A Hybrid Genetic Algorithm with Elitist Ant System in Grid Scheduling

Radha[1] and Dr. V. Sumathy[2]

[1.] Asst.Professor, Dept. of Computer Applications, Sri Ramakrishna College of Engg, Coimbatore-641 022,
[2.] Prof., Dept. of ECE,Government College of Technology,Coimbatore-641 022, INDIA
radhakbr10@gmail.com

Abstract: Services like resource discovery, monitoring and scheduling are more complicated in a grid environment due to the resource pool being large, dynamic and architecturally diverse. A Grid scheduler ensures resource selection decisions in an environment where it cannot control local resources, as the latter are distributed, and systems information is limited/dated and such interactions are closely linked to Grid Information Services functionality. This paper addresses dynamic scheduling of jobs to distributed computing resources. No single scheduling method is enough as scheduling problems have richness and variety. Makespan is the most common objective function of task scheduling problems. Makespan minimisation ensures jobs to level differences between each phases' completion time. In this paper, a hybrid Genetic Algorithm (GA) with incorporates Ant Colony Optimization (ACO) for grid scheduling is proposed. The proposed Hybrid Genetic Algorithm with Elitist Ant System (HGAEAS) demonstrates its effectiveness for Grid Scheduling.

[Radha and V. Sumathy. **A Hybrid Genetic Algorithm with Elitist Ant System in Grid Scheduling.** *Life Sci J* 2013;10(7s):510-515]. (ISSN: 1097-8135). http://www.lifesciencesite.com. 79

Keywords: Grid Computing, Grid scheduling, Genetic Algorithm (GA), Ant Colony Optimization (ACO), Makespan

1. Introduction

A Grid is a dynamic heterogeneous environment agglomerating geographically distributed resources and is defined as a process of taking scheduling decisions concerning resources spread over various administrative domains [1, 2]. This includes examining varied administrative domains to identify a single machine or multiple resources at single/multiple sites to schedule job. From a grid point of view, a job is anything needing a resource. A Grid scheduler, also called broker, takes decisions about resource selection where it lacks control over local resources, and system information often being limited or dated. These interactions are linked to Grid Information Services [3] functionality.

Figure 1 depicts a grid scheduler's main phases. A grid resource broker is responsible for resource discovery, deciding job allocation to a particular resource, binding user applications (files), hardware resources, initiating computations, adapting to grid resource changes and presenting the grid to the user as a single, unified resource [4]. The broker controls tasks physical allocation managing available resources constantly while dynamically updating grid scheduler when resource availability changes. Knowing processing speeds of available resources and user applications job length is a tedious task in a grid environment. Though it is easy to get information about available resources speed, it is complicated to know users computational processing time requirements. When computing power demand is greater than available resources only dynamic scheduling is useful. To think of the problem as an algorithm, it should dynamically estimate job lengths from user application specifications or historical data.

Scheduler structure depends on the number of resources managed and the domain where they are located. Generally, three models to structure schedulers can be distinguished: Centralized, Decentralized and Hierarchical [5]. The first is used to manage single/multiple resources located either in single/multiple domains and can only support uniform scheduling and is thereby suitable for cluster management (or batch queuing) systems.

The Decentralized model fits a typical Grid environment better as schedulers interact among themselves to decide resource allocation for jobs being executed. As it has no central component responsible for scheduling, this model is highly scalable and fault-tolerant. Resource owners can define their policies that require schedulers to enforce. But as remote jobs and resources status is not available at a single location, it is questionable whether it can generate highly efficient schedules.

The Hierarchical model fits Grid environments as it allows remote resource owners to enforce own policy on external users removing single centralization points. This looks like a hybrid model (combining central and decentralized models) and in all likelihood will suit Grid systems better. The scheduler at the top of the hierarchy is called a super-scheduler/resource broker, to interact with local schedulers to decide schedules.

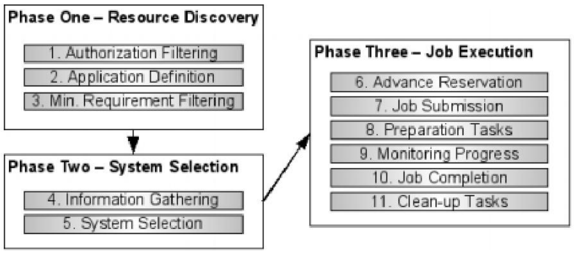

Figure 1: Main Phases of a Grid Scheduler

The operations management has to optimize limited resources use. Techniques combined into heuristic approaches [6] or in upper level multi-objective methodologies (i.e., meta-heuristics) [7], are the lone methods to schedule when there is a high problem dimension and/or complexity. As optimization techniques, metaheuristics are stochastic algorithms trying to solve many hard optimization problems that are effective than traditional methods. In recent years, metaheuristics attracted attention from the hard optimization community as a powerful tool, as it demonstrated promising results from both experiments and practice in engineering areas. So, recent scheduling problem research focused on such techniques [8-10]. This research examines meta-heuristic approaches for scheduling issues as the latter is a NP complete problem.

