Volume 5, Issue 7, July 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Review of Various Workflow Scheduling Techniques for Grid Computing Environment

Navneet Kaur
Scholar M.TECH (CSE)
India

Prabhjit Singh
AP, GIMET, Amritsar,
India

Abstract— Grid Computing provides resource sharing on large scale and provides platform for development of various computer science engineering applications. Effective scheduling is a key concern for the execution of performance-driven grid applications such as workflows. In this paper, various strategies have been purposed, including static and dynamic Strategies. The former schedules the tasks to resources before the actual execution time and later schedules them at the time of execution. Static scheduling performs better but it is not suitable for dynamic grid environment. Various heuristics or approximations are the performed to obtain near optimal solutions for minimum makespan, computation cost and communication cost of resources. However, in real environment, it is difficult to accurately predict the values due to heterogeneous and dynamic characteristics of grid environment. To have enhanced performance a new scheduling technique is likely to be purposed which will increase the efficiency of the heterogeneous grid also called AWT (Adaptive workflow technique). This approach schedule workflow tasks to the dynamic grid resources based on fact that job having early time (arrival time and service time) minimum will be executed firstly. The overall objective of this paper is to explore the existing research on the heterogeneous grid and scheduling application workflow techniques. Hence this paper ends up with the various challenges in the earlier techniques and also some future directions which can overcome the limitations of the earlier techniques.

Keywords— Adaptive workflow Scheduling, Dynamic Scheduling Strategies, earliest time first, Grid Computing, Resource Monitoring, Rescheduling.

I. INTRODUCTION

Recently, the rapid and advance development of networking technology and web has led to the possibilities of using large number of geographically distributed heterogeneous computing resources. These developments have led to the foundation of new paradigm known as Grid Computing [8]. Grid Computing is a type of parallel and distributed system that involves the integrated and shared use of resources. A grid has to provide strong incentive for participating sites to join and stay in it. The workflow scheduling in grid is one of the key challenges which mainly deals with assigning workflow tasks to the available grid resources. In general, scheduling tasks on distributed grid resources belongs to a class of NP-hard problems [14]. So heuristics or approximations are the preferred options to obtain near optimal solutions. Many heuristics and experiments have been devoted to this problem as discussed in literature [3,7] considering that accurate calculation is available for computation cost and communication cost of resources. However, in real actual environment, it is difficult to correctly predict the values due to heterogeneous and dynamic characteristics of the grid environment. The fluctuations in the resource availability (computing speed and links bandwidth) due to resource's local loads cause the original schedule to become sub optimal. Hence, it is a key challenge to maintain an application performance during its execution. In order to ensure high performance in dynamic grid environment, we considered the adaptive scheduling [9] where scheduling policy change dynamically as per the previous and current behaviour of the system to cope with the variations in the resource availability.
The rest of the paper is organized as follow- Section 2 begins with applications of parallel computing .Section 3 gives a brief introduction to the various scheduling application workflow techniques in grid computing. Section 4 describes the literature survey. Section 5 provides challenges of earlier techniques. Section 6 gives Conclusions and Future scope.

II. APPLICATIONS

A. DATA AND COMPUTATIONALLY INTENSIVE APPLICATIONS:

Grid computing has been applied to computationally-intensive scientific, mathematical, and academic problems like drug discovery, economic forecasting, seismic examination back and data processing in support of e-commerce. For e.g a chemist may use hundreds of processors to display thousands of compounds per hour. Teams of engineers globally pool resources to examine terabytes of structural data. Also meteorologists seek to visualize and explore petabytes of climate data with huge computational demands**.** [21]

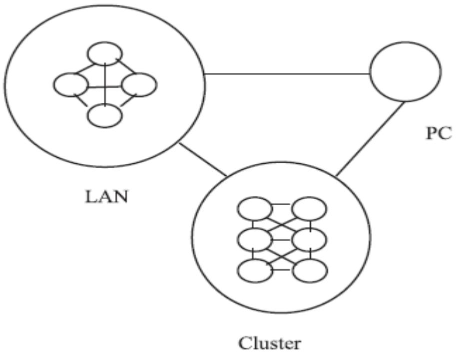

Fig 1: Architecture of Grid [16]

B. RESOURCE SHARING

Grids are used in computers, storage, sensors, network application that involves breaking the problem into discrete pieces [19], in which discovery and scheduling of tasks and workflow is done. It is widely used in coordinated problem solving distributed data analysis, computation, collaboration etc.

