Asian Journal of Information Technology 6 (8): 859-865, 2007
ISSN: 1682-3915
© Medwell Journals, 2007

Scheduling of Scientific Workflows Using Evolutionary and Threshold Accepting Algorithm for Grids

S. Benedict and V. Vasudevan
Software Technologies Group, TIFAC Core in Network Engineering,
Anna University, Nigeria

Abstract: Grid computing environment involves all kind of resources namely network, software, data, storage and processing units, evolving towards Global computing to solve a single large problem using Grid scheduling architecture that addresses the interaction between the resource management and data management. In this study, two different approaches have been proposed to solve Grid scheduling problem with the objectives of maximizing the Job Completion Ratio (JCR) and minimizing the lateness. A population based evolutionary algorithm that involves evolution during the search process and a single point local search meta-heuristics that work on a single solution called as hybrid evolutionary algorithm. A Threshold Accepting algorithm (TA) proposed is a single point local search meta-heuristic. Proposed algorithms are evaluated and the experimental results are presented for comparison.

Key words: Grid computing, genetic algorithm, scheduling, threshold accepting

INTRODUCTION

An important concept toward a computational infrastructure for resource sharing in the Internet is Grid computing. Grids (Berman *et al*., 2002) are considered as a type of parallel and distributed system that enables the sharing, selection and aggregation of geographically distributed autonomous resources depending on their availability, capability, cost and user QoS requirements to solve large scale problems in science and engineering (Cannataro *et al*., 2002; Frey *et al*., 2002). The large scale problems are very often, subdivided into smaller tasks as in Nile scheduler (Alessandro and Kaith, 2006; Amoroso *et al*., 1998) and dispersed to the Grid sites. The Grid sites accept tasks submitted from intra and inter sites. The availability of resources in Grid sites is still limited, although the computers have shown the growth in all extends, to run applications such as the mass spectrometric analysis in 3D rendering computations, molecular modeling and Brain activity analysis and so on. Many applications such as high energy physics experiments currently being developed by European Nuclear Research Centre (CERN) (Chien *et al*., 2005; Frey *et al*., 2002) and next generation experiments (Rajkumar, 2005) are creating a flood of data (Cannataro *et al*., 2002) to compute and thus considers maximization of job completion and minimization of lateness as an objective. The grid users anticipate the completion of workflows submitted to the system within the expected deadline.

Grid in reality becomes a major problem owing to the challenges in security issues (Song *et al*., 2006), protocol design, access control (Baker *et al*., 2002; Song *et al*., 2005), implementation models (Parashar and Browne, 2005), resource management and workflow management.

Taxonomy of workflow management systems for Grid computing explains the elements of grid workflow management system as workflow design, Information retrieval, workflow scheduling, fault tolerance and data movement.

Following are the grid workflow scheduling challenges (Alain *et al*., 2004):

- Unpredictable challenges in Grid resources (i.e., availability, accessibility and so on).
- Need for a multiple resource types for completing a job.
- Need for a parallel or concurrent execution of tasks in any workflows.

Advanced reservations for scheduling the workflows via a central scheduler were the traditional system. The overloading and the scheduler failure problem raised are

Corresponding Author: S. Benedict, Software Technologies Group, TIFAC Core in Network Engineering, Anna University, Nigeria

overridden by a two level scheduling scheme where the first level is used for frequent small jobs and second level for large jobs is often used. The market oriented approach (Chien *et al*., 2005) algorithm succeeded in the distributed scheduling of workflows, but could not appease completion of more workflows within the deadline.

On considering the large periodical jobs that are scheduled in advance, evolutionary techniques, namely, Genetic algorithm provide optimal results. This study addresses the problem of sequencing, scheduling of grid sites with n workflows in a grid computing environment with the simultaneous objectives of maximizing Job Completion Ratio (JCR) and minimizing lateness. An evolutionary approach and a Threshold accepting algorithm are proposed in this study of grid scheduling problem. Proposed algorithms are evaluated with other algorithms such as First Come First Serve (FCFS) and Earliest Deadline First (EDF).