Design space for Grid Schedulers in usually rich. First, it is based on the objective function a user wants to minimize/maximize - examples being minimizing overall job completion time, minimizing communication time and volume, and maximizing resource utilization. Second, it is also based on how job requirements, job performance models, and Grid resource models are used. The scheduler must choose carefully between different implementations of user authentication, allocation, and reservation. Other choices are scheduling application components for single/multiple users and whether rescheduling/re-planning is needed. The objective in this study is to minimize overall job completion time or the application makespan, the latter often being the performance feature in resource allocation [11]study. Makespan represents lapsed time from the first task's beginning to the end of the last scheduled task. Makespan minimisation arranges tasks to level differences between each work phase's completion time. This paper proposes a hybrid Genetic Algorithm (GA) incorporating Ant Colony Optimization (ACO) for grid scheduling with optimal makespan.

2. Related Works

To obtain an optimal answer in an acceptable time through heuristic search techniques to schedule grid resources, techniques like GA, Tabu search, simulated Annealing [7-10, 12-14] are used. Between them GA provides a better result to solve problems with NP complexity. Literature also proposes many GA based heuristics. Various algorithms attempt to overcome this problem by changes in fitness functions. Some works available in the literature are reviewed in this section.

Pinel et al [15] proposed a new parallel asynchronous cellular genetic algorithm for multi-core processors applicable to scheduling of independent grid tasks. Cellular genetic algorithms (CGAs) are GA type with decentralized populations that outperform regular GAs in many problems with different features and distinct domains. The proposed algorithm was analyzed for parallelism, different recombination and new local search operators. It improved earlier schedules on benchmark problems. The algorithm's parallelism makes it suitable for bigger problem instances.

Delavar et al [16] introduced a GA based scheduling algorithm for independent tasks. In grids with high fault rate, fault tolerant is used from check point method as it has more efficiency than other methods like retry, migration and replication. The proposed method proved effective and efficient in these situations. Servicing quality increases in grid environments while average task recovery time decreases greatly. The proposed algorithm aims to reduce repeating of GA generations to reach higher speeds considering communications costs (available in the fitness function) while maintaining fitness efficiency. Simulations are done with Gridsim to reveal improvement of the proposed algorithm rather than earlier algorithms.

Jin Xu, et al., [17] proposed many versions of Chemical Reaction Optimization (CRO) algorithm to tackle grid scheduling issues. CRO, a population-based meta heuristic was inspired by molecular interactions. This study was due to the fact that inspite of Grid computing solving high performance/high-throughput computing problems through sharing resources,- ranging from personal computers to supercomputers globally - a major issue was task scheduling. The study accounted for resources reliability in addition to makespan and flowtime. Task scheduling was conceptualised as a three objective optimization problem. A meta heuristic approach was chosen to locate an optimal solution as this was a NP hard problem. This paper compared CRO methods with four other meta heuristics on a range of instances. Simulation results proved that CRO outperformed 5 existing methods, this being very high in large-scale applications. The study also proved that vector-based representation was better than permutation-based procedures for independent task grid scheduling issues.

Garg [18] proposed a novel LP driven metascheduling algorithm to map independent applications within a given deadline and budget to rented grid resources aimed at minimizing the combined cost of all users who share a metascheduler. The proposed algorithm combines benefits of LP and GA for a solution to grid scheduling issues with concurrent users and multiple heterogeneous resources. Though only 3 QoS parameters are considered (number of processors, deadline and budget), the proposed algorithm is general enough to handle more parameters. Simulation studies prove its effectiveness compared to other greedy and genetic algorithms.

Guo, et al., [19] introduced Local Node Fault Recovery (LNFR) mechanism into grid systems, to ensure reliability in grid services. The study presented an in-depth look at grid service reliability modeling and analysis with an earlier mentioned fault recovery type. This study came about due to the fact that though there was information on tools and development of tools and grid system techniques, yet important issues like grid service reliability and grid task scheduling were not looked into at depth. Grid services reliability was rather low for some grid services with large subtasks needing large computation. Constraints like subtasks lifetimes and recoveries performed in grid nodes were introduced to ensure that LNFR was practical. Grid service reliability models were introduced under such constraints. Based on proposed grid service reliability model, a multi-objective task scheduling optimization model was also presented followed by development of ant colony optimization (ACO) algorithm for a successful solution. However, experiments showed that along with grid service reliability costs also increased, which was not anticipated. The reason was that resource price was arbitrary, while in practice price was linked to performance of grid resources like CPU processing capability and reliability. So, grid resources price played a major role in grid task scheduling.

