulers might be deployed.

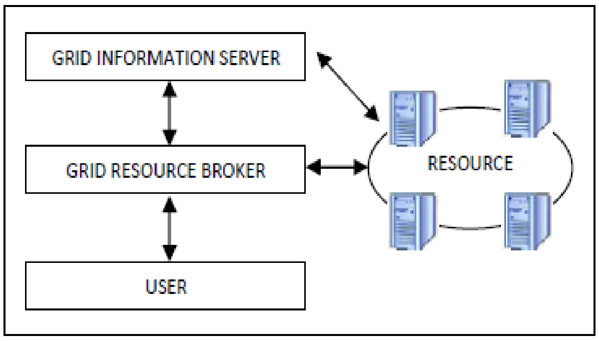

Fig 1: General Grid Scheduler Architecture

B. Grid Scheduling Process

Grid scheduling involves three main phases: *resource discovery*, which generates a list of potential resources; *information gathering* about those resources and selection of a best set; and *job execution*, which includes file staging and cleanup [10]. These phases and the steps that make them up are shown in Figure (2).

Phase One-Resource Discovery	**Phase Three-Job Execution**
1. Authorization Filtering	6.Advance Reservation
2.Application Definition	7.Job Submission
3.Min Requirement Filtering	8.Preparation Tasks
Phase Two-System Selection	9.Monitoring Progress
4. Information gathering	10.Job Completion
5. System selection	11.Clean-up Tasks

Fig 2: Three-phase plan for Grid scheduling.

III. CHALLENGES IN GRID SCHEDULING

Although Grids fall into the category of distributed parallel computing environments, they have a lot of unique characteristics, which make the Scheduling in Grids highly difficult [6]. An adequate Grid scheduling system should overcome these challenges to leverage the promising potential of Grid systems, providing High Performance services.

- ➢ Resource Heterogeneity
- ➢ Site Autonomy
- ➢ Local Priority
- ➢ Resource Non-Dedication
- ➢ Application Diversity
- ➢ Dynamic Behavior

Besides the characteristics of Grid Environment which make the performance evaluation of scheduling algorithms tedious; the following factors also make the task difficult:

- ➢ Performance prediction is difficult because end to end internet performance itself is extremely hard to analyze and predict.
- ➢ End to end performance observed on internet exhibits great diversity and thus different algorithms work more effectively for different topologies and also for different time periods on same topology.

IV. GRID SCHEDULING INFRASTRUCTURES AND APPROACHES

A. Types of Job Scheduling Infrastructures

1. Centralized: Single job scheduler on one instance, all information collected here.

Centralized Job Scheduling – Types

Multi Site Scheduling: A job can be executed on more than one machine in parallel. As job-parts are running on different machines, latency is important, different job-parts are started synchronously on all machines.

Single Site Scheduling: A job is executed on a single parallel machine. This means that system boundaries are not crossed. Efficient as communication inside a machine is fast.

Centralized Job Scheduling -- Advantages

- ➢ Efficiency: The scheduler is conceptually able to produce very efficient schedules, because the central instance has all necessary information on the available resources
- ➢ Centralization is useful e.g. at a computing center, where all resources are used under the same objective.
- ➢ Due to this fact even communication bottlenecks can be ignored.

2. Hierarchical: Two job schedulers, one at global and other at local level.

Hierarchical Job Scheduling -- Advantages

- ➢ Usage of a central scheduler to which jobs are submitted. In addition, every machine uses a local job scheduler
- ➢ Basically centralized as there is one global instance
- ➢ Main advantage: different policies can be used for global and local scheduling
- ➢ Meta-scheduler: redirects submitted jobs to local schedulers for resources based on some policy.

3. Decentralized: No central instance, distributed schedulers interact and commit resources.

Decentralized Scheduling

- ➢ No central instance is responsible.
- ➢ Distributed schedulers interact with each other and decide the allocations for each job to be performed.
- ➢ Information about state of all systems is not collected at a single point.
- ➢ Local job schedulers may have different but compatible scheduling policies.