Related work: Recent years have seen many efforts focused on the efficient utilization of Grid resources by Gridscheduling to appease the customer and provider satisfactions, such as Condor (Frey *et al*., 2002), Gridbus, Gridport, OptimalGrid (Kaufman *et al*., 2004), ComCiDis and Silver. Gridbus considers incentive as a design parameter for Grid computing. Few projects aggregates the computing power of networked machines into computational grids namely Alchemi, Grid way and Astrogrid for the way to access, process and store astronomical data obtained from world.

In Mausolf (2004), community scheduler is developed for implementing a grid meta-scheduler that works in heterogeneous environment with the latest version of Open Grid Services Infrastructure providing FCFS, Round Robin and throttle scheduling policy. A key to high throughput is the efficient use of distributed resources. The large amount of computational power is kept idle (Aloisio *et al*., 2002).

Two-level scheduling for large periodical jobs is discussed using Genetic algorithm in (Vincenzo and Marco, 2002). Here an optimal solution is selected using the parameters such as job characteristics, environment characteristics and data distribution characteristics. Six risk-resilient heuristics for security assured grid job scheduling is discussed in Song *et al*. (2006) using space time genetic algorithm. The NAS workload trace discussed in Song *et al*. (2006) are non-Directed Acyclic Graph (DAG) jobs. Real time Grids have to execute both DAG and non-DAG workflows.

In Abraham *et al*. (2000), the economic models for scheduling using evolutionary techniques and other heuristics namely Tabu Search, simulated annealing and hybrid approach with single objective of scheduling are considered. The works related to Halpern and Moses (1990) Real *et al*. (2003), considers scheduling using learning algorithm based approach, requiring more database for efficient scheduling.

Our approach, using TA and HYBRID based scheduling, combines two level scheduling, DAG and non-DAG workflows and combined simultaneous objective for searching a better solution in a search space.

PROBLEM DESCRIPTION

Figure 1 show the Inter-Grid architecture described with a three node Grid environment example conducted in the campus of TIFAC Core in Network Engineering. This architecture can be utilized for any practical applications for the normal grid environments.

The goal of the Inter-Grid Scheduler architecture is to find the allocation sequence of workflows on each Grid site. Four major entities are involved in this architecture:

- The Grid users submit their request for job completion in the form of workflows to the local grid managers.
- All the tasks should be received by the Grid managers and the decision for the scheduling is made on deploying the request to the Intra-Grid schedulers.
- The Intra-Grid schedulers have the updated information of the grid resources that are idle during time‘t’. This information is frequently updated. The smaller jobs can be scheduled within the deadline by the Intra-Grid schedulers where the scheduling is often dynamic.
- For data intensive applications where the jobs are larger require the necessity of the resources world wide. At that moment, there is a necessity of Inter-Grid schedulers which is static often.