C. OTHER APPLICATIONS

(i) Distributed supercomputing- It uses grid to deal with problems that cannot be solved on a single system.
(ii) High-Throughput Computing-Uses the grid to schedule huge numbers of loosely coupled or independent tasks, with the purpose of putting unused processor cycles to work.
(iii) On-Demand Computing -Uses grid capability to meet up short-term requirements for resources that are not locally available.

III. SCHEDULING TECHNIQUES

Scheduling techniques for workflows are broadly categorized as:

Static Scheduling

In the static mode, every task comprising the job is assigned one time to a resource. [20] Thus, the assignment of an application is static, and a firm estimation of the cost of the computation can be made in move ahead of the actual execution. One of the major profits of the static model is that it is easier to program from a scheduler's point of observation. The assignment of tasks is fixed and estimating the cost of jobs is also simplified. The static model allows a "global view" of tasks and costs. Heuristics and experimental calculations can be used to decide whether to incur somewhat higher processing costs in order to keep all the tasks concerned in a job on the similar or tightly-coupled nodes.

Dynamic Scheduling

Dynamic scheduling is generally applied when it is difficult to approximate the cost of applications or jobs are approaching online dynamically also called online Scheduling. A good example of this scenario is the job queue management in some meta-computing systems like Condor and Legion [20]. Dynamic task scheduling has two major components (a) system state estimation (additional than cost estimation in static scheduling) and decision making. (b)System state assessment that involves collecting state information throughout the Grid and constructing estimation on the basis of the estimation, decisions are made to assign a task to a selected resource.

MET (Minimum Execution Time) : MET assigns each task to the resource by means of the best expected execution time for that task, no matter whether this resource is accessible or not at the current time. The inspiration behind MET is to give each task its best machine. This heuristic is not appropriate to heterogeneous computing environments.

MCT (Minimum Completion Time): MCT assign each task, in a random order, to the resource with the minimum expected completion time for that task. This cause some tasks to be assigned to machine that do not have the minimum execution time for them. The insight behind MCT is to combine the benefits of opportunistic load balancing (OLB) and MET, while avoiding the conditions in which OLB and MET perform poorly.

Min-min [5]: Min-Min begins with a set of tasks which are all unassigned. First, it computes minimum completion time for all tasks on all resources. Then among these minimum times the minimum value is selected which is minimum time among all the tasks on any resources. Then with this aim task is scheduled on the resource on which it takes the minimum time and the available time of that resource is updated for all other tasks .It is updated in this manner, suppose a task is assigned to a machine and it takes 20 seconds on the assigned machine, then the execution time of all other tasks on this assigned machine will be increased by 20 seconds. After this the unassigned task is not considered and the same process is repeated until all the tasks are assigned resources.

Max-min [5]: Max-Min is almost same as min-min algorithm. In this after finding out the completion time, the minimum execution times are found out for each and every task. Then among these minimum times the maximum value is selected which is the maximum time among all the tasks on any resources .Then that task is scheduled on the resource on which it takes the minimum time and the available time of that resource is updated for all other tasks. The updating is done in the same manner as min-min. All the tasks are assigned resources by this procedure.

Heterogeneous earliest finish time (HEFT) [1]: Topcuoglu et.al, presented the HEFT algorithm. It is most popular list based experiment. In this, rank is computed using average execution time and average communication time. It orders the tasks based on priorities and then assign them suitable resources to achieve high performance This algorithm finds the average execution time of each task and also the average communication time between the resources of two tasks. Then the tasks in the workflow are ordered on a rank function. Then the task with a higher rank value is given higher priority. The tasks are scheduled in priorities and each task is assigned to a resource that complete the task in earliest time.