Advantages of Decentralized Scheduling

- ➢ No communication bottleneck.
- ➢ Scalable to greater extent.
- ➢ Failure of single component doesn't affect whole metasystem.
- ➢ Better fault tolerance and reliability than centralized systems which have no back-ups.
- ➢ Site-autonomy for scheduling can be achieved easily as the local schedulers can be specialized on the needs of the resource provider or the resource itself.

Disadvantages of Decentralized Scheduling

- ➢ The lack of a global scheduler, which knows all job and system information at every time instant, usually leads to sub-optimal schedules.
- ➢ The support for multi-site applications is rather difficult to achieve.
- ➢ As all parts of a parallel program must be active at the same time, the different schedulers must synchronize the jobs and guarantee simultaneous execution.

B. Grid Scheduling Approaches

1. *Static versus Dynamic*

In static scheduling, information regarding of all resources in the grid and all tasks in an application is assumed to be available by the time the application is scheduled. In dynamic scheduling, the basic idea is to perform task allocation on the fly as the application executes. This approach is useful when the execution time determination is impossible.

2. *Approximate versus Heuristic*

An approximate algorithm uses formal computational models. Instead of searching the entire solution space for an optimal solution, they are satisfied when the sufficient good solution is found. This technique can be used to decrease the time taken to find an acceptable schedule where metric is available for evaluating a solution.

3. *Distributed versus Centralized Scheduler*

In dynamic scheduling scenarios, the responsibility for making global scheduling decisions may lie with one centralized scheduler, or in shared by multiple distributed schedulers. The centralized strategy has the advantage of ease of implementation, but suffers from lack of scalability, fault tolerance and the possibility of becoming a performance bottleneck.

4. *Cooperative versus Non-Cooperative*

If the distributed scheduling algorithm is adopted, we have to see that the nodes are working cooperatively or independently. In the case of non-cooperativeness, individual schedulers act as alone autonomous entities and arrive at decisions regarding their own optimum objects independent of the effects of the decisions on the rest of the system.

5. *Adaptive Scheduling*

An adaptive scheduling to the scheduling problem is one in which the algorithms and parameters used to make scheduling decisions change dynamically according to the previous, current and future resource status.

6. *Heuristic Approaches*

Dealing with the many constraints and optimization criteria in a dynamic environment is very complex and hard to compute. Meta-heuristic approaches are considered undoubtedly the *de facto* approach.

a. Local Search Based Heuristic approach:

It is a family of methods that explore the solution space by jumping from one solution to another one and constructing a path in solution space with the aim of finding the best solution for the problem. Methods in this family range from simple ones such as Hill Climbing, Simulated Annealing to more sophisticated ones such as Tabu Search method.

b. Population Based Heuristic approach:

It is a large family of methods that has shown their efficiency for solving combinatorial optimization problems. It usually requires large running times if sub-optimal or optimal solutions are to be found. We could distinguish three different categories of population-based methods: Evolutionary Algorithms (Genetic Algorithms (GAs), Memetic Algorithms (MAs) and their variations), Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO).

V. HEURISTIC SCHEDULING ALGORITHMS

The resource scheduling in grid is a NP complete problem. Various algorithms have been designed to schedule the jobs in computational grid [17]. The most commonly used heuristic algorithms are Genetic Algorithms (GA), Simulated Annealing Algorithm (SA), Ant Colony Optimization Algorithm (ACO) and Particle Swarm Optimization Algorithm (PSO). The research on them received good results and their efficiency had been proved. There are also many related improved work under study. GA to select an optimal or suboptimal scheduling of the tasks is used in [4], [7], [18]. ACO in grid task scheduling is used in [9], [14], [15] and got good results. PSO algorithm is able to get good schedule for grid computing is shown in [1], [19]. SA heuristic to the scheduling of parallel applications is used in [2], [12], [16]. Hybridized algorithm to improve the performance is also proposed in [13], [20].