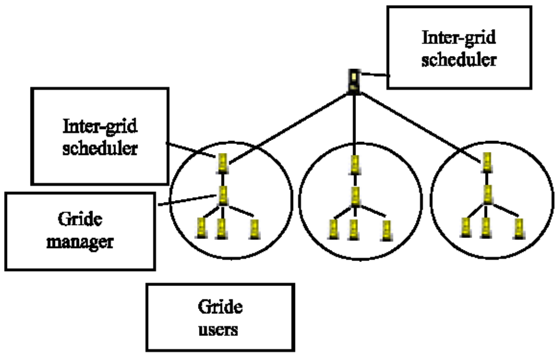

Fig. 1: Inter-Grid scheduler architecture

Table 1: Experimental workflows

SN	Work flow	Tasks/ (duration)	Grid sites
1	W1	$T_1(2),T_2(4),T_3(3),T_4(1),T_5(3),T_6(2),T_7(3)$.	Any
2	W2	$T_1(3),T_2(2),T_3(3),T_4(4),T_5(3),T_6(1)$.	Any
3	W3	$T_1(2),T_2(4),T_3(1),T_4(2),T_5(3),T_6(4),T_7(5)$.	Any
4	W4	$T_1(2),T_2(3),T_3(1),T_4(4),T_5(5),T_6(1),T_7(2), T_8(3)$.	Any
5	W5	$T_1(1),T_2(3),T_3(2),T_4(4),T_5(5),T_6(6)$.	Any
6	W6	$T_1(1),T_2(3),T_3(1),T_4(2),T_5(4),T_6(2)$.	Any
7	W7	$T_1(1),T_2(4),T_3(2),T_4(5),T_5(3),T_6(2)$.	Any
8	W8	$T_1(1),T_2(3),T_3(2),T_4(3),T_5(1),T_6(1),T_7(2), T_8(3)$.	Any
9	W9	$T_1(3),T_2(3),T_3(4),T_4(6),T_5(2),T_6(4),T_7(2), T_8(1)T_9(5)$.	Any
10	W10	$T_1(4),T_2(5),T_3(4),T_4(4),T_5(5),T_6(1),T_7(2)$.	Any

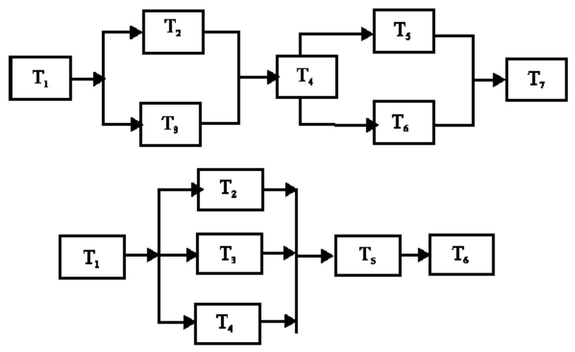

Fig. 2: DAG for W1 and W5

The workflow allocation strategy in a Grid environment differs as in Hamscher *et al.* (2000). The goal of the Inter-Grid scheduler is to receive the request from different intra-grid schedulers and make an optimistic scheduling such that it accommodates many workflows completing within its deadline. The simulator developed is to show the comparative performance of the TA and HYBRID along with the other traditional algorithms.

The setting of the experiment consists of workflows with following assumptions:

- Each workflow received in the Inter-Grid scheduler consists of a set of Tasks T_1, T_2, T_3 and so on.
- The task in each workflow is a DAG model. The output from a task can be transferred to other tasks and its transmission time is negligible.
- At any time, a task can be executed only on a Grid site which is reported to the Inter-Grid scheduler as idle via Intra-Grid scheduler (Wieczoek *et al.*, 2005). There is no pre-emption of tasks or workflows. The sequential order of workflow allotment changes. For experimental purpose the following DAG workflows as given in Table 1 are considered.

Here, we present a scheduling approach for the wide area problem. The resources and jobs are dispersed geographically on a wide area network. The scheduler provides a cost-effective, scalable WAN solution to connect and integrate grid resources. The DAG workflow model for W1, W5 are shown in Fig. 2.

The duration for each task in any workflow and the required grid resources are given in the Table 1. The tasks taken for experiment have its predecessors and successors, such as T_1 follows T_2 or T_2, T_3 are parallel computation once when the task T1 is executed.

THE HYBRID APPROACH

The population based algorithm uses several individuals to search the solution space of a problem. HYBRID algorithm (HYBRID) proposed in this study is similar to Genetic Algorithm (GA) and uses the genetic operators such as crossover and mutation to explore the solution space. However, in addition to genetic operations it uses a local search phase to improve genotypes. Thus, the local search and population-based evolutionary method are combined together (Table 2).

Hybrid procedure

Chromosome structure: Workflow numbers are taken as genes. The sequence of the workflows is used to represent a chromosome. Population size is taken equivalent to the number of workflows. Initial population is generated randomly.