Adaptive workflow scheduling [16]: The technique involves static task scheduling, cyclic resource monitoring and rescheduling the left over unexecuted tasks in order to deal with changes and fluctuations occurring at run time and to achieve minimum execution time (makespan) of the workflow grid application. The method of proposed AWS differs from other approaches in literature by taking into account the dynamic availability of resources together computing nodes and communication links due to presence of local load or load by other users. It considers (i) Degradation of resource performances especially computing speed of nodes and network links bandwidth as a source for triggering rescheduling. (ii) Evaluate the profit of rescheduling considering cost of reevaluating the schedule and operating cost due to transfer of data. (iii) Accessibility of newly added resources.

4-levels/RMFF job scheduling [17] -In 4-levels model, the grid resources are connected resting on a form of tree structure. The grid is separated by clusters, in each cluster there is an owner i.e. cluster manager. Each cluster is constructed from different sites. Within each site, the actual resources of the grid are present. The naming of 4-levels comes from the levels of this model namely grid, cluster, site and resources levels .One of the advantages of this model is its organized connectivity, which gives the administrator of the grid the permission to organizing it into sites and cluster according to his preference.

Adaptive workflow technique (AWT): An AWT is a methodology for improving performance of parallel computing jobs by reducing makespan of jobs. The basic idea of algorithm is to use their arrival time and service time of jobs as a parameter in achieving efficient improvement rate of jobs and also their Mean response time. In this study, we perform optimization on existing AWS algorithm, thereby reducing the makespan of jobs and propose a new technique called AWT (Adaptive workflow technique) and algorithm called ETF (Earliest Time First). This approach schedule workflow tasks to the dynamic grid resources based on fact that job having early time (arrival time and service time) minimum will be executed firstly.. The proposed earliest time first workflow scheduling (ETF) approach involves initial static scheduling, calculating scheduling heuristics, and calculating scheduling attributes with the aim to achieve the minimum execution time for workflow application. The approach is different from other techniques in literature to best of our knowledge.

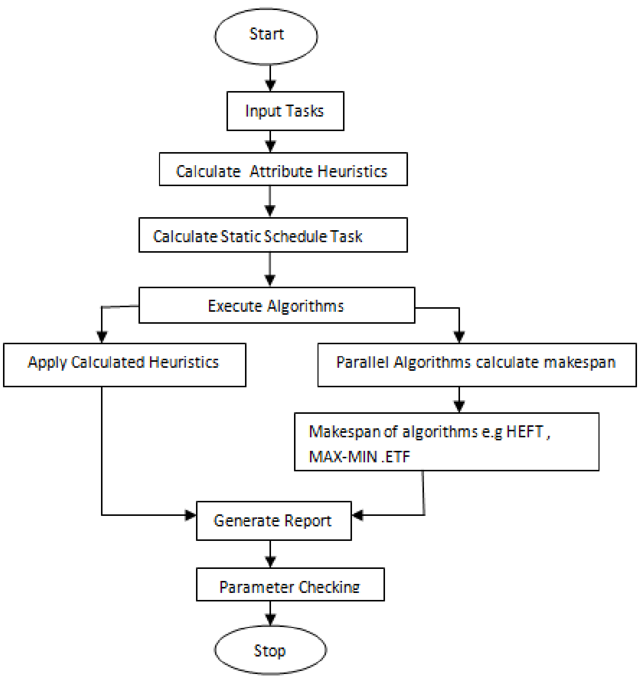

Fig 2.Flow Chart of ETF

IV. LITERATURE SURVEY

The problem of scheduling for workflow (DAG-based) has been discussed here. Most of work attempts to achieve minimum execution time (makespan) on heterogeneous grid environment.