The main contribution of this paper is developing a GA based hybrid optimization for grid scheduling. Hybridization among different meta-heuristics was effective for many problems by outperforming single methods [20]. Hybridization expects better convergence than a pure GA search and it can improve GA efficiency.

3. Methodology

Optimization parameter

This study considers scheduling as a single objective optimization problem, where makespan is minimized. Makespan, the finishing time of latest task, is defined as

$$\min_{S} \max \left\{ F_j : j \in jobs \right\}$$

where F_j denotes the finishing time of job j in schedule S. It is advantageous to define a machine's completion time for a given schedule, as this indicates when the machine will finalize processing of jobs assigned earlier as well as those already planned. Completion time values are used to compute the makespan.

A Hybrid Genetic Algorithm with Elitist Ant System (HGAEAS) in Grid Scheduling

To avoid premature GA convergence, due to interference from mutation and genetic drift, sharing and crowding decrease the amount of duplicate schemata in the population. Elitism is incorporated to keep most superior individuals within the population.

Enhancement, crossover, and mutation are the three HGAEAS's design algorithm operators for proposed Grid Scheduling. The proposed system's block diagram is given in Figure 2. A step by step account of an overall learning process is provided below.

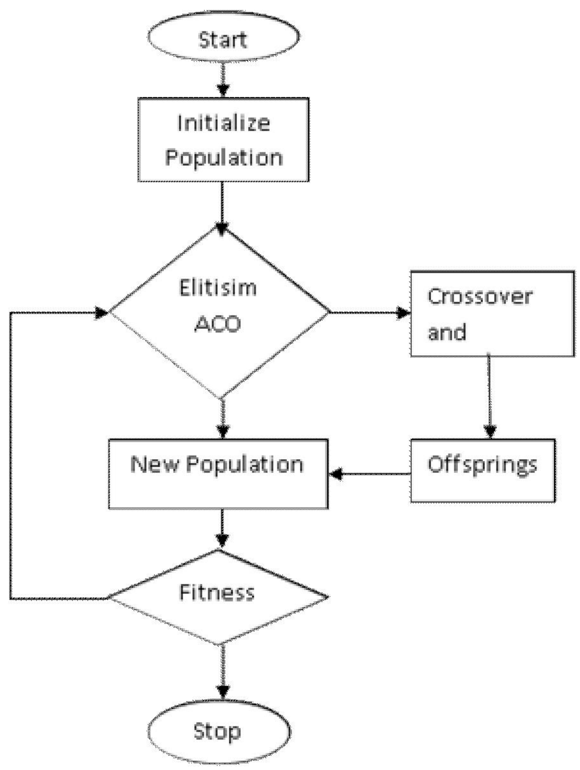

Figure 2 Block diagram of the proposed search process to minimize Makespan

Initialization

Both GA and ACO algorithms in HGAEAS are initialized with the same population. Primarily, P_s individuals are generated randomly as they form the population. The population formed can be considered as either chromosomes in GA or ants in ACO. After initialization, new generation individuals are created by enhancement, crossover, and mutation operations.

Enhancement

Both genetic algorithm and ant colony optimization operate on a shared population, serving to enhance accomplishing a shared optimization goal through population and the hybrid algorithm's pheromone updates. GA produces new solutions that replace population members improving population members average quality. Instead of reproducing such elite onto the next generation, the ACO algorithm first enhances them. This leads to improving pheromone traces updates, which further enhances ACO solution construction. On the other hand, ACO constructs new solutions replacing the population's worst solutions and thereby improving the average quality of the parent population members. These ACO enhanced elite when used as parents, the offspring produced are of better quality than those produced by original elite. This enhances the GA convergence increasing chances of further producing better GA solutions. It is believed that this collaborative pattern increases the chance of uncovering good solutions for the whole hybrid algorithm.

Artificial ants used in ACO are stochastic solution construction procedures probabilistically building a solution by iteratively adding solution components to partial solutions considering (i) heuristic information on the problem instance being solved, and (ii) pheromone trails changes dynamically to reflect agents' acquired search experience.