Crossover: It is essential to have better crossover operator, termed as crossover probability, suitable for the given coding scheme to generate feasible offspring to pass good qualities from parent to offspring. Example: (Cross over site = 5th position)

Pair of strings before crossover
W6, W2, W7, W5, W1, W4, W3, W8, W10, W9.
W7, W1, W4, W10, W9, W2, W5, W6, W3, W8
Pair of strings after crossover
W6, W2, W7, W5, W1, W4, W10, W9, W3, W8.
W7, W1, W4, W10, W9, W6, W2, W5, W3, W8.

Mutation: Mutation is an operator to exchange genes in a chromosome that is generated by a crossover operator a transition from a current solution to its neighborhood solution is possible by mutation. For example, if the generated random numbers are 2 and 8, then the corresponding part numbers in these positions are interchanged and the new sequence is obtained,

String before mutation
W6, W2, W7, W5, W1, W4, W10, W9, W3, W8
String after mutation
W6, W9, W7, W5, W1, W4, W10, W2, W3, W8

Table 2: HYBRID parameters

Parameters	Selection
Population size	n
Probability of crossover	1
Probability of mutation	1
Number of local search times	n

Local search: After mutation, local search is applied. Insertion perturbation scheme is used as a local search technique. After the local search, improved sequences are added to the population. All members of the population undergoes local search. After evaluating a certain number of solutions, the program execution is terminated. The same criterion is applied for all the algorithms considered in this study.

THRESHOLD ACCEPTING ALGORITHM

Threshold accepting algorithm is a simple procedure and it posses the ability to generate high quality solutions. The effort focuses on developing an innovative method which produces quality solutions in a reasonable amount of time. Applications of TA in scheduling include (Gunder and Scheuer, 1990; Gunder and Wirsching, 1989) among several others.

TA algorithm: The algorithm begins with an initial solution *S* and an initial threshold value T_1. Neighborhood structure is defined by the perturbation mechanism employed. A neighborhood solution S^* to the current solution is generated by using the perturbation scheme. The current and the candidate solution are evaluated and the objective function value is obtained.

If the candidate solution is acceptable, it becomes the current solution and this completes an iteration of the TA procedure. After the completion of individual iterations, the threshold value is reduced by r known as threshold reduction step size and the iteration is repeated. The algorithm is terminated when a final threshold T_2 is reached. The Threshold Accepting (TA) method is very similar to Simulated Annealing (SA) method.

The essential difference between SA and TA is that different acceptance rules are used. TA accepts every new move, which is not much worse than the old one. SA accepts worse solutions only with rather small probabilities. An apparent advantage of TA is its greater simplicity. It is not necessary to compute probabilities or make random decisions. A brief description of the TA procedure is given as shown in Fig. 3.

TA operators: The best solution obtained in TA depends on the TA operators such as initial threshold, final threshold, threshold reduction step size and the number of iterations to be performed at a particular threshold that

```
/* Structure for TA algorithm */
Step 1. Get an initial solution S
Step 2. Assign an initial threshold T1 and threshold
     reduction step size r
Step 3. Set the number of moves to be performed at
          each threshold T
Step 4. Perform the following steps neighbor_move      times.
      4.1 Select a neighbor S* from the neighbor
            space of S and compute f (S*)
      4.2 If S* is acceptable, set S=S*.
      4.3 Reduce the threshold by setting T=T-r
                                                          If (T>0)
                                                          Goto
step 4.1
Step 5. Return the best solution found.
```

Fig. 3: Structure for TA algorithm

Table 3: TA parameters

Parameters	Selection
Initial threshold	0.099
Final threshold	0
Reduction step size	0.001
Number of iteration particular threshold	n

overrides simulated annealing optimization techniques. The initial threshold and threshold reduction step size are fixed such that a reasonable number of iterations can be carried before the algorithm freezes. In the present study, an initial threshold of 0.099 is chosen and threshold reduction step size is fixed as 0.001 (Table 3).

Perturbation schemes: The three permutation techniques used in this study are as follows:

- Pair wise exchanges
- Insertion method
- Random Insertion Permutation Scheme (RIPS)

Pair wise exchanges: The proposed perturbation scheme namely the pair wise exchange can be best explained with an example. Consider the workflow sequence (1,2,3,4,5) as a seed sequence and that the integers i,j (i,j<=W_n) are randomly generated. Suppose in the first instance i = 1 and j = 4, the pair wise exchange technique will generate the new sequence (4,2,3,1,5); which is the result of exchanging the first and fourth integers in the starting sequence.