A. Genetic Algorithm

The Genetic Algorithm **(**GA**)** is one of the best methods to search the large solution space. This method operates on the population of chromosomes for a given problem. The general procedure of GA search is as follows:

Population generation: A population is a set of chromosomes and each represents a possible solution, which is a mapping sequence between tasks and machines.

Chromosome evaluation: Each chromosome is associated with a fitness value, which is the makespan of the task-machine mapping this chromosome represents.

Crossover and Mutation operation: Crossover operation selects a random pair of chromosomes and chooses a random point in the first chromosome. Crossover exchanges machine assignments between corresponding tasks.

Mutation randomly selects a chromosome, then randomly selects a task and randomly reassigns it to a new machine. Finally, the chromosomes from this modified population are evaluated again. This completes one iteration of the GA. The GA stops when a predefined number of evolutions is reached.

1. Advantages

- This is an evolutionary technique for large space search.
- It is a search technique used in computing to find exact or approximate solutions to optimization and search problems.
- This algorithm is also known as evolutionary computation Algorithm.
- It used techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

2. Disadvantages

- In a genetic scheduling algorithm, one open problem is how to use dynamically predicted performance.
- Information to help the speed of convergence (as the current genetic scheduling algorithms are static).

B. Ant colony optimization algorithm

In the ACO algorithm, artificial ants simulate the foraging behavior of real ants. The artificial ants deposit pheromones that guide other ants between food and their nest. Initially, no pheromone trail exists between food and the nest. The ant pheromone trail forms iteratively. At this point, the probability that ants will turn left or right is equal. Over time, ants on a short route leave more pheromone than ants on a long route. This pheromone trail acts as positive feedback attracting other ants to the short route. This natural principle has been applied to solve optimization problems. The ant system simulates the search approach used by ants and then formulates and systematizes the scheme to solve optimization problems.

ACO, a heuristic algorithm is used for rapidly solving the JSP. The ACO was first proposed by Dorigo. Dorigo then extended the ant system and generated ACO [5], [11].

1. The ACO algorithm

Artificial ants are created using a program that simulates the natural behavior of ants. These artificial ants solve problems using three principal characteristics: (1) ants prefer routes with the most pheromone; (2) pheromone accumulates faster on short routes than on long routes; and (3) ants communicate indirectly through the pheromone. Moreover, artificial ants identify superior routes they pass (exploitation) and unsearched combinations of routes (exploration) to obtain enhanced solutions.

2. The ACO procedure

The search proceduress of the ACO algorithm for the JSP can be divided into four steps.

Step 1 Initialize ACO parameters.

The ACO parameters are as follows: N is the number of ants; τ (r,u) is the pheromone trail of edge (r, u); ρ is the proportion of local pheromone evaporated; α is the proportion of global pheromone evaporated; β is the parameter determining the relative importance of heuristic information; η(r,u) is the greedy heuristic; and q0 is a comparative proportion ratio of controlled exploitation and exploration. When an ant k in node r selects the next node s in the ACO algorithm, the following transition rule is followed:

$$s = \begin{cases} \arg\max\limits_{u \in J_k(r)} \left\{ [\tau(r,u)] [\eta Ir,u)]^{\beta} \right\} & \text{when } q \le q_o \ \text{(exploitation)} \\ S & \text{others (exploration)} \end{cases} \qquad (1)$$

where q is a random variable between 0 and 1, q0 is a parameter from the uniform distribution between 0 and 1, and S is a random variable derived by Eq. 2. When an ant selects the node after node r, random variable q is selected. When q≤q0, then the best node is selected for moving from $J_k(r)$, thereby creating a significantly improved solution using Eq. 1; otherwise, a node is selected with a random probability distribution using Eq. 2 (explore new solutions).

$$P_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)]^{\alpha} [\eta Ir,s)]^{\beta}}{\sum\limits_{u=J_k(r)} [\tau(r,s)]^{\alpha} [\eta Ir,s)]^{\beta}} & \text{when } s \in J_k(r) \\ 0 & \text{others} \end{cases} \qquad (2)$$

Step 2 Local update of the pheromone trail.