In **2002** H.topcuoglu [1]-presents Heterogeneous Earliest Finish Time (HEFT) and Critical –Path-on-a-processor (CPOP) the Two most popular list based experiment. In this, rank is computed using average execution time and average communication time. It orders the tasks based on priorities and then assign them suitable resources to achieve high performance

IN **2004** R. Sakellario[4] paper presents a low-cost rescheduling policy SLACK which considers rescheduling at a few, carefully selected points during the execution. This policy achieves performance results, which are comparable with those achieved by a policy that dynamically attempts to reschedule before the execution of every task. It uses the concept of spare time, which does not have impact on the schedule length of the workflow. If execution time of task goes beyond the spare time then only rescheduling event is triggered.

In **2005** A. Mandal, K . Kennedy, C. Koelbel, G. Marin, J. Mellor-Crummey, B. Liu, L. Johnson [5] author described new strategiesi.e MAX-MIN, MIN-MIN for scheduling and executing workflow applications on grid resources using the GrADS [Ken Kennedy et al., 2002] infrastructure. The results of experiments show that strategy of performance model based, in-advance heuristic workflow scheduling results in 1.5 to 2.2 times better makespan than other existing scheduling strategies.

F. Berman, H. Casanova, A. Chien, K. Cooper [6] purposed recent extensions to the GrADS software framework in which a new approach to scheduling workflow computations, applied to a 3-D image reconstruction application and a simple stop/migrate/restart approach is applied to rescheduling Grid applications also a process-swapping approach to rescheduling, applied to an N-body simulation

L.F. Bittencourt, E.R.M. Madeira, F.R.L. Cicerre, L.E. Buzato[7] presents a new scheduling algorithm- PCH algorithm that uses a hybrid clustering-list-scheduling strategy. In this tasks with communication cost are grouped together and assigned to the same resource in a cluster. It aims to reduce the schedule length by reducing the communication Cost.

In **2007** Z. Yu, W. Shi [9] In this paper authors proposed and model evaluation metrics for the Grid Service performance. In addition, also proposed a low-overhead rescheduling method, referred to as adaptive list scheduling for service (ALSS), to adapt to the dynamic nature of a grid environment. ALSS provides stable performance for workflow applications, even in abnormal circumstances.

In **2010** S.H. Chin, T. Suh, H.C. Yu [10] In this author proposed an adaptive rescheduling algorithm AHEFT based on static strategy. This paper propose a adaptive rescheduling concept, which allow the workflow planner works collaboratively with the run time executor and reschedule in a proactive way had the grid environment changes significantly.

In **2011** H.A. Sanjay, S.S. Vadhiyar[11]-purposed three strategies or algorithms for deciding when and where to reschedule parallel applications that execute on multi-cluster Grids. Using large number of simulations, it was shown that the rescheduling plans developed by the algorithms can lead to large decrease in application execution times when compared to executions without rescheduling on dynamic Grid resources.

In **2012** A. Olteanu, F. Pop, C. Dobre, V. Cristea[12]- proposes a generic rescheduling algorithm rescheduling i.e used for wide and large scale distributed systems (RE-LSDS) to support fault tolerance and resilience.. The system was evaluated and implemented in a real-world implementation for a Grid system. The proposed method supports fault tolerance and offers an improved mechanism for resource management.

In **2013** M. Rahman, R. Hassan, R. Ranjan, R. Buyya[13]- This Paper introduced the dynamic critical path based workflow scheduling algorithm for grid namely DCP-G that provides efficient schedule in static environment. Additionally, it adapts to the dynamic grid environment, where resource information is updated and changed after fixed interval and rescheduling (Re-DCP-G) is performed if needed. It also describe the exactness of hybrid heuristic algorithm that combine the features of the adaptive scheduling technique with meta-heuristics for optimizing execution time and cost in dynamic cloud environment.

In **2014** E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P.J. Maechling, K. Wenger, Pegasus[15]- Describes the design, development and evolution of the Pegasus Workflow Management System. This performs mapping of abstract workflow descriptions onto distributed computing infrastructures in order to achieve reliable and scalable workflow execution.