Insertion techniques: Consider the sequence 1,2,3,4,5 as a seed sequence and that the i,j (i,j<= W_n) are randomly generated. Suppose in the first instance i=1 and j=4. The insertion technique will generate a new sequence 2,3,4,1,5. i.e. the result of inserting the first integer in the sequence in the fourth position. In this case, the same sequence is generated, the movement of the integers (workflows) is forwards or backwards as i<j or i>j.

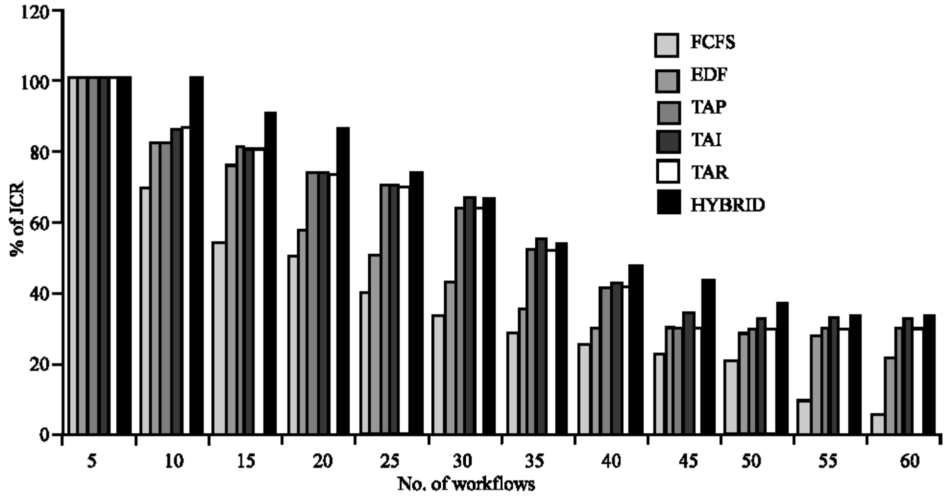

Fig. 4: Job completion ratio

Random Insertion Perturbation Scheme (RIPS): Consider a seed sequence S={1,2,3,4,5}. The digit in the first position can be inserted at any position to its right. Hence, the digit in the first position is inserted at any position between 2 and W_n (for example W_n =5) and a random number generation between 2 and W_n is used to select the job position. Suppose the selected position is 3, first digit is inserted in position 3, yielding a new sequence, {2,3,1,4,5}. Consider the digit in the second position of sequence S and chose randomly two positions for its insertion. Note that this digit can be inserted at any position between (2+1)th to nth position (i.e. a position to its right) and between 1st and (2-1)th position (i.e., a position to its left). Suppose position 1 is selected to the left and position 4 is selected to the right of digit 3, the new sequences thus generated are {2,1,3,4,5} and {1,3,4,2,5}. Similarly, for the digits in positions 3 and 4, we select two positions randomly, one to the right and one to the left and obtain the resulting sequences {1,3,2,4,5}, {1,2,4,3,5}, {4,1,2,3,5} and {1,2,3,5,4}.

Combined Simultaneous Objectives (CSO): The two objectives namely minimizing lateness of executing individual workflows 'F1' and maximizing number of workflows completed with in deadline 'F2' are combined using the weighting factors after every solution move among the neighborhood for TA. The CSO is represented as given by the formula below:

Minimize CSO

$$\mathrm{CSO} = \left(\left(0.5\times \mathrm{F1}\right) + \left(0.5\times\left(1\div \mathrm{F2}\right)\right)\right) \tag{1}$$

The individual objective functions F1 and F2 are given by the formulae 2 and 3.