The transition rule for choosing the next node in a search until all nodes are chosen is called a completion solution. When an ant establishes a complete solution, the pheromone on the route must be updated via a local updating rule. Local updating rules are designed to follow ants when they identify a different route and to calculate earliness and tardiness. The local updating rule is demonstrated by Eq. 3:

$$\tau(r,s) = (1 - \rho) . \tau(r,s) + \rho \ . \Delta\tau(r,s) \qquad (3)$$

where ρ is a local evaporating parameter with a value of 0–1. Furthermore, Δτ (r,s)= τ0 and τ0 represents the initial quantity of pheromone. When N ants generate a complete solution during an iteration, the best solution is selected via a local search and replaces the original solution. This procedure helps locate ideal solutions other ants can learn.

Step 3 Globally update the pheromone trail.

Following the local search, the best solution chosen from the routes of N ants is treated with the global updating rule to update the pheromone trail. The global updating rule is Eq. 4:

$$\tau(r,s) = (1 - \alpha) . \tau(r,s) + \alpha \ . \Delta\tau(r,s) \qquad (4)$$

where

$$\Delta\tau(r,s) = \begin{cases} (L_{gb})^{-1} & \text{if } (r,s) \in \text{best route presently} \\ 0 & \text{others} \end{cases}$$

In Eq. 4, α is a global parameter for pheromone evaporation and is in the range of 0–1. Moreover, L_{gb} is the best solution to date. Therefore, only the best route is determined by the amount of pheromone in an iteration. When an iteration does not reach the previously set value, then searching continues; otherwise, searching ends.

Step 4 Stop and generate a best sequence.

The ants usually build a solution using both the information stored in the pheromone trail and the heuristic function. The ant solution building technique is an attempt to follow the concept of the best heuristic method. Each ant starts with an empty schedule and the processor pji best which will complete each unscheduled job j1,..,jn earliest is established. A job j is then probabilistically chosen to schedule next based on the pheromone value between j and its best processor and heuristic value. The chosen job is then allocated to the best selected ant of each iteration. This process is repeated until all jobs have been scheduled and a complete solution has been built.

3. Advantages

- Inherent parallelism.
- Positive Feedback accounts for rapid discovery of good solutions.
- Suitable for real world problems.
- Low cost for solution.
- It is tolerant of imprecision.
- Conceptual Simplicity.
- Hybridization with other methods.
- Robust to dynamic changes.
- Solves problems that have no solutions.
- Less computation time.
- Efficient for Traveling Salesman Problem and similar problems.

4. Disadvantages

- Theoretical analysis is difficult.
- Sequences of random decisions (not independent).
- Probability distribution changes by iteration.
- Research is experimental rather than theoretical.
- Time to convergence uncertain (but convergence is guaranteed!).
- Tradeoffs in evaluating convergence.
- Coding is somewhat complicated not straightforward.

5. Applications of ACO

- Routing in telecommunication networks
- Traveling Salesman
- Graph Coloring
- Scheduling
- Vehicle Routing Problem
- Assignment problem
- Set Problem
- Data Mining

- Image Processing
- Protein Folding

C. Simulated Annealing

The Simulated Annealing (SA) is a generalization of optimization method for examines the equations of frozen states of n-body systems.

1. Structure of the SA algorithm

```
1. The problem Specific Decisions
        Step 1. Formulation the problem parameters;
        Step 2. Determination of the initial schedule, generate a feasible solution V;
2. The Problem Generic Decisions
        Step 3. Initialization the cooling parameters:
        • Set the initial value of the temperature parameter, T=8;
        • Set the temperature length L=3;
        • Set the cooling rate F=0.9;
        • Set the number of iterations K=0;
3. The Generation Mechanism, Selecting and Acceptance Strategy of Generated Neighbors
        Step 4.
        • Select a neighbor V’ of V where V’∈I(V)
        • Let C(V’)=the cost of the schedule V’
        • Compute the move value Δ=C(V’)-C(V)
        Step 5.
        • IF Δ≤0 accept V’ as a new solution and set V=V’
        • ELSE
        • IF $e^{\Delta/T} > \theta$ set V=V’
        • Where θ is a uniform random number $0<\theta<1$
        • OTHERWISE retain the current solution V
4. Updating the Temperature
        Step 6. Updating the annealing scheduling parameters using the cooling schedule
        $T_{k+1}=F*T_k$   k=1, 2,…
5. Termination of the Solution
        Step 7. IF the stopping criteria is met THEN
        Step 8.
        • Show the output
        • Declare the best solution
        • OTHERWISE GO to step 4.
```

2. Advantages

- Simulated annealing is proved to converge to the optimal solution of the problem;
- An easy implementation of the algorithm.