V. CHALLENGES OF SCHEDULING ALGORITHMS IN GRID COMPUTING

Although a Grid also falls into the category of distributed parallel computing environments, it has a lot of distinctive characteristics which make the scheduling in Grid environment highly difficult. **[21]** An satisfactory Grid scheduling system should overcome these challenges to manipulate the promising potential of Grid systems, providing high-performance services. The challenges forced by Grid systems are examined following:

1. Resource Heterogeneity- A computational Grid mainly has two categories of resources: networks and computational resources. Heterogeneity exists in both categories of resources. Firstly the networks used to interconnect these computational resources may differ significantly in terms of their bandwidth and communication protocols. Second, computational resources are usually heterogeneous in that these resources may have different hardware, such as instruction set, computer architectures, number of processor, CPU speed and different software such as different operating systems, file systems, cluster management software. The heterogeneity results in differing capability of processing jobs. Therefore an adequate scheduling system should address the heterogeneity and further influence different computing power of diverse resources.

2. Site Independence- Generally a Grid may comprise multiple administrative domains. Each domain shares a common security and management policy. Each domain generally authorizes a group of users to use the resources in the domain. A single overall Performance goal is not realistic for a Grid system since each site has its own performance goal and scheduling decision is made independently of other sites. Local priority is another important issue where each site within the Grid has its own scheduling policy. In this assumption, the scheduler has sufficient information of resources and therefore effective scheduler is much easier to obtain. But in Grid environments, the Grid scheduler has only limited control over the resources. Site autonomy significantly complicates the design of effective Grid scheduling.

3. Resource Non-dedication -Because of non-dedication of resources its usage contention is a major issue. Competition may exist in both computational resources and interconnection networks. Due to the non-dedication of resources, a resource may join multiple grids at the same time. The workloads from both local users and other Grids share the resource concurrently. [21]So, Schedulers must be able to think about the effects of contention and calculate the available resource capabilities.

4. Application diversity -The problem arises because the Grid applications are from a wide range of users, each having its own particular requirements. For example, various applications may require sequential execution and some application consists of a set of independent jobs and others may consist of a set of dependent jobs. In this situation, building a general-purpose scheduling system seems very difficult. An adequate scheduling system should be able to handle a variety of applications**.**

5. Dynamic performance: In traditional parallel computing environments, such as a cluster the group of resources is assumed to be fixed or stable. In a Grid environment, dynamics exists in both the networks and computational resources. The network bandwidth, the availability and capability of computational resources will exhibit dynamic behavior. On one side new resources may join the Grid, and on the other hand, some resources may become unavailable due do problems such as network failure. Also when new resource joins the Grid, the scheduler should be able to detect it automatically and control the new resource in the later scheduling decision making. [21]. These challenges pose significant obstacles on the problem of designing an efficient and effective scheduling system for Grid environments. As a result, new scheduling frame work must be developed for Grids, which should reflect the unique characteristics of grid systems.

VI. CONCLUSION AND FUTURE WORK

In the application based on grid computing, an effective algorithm is required in order to have minimum makespan and computational cost parameters. To diminish this problem a new technique has been proposed called AWT (ADAPTIVE WORKFLOW TECHNIQUE) which has improved the performance of the grid. As our literature indicates that various scheduling techniques are available for calculating the makespan. Paper presents a high-performance and makespan reducing algorithm to find enhanced solutions for scheduling in grid computing systems. The algorithm uses arrival time and service time minimum as a concept to fine-tune the solutions obtained by HEFT and MAX-MIN algorithm to further perk up the scheduling results in terms of makespan. So to remove these issues ETF (EARLY TIME FIRST) algorithm will be proposed in near future. This is primarily different from other algorithm, which requiring a much longer computation time. Early time first can be applied to grid computing systems to enhance the performance of scheduling problems. The Researchers can implement and can develop the new approach of calculating makesapan . In this paper various scheduling techniques have been described. So for the future work we can extend and explore the new scheduling techniques in order to provide high and stable performance to workflow application.