F1=Minimizing lateness of executing individual workflows:

$$\mathrm{F1} = \sum_{i=1}^{m}(\mathrm{tw}_i - \mathrm{dtw}_i) : \text{Only when } \mathrm{tw}_i \geq \mathrm{dtw}_i \tag{2}$$

F2= Maximizing number of work flows completed with in dead line

$$\begin{aligned} \mathrm{F2} &= \sum_{i=1}^{m}\mathrm{xw}_i \\ \text{where } \mathrm{xw}_i &= 1 \quad \forall \mathrm{tw}_i \leq \mathrm{dtw}_i \\ &= 0 \quad \text{Otherwise} \end{aligned} \tag{3}$$

RESULTS AND DISCUSSION

This optimal schedule for the Inter-Grid computation of workflows is obtained by the procedure using TA. This is compared with sequences obtained by different scheduling rules viz. HYBRID, FCFS and EDF. For the experimental problem the sequence obtained by HYBRID gives minimum lateness for each workflow received in Inter-Grid schedulers and maximize the job completion ratio and hence minimal CSO.

However, TA approach search the solution faster and use the minimum average CPU time. The satisfaction is obtained in getting the results with multiple objectives at a time. The maximum number of workflows completed by different algorithms such as FCFS, EDF, TAP, TAI, TAR and the HYBRID are shown in Fig. 4.

The minimization of CSO for scheduling on increasing the number of workflows happens with HYBRID. The result for the illustration of the minimization of the CSO of the scheduling scheme is shown in Fig. 5.

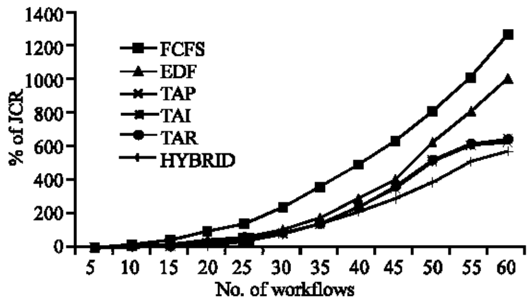

Fig. 5: CSO of the workflows

Table 4: Average CPU running time

Number of workflow	Average CPU time in milli see			
	FCFS	EDF	TA	HYBRID
10	5.5	4.7	118.8	1425.2
20	6.1	6.1	412	4944
30	67	10.8	804	16080
40	89.4	87.5	1109.9	27725
50	144.8	97.7	1521	45630
60	191.8	187.9	1951	68285

As shown, the CSO increases abruptly for the scheduling schemes with FCFS and EDF algorithms compared with HYBRID and TA. TA provide a near to optimal solution much faster which remain more important for the data mission critical applications compared to HYBRID as shown in Table 4.

The result manifest the cruciality of the TA algorithm since it is obvious that the lateness for searching the near to optimal solution are unaffordable. On suitably choosing the reduction step size, the algorithm outperforms for its better execution time within deadline.

CONCLUSION

In this research, simultaneous optimization of dual objective of an Inter-Grid Scheduler is addressed. We have developed a TA algorithm and HYBRID to generate nearer-to-optimum schedules. The procedures are tested with an example problem environment and the results are compared with that of other available scheduling procedures namely FCFS and EDF. It is found that HYBRID is more effective than the other algorithms such as FCFS, EDF and TA in the perspective of CSO and TA in the perspective of required CPU time.

In the near future we combine TA along with the Tabu search method to increase the efficiency. Similarly, the ant colony properties can be included for scalability in the existing algorithm. The procedure can also suitably be modified and applied to any kind of Grid scheduling with different problem environment and optimizing any number of objectives concurrently.

REFERENCES

Abraham, A., R. Buyya and B. Nath, 2000. Nature's Heuristics for Scheduling Jobs on Computational Grid s. The 8th IEEE . Int. Conf. Adv. Comput. Commun. (ADCOM), Cochin, India, pp: 30-35.

Aloisio, G. and D. Taha, 2002. Grid computing: Toward a new computing Infrastructure. Future Generat. Comput. Sys., 18: 32-34.