3. Disadvantages

- As the system temperature cools, it is more difficult for poorer solutions to be accepted. When the temperature is lower, there is less possibility to find a better solution starting from another poorer one. After each mutation, the system temperature is reduced to 90% of its current value.

D. Particle Swarm Optimization

Particle swarm optimization (PSO) is an algorithm modeled on swarm intelligence, that finds a solution to an optimization problem in a search space, or model and predict social behavior in the presence of objectives.

In PSO, Particles change their position by flies in a problem space looking for the optimal position. The status of a particle can be described by its position and velocity. For each particle, the dimension is equal to the size of ready tasks, and the position at each element is corresponded to a resource. At each step, particle modifies its velocity using (4) and updates their positions using (5).

$$v_{ij}(t+1) = \omega_{ij}(t) + c_1 r_1(p_{ij}(t)-x_{ij}(t)) + c_2 r_2 (p_{gj}(t)-x_{ij}(t)) \qquad (4)$$

$$x_{ij}(t+1)= x_{ij}(t)+ v_{ij}(t+1) \qquad (5)$$

1. Advantages

- Algorithmic simplicity.
- Large numbers of members that make up the particle swarm make the technique impressively resilient to the problem of local minima.

2. Disadvantages

- The swarm may prematurely converge. This point is not guaranteed for local optimum.
- The fast rate of information flow between particles, resulting in the creation of similar particles with a loss in diversity that increases the possibility of being trapped in local optima.
- Problem dependent which results in high performance variances.

VI. CONCLUSIONS

New challenges in Grid environments make it an interesting topic, and many research projects are underway. Through our experimental investigations on current scheduling algorithms working in the Grid computing scenario, it is identified that heterogeneity; dynamism, computation and data separation are the primary challenges concerned by current research. It has been convincingly proved in the recent research papers and through this article that job scheduling on computational grids is best solved by heuristic approach. The article tried to cover the state-of-the-art studies of such heuristics namely GA, PSO, SA and ACO to grid systems.

REFERENCES

[1] Ajith Abraham, Hongbo Liu, Weishi Zhang and Tae-Gyu Chang, "Scheduling jobs on computational grids using fuzzy Particle Swarm Alogrithm", *KES2006 10th International Conference on Knowledge-Based & Intelligent Information &Engineering Systems*, Bournemouth International Conference Centre, pp.500-507, October 2006.

[2] Bouleimen.K and Lecocq.H, “A new efficient Simulated Annealing Algorithm for the resource-constrained project scheduling problem and its multiple modes version”, *European Journal of Operational Research*, vol.149, pp. 268-281, 2003.

[3] Buyya.R, Abramson.D, and Venugopal.S, “The Grid Economy”, *Proceedings of the IEEE*, pp. 698-714, 2005.

[4] Di Martino.V and Mililotti.M, “Scheduling in a grid computing environment using Genetic Algorithms”, Proceedings *of the International Parallel and Distributed Processing Symposium: IPDPS 2002 Workshops*, pp.3816-3822, 2002.

[5] Dorigo.M, Maniezzo.V and Colorni.C, “Ant System: An Autocatalytic Optimizing Process”, *Technical Report*, pp. 91-106, 1991.