ACO interpretation as an extension of construction heuristics is appealing due to many reasons. ACO stochastic component allows the ants to build various solutions and thereby explores many larger solutions than greedy heuristics. Simultaneously use of heuristic information readily available for many problems guides ants towards most promising solutions. More importantly, ants' search experience influences in a way meaningful of reinforcement learning the solution construction in future algorithm iterations [21]. Additionally, the use of an ant colony gives the algorithm increased robustness and in many ACO applications the collective interaction of a population of agents is needed to solve problems efficiently [21].

The proposed ACO approach tries solving optimization problems through he following two steps:

- Pheromone models construct candidate solutions
- The pheromone values are modified by the candidate solutions as necessary to create high quality solutions.

An artificial ant agent builds a constructive information based solution called a pheromone, provided by previous ant agents which have already built solutions. After building new solutions, pheromone traces are updated by artificial ants considering existing solutions quality. A solution components set $C=\{c_1,.,c_n\}$ is used for constructing solutions in the form of solution component sequences by the artificial ants. Most ACO algorithms choose next solution component through probabilities which are also known as transition probabilities. They are defined as follows:

$$p\left(c_i \middle| s^p\right) = \frac{\tau_i^{\alpha}.\eta\left(c_i\right)^{\beta}}{\sum\limits_{c_j \in N\left(s^p\right)} \tau_i^{\alpha}.\eta\left(c_i\right)^{\beta}}, \quad \forall c_i \in N\left(s^p\right)$$

where η - weighting function depending on current partial solution,

η (c_i) - heuristic value assigned at each construction step to a feasible solution component $c_i \in N(s^p)$.

α and β - positive parameters determining relations between pheromone information and heuristic information.

Pheromone evaporation ensures rapid algorithm convergence toward a sub-optimal region implementing a form of overlooking so as to favor new area of exploration in the search space. One or more solutions from earlier iterations intensify the pheromone trail parameters value on solution components, which is updated as follows:

$$\tau_i \leftarrow \left(1-\rho\right)\tau_i + \rho \sum_{\left\{s \in S_{upd} \middle| c_i \in s\right\}} F\left(s\right)$$

for i=1,...,n. Here, $S_{upd}$is the set of solutions used for the update.

$\rho \in (0,1]$ - evaporation rate,

$F : S \rightarrow R^{+}$ - quality function such that $f(s) < f(s') \Longrightarrow F(s) \geq F(s')$, $\forall$ $s \neq s' \in S$. F(.)

Update rule are obtained by diverse specifications of S_{upd}, which is usually a subset of $S_{iter} \cup \{s_{bs}\}$, where S_{iter} represents the set of solutions constructed in the current iteration, and s_{bs} is the best-so-far solution. The solution thus found is input to the Genetic Algorithm's next stage which is the cross over stage.

Parents are chosen from ACO enhanced elites to warrant production of better individuals through a crossover operation. Parents for crossover operations are selected using tournament selection scheme where two enhanced elite are selected randomly. The fitness values of the selected elites are compared so that those with better fitness value are selected as a parent. The conventional view is that crossover is the more important of two techniques to explore a search space rapidly. Mutation provides limited random search, and ensures that no point in the search space has a zero examination probability. If the GA is implemented correctly, the population evolves over successive generations increasing the fitness of the best and average individuals in each generation towards a global optimum. Selection is the process of conserving the fittest individuals for the next generation. The selection part involves determination of individual fitness through a fitness function. The second part includes converting fitness function into an expected value followed by the last part where expected value is now converted to a specific number of offspring.

4. Result and Discussion

The effectiveness of the proposed scheduling method is assessed and evaluated using makespan. Makespan is the time taken by the grid system to complete the latest task. The experiments were conducted using 20 resource clusters and 100 jobs. Experiments were conducted using the hybrid HGAEAS grid scheduling. Figure 3 shows the Makespan time vs. number of iterations.

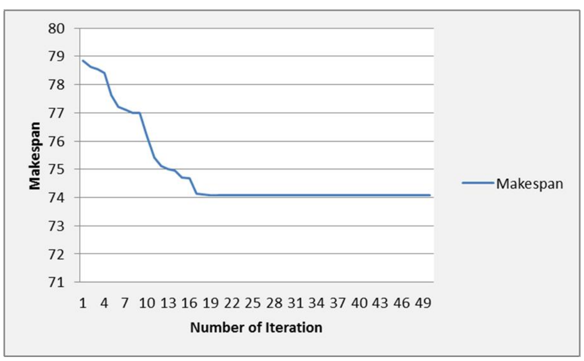

Figure 3: Makespan vs. Number of Iterations.