REFERENCES

[1] H. Topcuoglu, S. Hariri, M.-Y. Wu, “Performance-effective and low-complexity task scheduling for heterogeneous computing”, in: IEEE Transactions on Parallel and Distributed Systems, 13(3), 2002, pp. 260-274.

[2] E. Huedo, R.S. Montero, I.M. Llorente, “Experiences on adaptive grid scheduling of parameter sweep applications”, in: Proceedings of the 12th Euromicro Conference on Parallel, Distributed and Network-based Processing, 2004, pp. 28-33.

[3] R. Bajaj, D.P. Agarwal, Improving scheduling of tasks in a heterogeneous environment, in: IEEE Transactions on Parallel and Distributed Systems, 15(2), 2004, pp. 107-118.

[4] R. Sakellariou, H. Zhao, A low-cost rescheduling policy for efficient mapping of workflows on grid systems, Scientific Programming 12 (4) (2004) 253-262.

[5] A. Mandal, K. Kennedy, C. Koelbel, G. Marin, J. Mellor-Crummey, B. Liu, L. Johnsson, “Scheduling strategies for mapping application workflows onto the grid”, in: In Proceedings. 14th IEEE International Symposium on High Performance Distributed Computing, 2005, pp. 125-134

[6] F. Berman, H. Casanova, A. Chien, K. Cooper, et al., New grid scheduling and rescheduling methods in the GrADS project, Int J Parallel Program 33 (2-3)(2005) 209-229

[7] L.F. Bittencourt, E.R.M. Madeira, F.R.L. Cicerre, L.E. Buzato, A path clustering heuristic for scheduling task graphs onto a grid (short paper), in: Proceedings of the 3rd ACM International Workshop on Middleware for Grid Computing, 2005. Grenoble, France.

[8] R. Buyya, S. Venugopal, “A Gentle Introduction to Grid Computing and Technologies”, CSI Communications, July 2005.

[9] Z. Yu, W. Shi, “An adaptive rescheduling strategy for grid workflow applications”, in: In IEEE International Parallel and Distributed Processing Symposium, 2007, IPDPS, 2007, pp. 1- 8.

[10] S.H. Chin, T. Suh, H.C. Yu, Adaptive service scheduling for workflow applications in service-oriented grid, J. Supercomputing 52 (3) (2010) 253-283.

[11] H.A. Sanjay, S.S. Vadhiyar, Strategies for rescheduling tightly-coupled parallel applications in multi-cluster grids, J. Grid Comp. 9 (3) (2011) 379-403

[12] A. Olteanu, F. Pop, C. Dobre, V. Cristea, “A dynamic rescheduling algorithm for resource management in large scale dependable distributed systems”, Comp. Math. Appl. 63 (9) (2012) 1409-1423.

[13] M. Rahman, R. Hassan, R. Ranjan, R. Buyya, "Adaptive workflow scheduling for dynamic grid and cloud computing environment" , Conc. Comp. Prac. Exp. 25 (13) (2013) 1816-1842.
[14] M. Gareym, D. Johnson, Computers and Intractability: "A Guide to the Theory of NP-completeness", WH Freeman & Co., San Francisco, 1979.
[15] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P.J. Maechling, K. Wenger Pegasus, a workflow management system for science automation, Future Generation Computer Systems (2014).
[16] Ritu Garg*, Awadhesh Kumar Singh"Adaptive workflow scheduling in grid computing based on dynamic resource availability" Engineering Science and Technology, an International Journal xxx (2015) 1- 14
[17] Web reference at http://airccse.org/journal/ijgca/papers/4313ijgca01.pdf
[18] Web reference at http://www.sciencedirect.com/science/article/pii/S2215098615000087
[19] Web reference at http://www.ibmpressbooks.com/articles/article.asp?p=169508&seqNum=4
[20] Web reference at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.462.116&rep=rep1&type=pdf
[21] Web reference at http://www.cs.sjtu.edu.cn/~yzhu/reports/SJTU_CS_TR_200309001.pdf