[6] 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, January 2006.
[7] Fatos Xhafa, Bernat Duran, Ajith Abraham and Keshav P. Dahal, "Tuningstruggle strategy in Genetic Algorithms for scheduling in computational grids" *7th International Conference on Computer Information Systems and Industrial Management Applications*, Ostrava, The Czech Republic. pp. 275-280, June 26 - June 28, 2008.
[8] Foster and Kesselman.C, "The Grid 2: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, USA, 2003.
[9] Hui Fan, Zhen Hua, Jin-Jiang Li, Da Yuan, "Solving a shortest path problem by Ant algorithm", *Proceedings of the Third International Conference on Machine Learning and Cybernetics*, Shanghai, pp. 26-29 August 2004.
[10] Jennifer M. Schopf, "Ten actions when grid scheduling - The User as a Grid Scheduler", http://www.mcs.anl.gov/uploads/cels/papers/P1076.pdf
[11] Marco Dorigo, Vittorio Maniezzo and Alberto Colorni, "The Ant System:Optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics–Part B*, Vol.26, No.1, pp.1-13, 1996.
[12] R.F. de Mello and L.J. Senger, "On Simulated Annealing for the scheduling of parallel applications", *Computer Architecture and High Performance Computing, 2008. SBAC-PAD '08*. 20th International Symposium, pp.29-36, Oct. 29 2008-Nov. 1 2008.
[13] SHU Wanneng and ZHENG Shijue, "A parallel Genetic Simulated Annealing hybrid algorithm for task scheduling", Wuhan University *Journal of Natural Sciences (ENGLISH EDITION)*, vol. 11, pp.1378-1382, 2006.
[14] Siriluck Lorpunmanae , Mohd Noor Sap, Abdul Hanan Abdullah and ChaiChompoo inwai, "An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment", *IEEE International Journal paper*, 2007.
[15] Stefka Fidanova and Mariya K. Durchova: "Ant Algorithm for grid scheduling problem", pp. 405-412, LSSC 2005.
[16] Stefka Fidanova: "Simulated Annealing for grid scheduling problem". *John Vincent Atanas off Symposium* pp.41-45, 2006.
[17] Thilagavathi.J and Antony Selvadoss Thanamani, "A Survey on Dynamic Job Scheduling in Grid Environment Based on Heuristic Algorithm ", *International Journal of Computer Trends and Technology*, vol.3, Issue 4, 2012.
[18] Yang Gao, Hongqiang Rong, Frank Tong, Zongwei Luo and Joshua Huang, "Adaptive job scheduling for a service grid with Genetic Algorithm". *Grid and Cooperative Computing: Second International Workshop, GCC 2003*, Shanghai, China, Part II. LNCS 3033, pp.65-72, December 7-10, 2003.
[19] Zhang. L, Chen.Y, Sun.R, Jing.S, and Yang.B, "A task scheduling algorithm based on PSO for grid computing". International Journal of Computational Intelligence Research, vol.4, 2008.
[20] ZHANG Yan-mei, CAO, Huai-hu and YU Zhen-wei, "A hybrid task scheduling algorithm in grid", Journal *Of Donghua University (English Edition*). vol. 23, pp.84-92, 2006.

Mrs. D. Thilagavathi received her MCA degree from Bharathidasan University in 2001 and completed her M.Phil. degree in Computer Science from Bharathiar University in 2005. She is currently pursuing her Ph.D. at the Research Department of Computer Science, NGM College, Pollachi, under Bharathiar University, Coimbatore. Her research interests include Object Oriented Analysis and Design and Grid Computing. She has 12 years of teaching experience. She is presently working as an Assistant Professor and Head, Department of Computer Technology, NGM College, Pollachi.

Dr. Antony Selvadoss Thanamani is presently working as Professor and Head, Dept of Computer Science, NGM College, Coimbatore, India (affiliated to Bharathiar University, Coimbatore). He has published more than 100 papers in international/national journals and conferences. He has authored many books on recent trends in Information Technology. His areas of interest include E-Learning, Knowledge Management, Data Mining, Networking, Parallel and Distributed Computing. He has to his credit 24 years of teaching and research experience. He is a senior member of International Association of Computer Science and Information Technology, Singapore and Active member of Computer Science Society of India, Computer Science Teachers Association, New York.