Amorosa, A. and K. Marzullo, 2006. Multiple Job Scheduling in a connection limited Data parallel system. IEEE. Trans. Parallel Distributed Sys., 17: 125-134.

Amoroso, A., A. Ricciardi and K. Marzullo, 1998. Wide Area Nile: A case study of a wide area Data-Parallel applications, Proc. 18th IEEE Int.Conf. Distributed Computing Sys., IEEE. Comput. Soc. Pub., pp: 506.

Andrieux, A. *et al.*, 2004. Open Issues for Grid scheduling, Available: http://www.nesc.ac.uk/ technicalpapers/ UKeS-2004-03.

Baker, M., R. Buyya and D. Laforenza, 2002. Grids and Grid Technologies for Wide-Area Distributed Computing, Software: Practice and Experience (SPE). J. Wiley Press, pp: 1437-1466.

Berman, F., G. Fox and T. Hey, 2002. Grid Computing: Makiing the Global Infrastructure a Reality. Wiley Pub., pp: 110-115.

Cannataro, M., D. Talia and P. Trunfio, 2002, Distributed data mining on the grid, Future Generation Computer Sys., 18: 1101-1112.

Chien, C., P.H. Chang and V. Soo, 2005. Market Oriented Multiple Resource Scheduling in Grid Computing Environments. Proc. 19th Int. Conf. Adv. Inform. Networking and Applications (AINA) IEEE. Comput. Soc., pp: 867-872.

Dueck, G. and J. Wirsching, 1989. Threshold accepting algorithms for Multi-constraint 0-1 Knapsack problems. IBM Germany, 1-15, TR 89.10.016.

Frey, J., T. Tannenbaum, I. Foster, M. Livny and S. Thecke, 2002. Condor-G: A Computation Management Agent for Multi-Instituitional Grids. J. Cluster Comput., 5: 237-246.

Gunter Dueck and Tobias Scheuer, 1990. Threshold accepting: A General purpose optimization algorithm appearing superior to simulated annealing, J. Computational Physics, pp: 161-175.

Halpern, J.Y. and Y. Moses, 1990. Knowledge and common knowledge in a distributed environment. J. ACM., 37: 549-587.

Hamscher, V., U. Schwiegelshohn, A. Streit and R. Yahyapour, 2000. Evaluation of Job Scheduling strategies for Grid computing, GRID 2000 First IEEE/ ACM International workshop, proceedings. Springer Pub., pp: 191-202.

Jeff Mausolf, 2004. Use Community scheduler framework to implement grid meta-scheduler. IBM, http:// www-128.ibm.com/developerworks/grid/library/gr-meta.html.

Kaufman, J. *et al*., 2003. Optical Grid-Autonomic computing on the grid, IBM, http://www-128.ibm.com/developerworks/library/gr-opgrid/

Parashar, M. and J.C. Browne, 2005. Conceptual and Implementation models for the grids. Proc. IEEE., 93: 653-668.

Rodrigo Real, Adenauer Yamin, Luciano da Silva, Gustavo Frainer, Iara Augustin, Jorge Barbosa and Claudio Geyer, 2003. Resource Scheduling on Grid: Handling Uncertainty Proceedings of the Fourth Int. Workshop on Grid Comput., pp: 205-207.

Shanshan Song, Kai Hwang and Yu-Kwong Kwok, 2005. Trusted Grid computing with security binding and Trust Integration, J. Grid Computing.

Shanshan Song, Kai Hwang and Yu-Kwong Kwok, June 2006. Risk Resilient Heuristics and Genetic algorithm for security assured Grid Job Scheduling, IEEE. Trans. Comput., 5: 703-719.

Vincenzo Di Martino and Marco Mililotti, 2002. Scheduling in a Grid computing environment using Genetic Algorithms, Proceedings of the Parallel and Distributed Processing Symposium, IEEE.

Wieczoek, M., R. Prodan, T. Fahringer, 2005. Scheduling of Scientific Workflows in the ASKALON Grid Enivironment. SIGMOD. Rec., 34: 56-62.