It is observed from Figure 3 that with the incorporation of elitism in the GA, optimal makespan is achieved. Also, the convergence is achieved in about 20 generations.

5. Conclusion

Grid computing environments scheduling aim is in efficient job mapping generated by applications to available resources. Jobs and resources can be dynamically added/ dropped to and from the system. Grid scheduling remains a global optimization challenge. As scheduling is a NP complete problem, this research examines meta-heuristic approaches for scheduling problems. The objective function of this study is minimizing overall job completion time or the application's makespan. This paper proposes a hybrid Genetic Algorithm (GA) incorporating Ant Colony Optimization (ACO) for grid scheduling. The approach aims to generate an optimal schedule to complete jobs within minimum time. Experiments demonstrate the proposed method's effectiveness.

References

1. Foster and C. Kesselman, Eds., The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, July 1998.
2. Foster, I., Kesselman, C., Nick, J. and Tuecke, S. "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", Technical Report, Open Grid Service Infrastructure WG, Global Grid Forum, 2002.
3. Subramaniam, N. "Grid computing: An overview", Idea Group, 2003.
4. Abraham, A., Liu, H., Zhang, W., & Chang, T. G. (2006). Scheduling jobs on computational grids using fuzzy particle swarm algorithm. In Knowledge-Based Intelligent Information and Engineering Systems (pp. 500-507). Springer Berlin/Heidelberg.
5. Krauter, K., Buyya, R., & Maheswaran, M. (2002). A taxonomy and survey of grid resource management systems for distributed computing. Software: Practice and Experience, 32(2), 135-164.
6. Xhafa, F., & Abraham, A. (2010). Computational models and heuristic methods for Grid scheduling problems. Future Generation Computer Systems, 26(4), 608-621.
7. Xhafa, F., & Abraham, A. (2008). Meta-heuristics for grid scheduling problems. Metaheuristics for Scheduling in Distributed Computing Environments, 1-37.
8. Kołodziej, J., & Xhafa, F. (2012). Integration of task abortion and security requirements in GA-based meta-heuristics for independent batch grid scheduling. Computers & Mathematics with Applications, 63(2), 350-364.
9. Xhafa, F., Gonzalez, J., Dahal, K., & Abraham, A. (2009). A GA (TS) hybrid algorithm for

scheduling in computational grids. Hybrid Artificial Intelligence Systems, 285-292.
10. Khan, S. U. (2012). Multi-level hierarchic genetic-based scheduling of independent jobs in dynamic heterogeneous grid environment. Information Sciences.
11. Agarwal, A., & Kumar, P. (2011). Multidimensional QoS Oriented task scheduling in Grid Environments. International Journal of Grid Computing & Applications (IJGCA) Vol, 2.
12. V. Di Martino, M. Mililotti, Sub optimal scheduling in a grid using genetic algorithms, 2004.
13. Wanneng Shu, Optimal Resource Allocation on Grid Computing Using Quantum Chromosomes Genetic Algorithm, 2007.
14. A. A. H. Liu and A. Hassanien. Scheduling jobs on computational grids using fuzzy particle swarm algorithm. Future Generation Computing Systems, 2009.
15. Pinel, F., Dorronsoro, B., & Bouvry, P. (2010, April). A new parallel asynchronous cellular genetic algorithm for scheduling in grids. In Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on (pp. 1-8). IEEE.
16. Delavar, A. G., Nejadkheirallah, M., & Motalleb, M. (2010, July). A new scheduling algorithm for dynamic task and fault tolerant in heterogeneous grid systems using Genetic Algorithm. In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on (Vol. 9, pp. 408-412). IEEE.
17. Xu, J., Lam, A. Y., & Li, V. O. (2010, May). Chemical reaction optimization for the grid scheduling problem. In Communications (ICC), 2010 IEEE International Conference on (pp. 1-5). IEEE.
18. Garg, S. K., Konugurthi, P., & Buyya, R. (2011). A linear programming-driven genetic algorithm for meta-scheduling on utility grids. International Journal of Parallel, Emergent and Distributed Systems, 26(6), 493-517.
19. Guo, S., Huang, H. Z., Wang, Z., & Xie, M. (2011). Grid service reliability modeling and optimal task scheduling considering fault recovery. Reliability, IEEE Transactions on, 60(1), 263-274.
20. E. Talbi, (2002) A taxonomy of hybrid meta-heuristics, J. Heuristics 8 (5) 541-564
21. Dorigo, M., & Stützle, T. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. In Handbook of metaheuristics (pp. 250-285). Springer US.

1/8/2013