Probabilistic Availability based Task Scheduling Algorithm

Chitra S[1],
[1]*Research Scholar, NHCE, Bangalore, VTU, India,*
chitraravi.cr@gmail.com

Dr. Prashanth C.S.R[2]
[2]*HOD, Department of Computer Science and Engineering, NHCE, Bangalore, VTU, India,*
drprashanthcsr@gmail.com

Abstract- In high performance computing environment such as grids and clouds, resource availability is highly dynamic, unpredictable and unreliable, due to various factors such as system load, system failure, available network bandwidth, network failure, introduction of new resources, contention among remote task and the local tasks of resources and so on. In such environments, the dynamic availability of resources can affect execution of parallel applications. Parallel applications modelled by Directed Acyclic Graphs (DAG) scheduled on a network of heterogeneous processors so as to minimize finish time is a known NP complete problem. The Heterogeneous Earliest Finish Time (HEFT) is a static task scheduling algorithm that assumes probability of resource availability is 100% whereas in real life situations the case is not true. The temporal availability of resources is to be considered in task scheduling decisions. We propose a new Probabilistic Availability based Task Scheduling Algorithm (PATSA) where tasks are scheduled not only based on earliest finish time of task but also on resource availability probability so as to minimize makespan. Our algorithm was compared with existing HEFT algorithm for a certain set of random DAGs based on performance metrics such as Makespan, Schedule Length Ratio and Speedup.

Keywords-Directed Acyclic Graphs, High Performance Computing, Makespan, Resource Availability, Schedule Length Ratio, Speedup

I. INTRODUCTION

Due to administrators' policies, resources in a heterogeneous computing system show different availability properties[6]. The performance and availability of the resources, which are already heterogeneous, may vary dynamically, even during the course of an execution. Effective task scheduling is a key concern for the execution of high performance applications[5]. In heterogeneous computing environment, there is uncertainty in system parameters[9]. So it is hard to estimate task execution and transfer times on resource queues. Also due to CPU load, actual execution time may vary from the estimated execution time. Heterogeneous processor speeds affect task execution times and hence execution time of each application on various processors varies.

A new algorithm called Probabilistic Availability based Task Scheduling Algorithm (PATSA) is developed to schedule tasks on a finite number of processors. This algorithm is based on assigning a priority for each node by using Rank value, and selecting the processor based on resource availability probability and earliest finish time.

Section 2 gives problem definition. Related work is discussed in Section 3. Section 4 describes the architecture model for PATSA algorithm. Section 5 presents our PATSA algorithm. Results are discussed in Section 6. In Section 7 conclusions are given.

II. PROBLEM STATEMENT

The problem can formally be defined as "To develop efficient heuristic to schedule the tasks representing a parallel application, modelled by Directed Acyclic Graph on a network of heterogeneous processors such that precedence constraints are satisfied and minimizing overall execution time (finish time) considering the probability of resource availability".

III. RELATED WORK

A list scheduling algorithm such as HEFT [1], has two phases: Task prioritizing and Processor selection. Tasks are prioritized for scheduling based on upward ranking. Tasks are arranged in decreasing order of ranku to generate the task list. Tie-break is random. Then, in the processor selection phase, the selected task is assigned to processor with minimum earliest finish time using insertion-based approach. The complexity of the HEFT algorithm is O(e x r) where ***e*** is the number of edges and ***r*** is the number of processors.

Different policies of resource owners/administrators regarding the availability of resources in the grids, different resource failure properties does not guarantee consistent availability of resources for scheduling tasks[6]. A probabilistic estimate of resource availability can be considered while making scheduling decisions. Resource availability information is critical to achieve better system performance[13].

IV. ARCHITECTURE MODEL

Parallel application in high performance computing environment is modelled by Directed Acyclic Graph (DAG). Let P = $\{p_1,p_2,.....p_m\}$ be the finite number ***m*** of heterogeneous processors to process ***n*** parallel applications $\{J_1,J_2,.....J_n\}$. A parallel application J_i has several tasks $\{t_1, t_2,......t_k\}$. There is precedence

relation between the tasks. A resource is a network node which can be either a standalone machine or a cluster. Hence we define the set of resources R = {R_1,R_2,...R_n}. Each resource R_i has a queue Q_i □ Q = {Q_1,Q_2,.....Q_l } attached. Every Q_i contains a set of tasks T'□T assigned to be solved on resource. Resource Availability[12] is the probability that the computing resource is continuously functional at any random period of time.
A sample task graph[1] is shown in Figure 1. Table 1 shows the computation cost matrix.

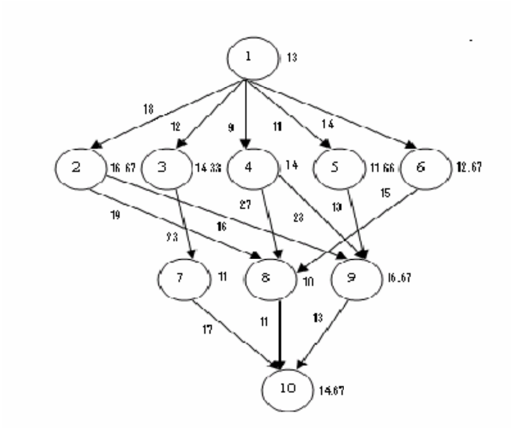

Figure 1 : Example Task Graph

Table 1 : Computation Cost Matrix

Task	P_1	P_2	P_3
1	14	16	9
2	13	19	18
3	11	13	19
4	13	8	17
5	12	13	10
6	13	16	9
7	7	15	11
8	5	11	14
9	18	12	20
10	21	7	16

V. PROBABILISTIC AVAILABILITY BASED TASK SCHEDULING ALGORITHM

The temporal availability of resources leads to serious performance consequences while executing parallel applications in grids or clouds. Probabilistic Availability based Task Scheduling Algorithm (PATSA) is novel static task scheduling algorithm for a finite number of processors considering probability of resource availability.
There are two phases in PATSA algorithm: Task prioritizing phase and Processor Selection phase. In the first phase, a priority list is formed by ordering tasks in the decreasing order of upward rank or blevel. Tie-break is random. In the second phase, the processor is selected based on the earliest finish time that takes into account the resource availability constraint. Reduced resource availability increases the task execution time and hence the makespan. For normalization, the EFT of each task is adjusted by a factor which is the ratio of maximum probability of resource availability for the task to the individual probability of resource availability. The time complexity is O(e x r) where 'e' is the number of edges and 'r' is the number of processors.
The task characteristics to be considered are task execution time, start time of task, finish time of task and task priority.

```
ALGORITHM PATSA
BEGIN PATSA
// N is set of vertices of DAG
// P is  set of Processors
//RdyTskLst is the ready task list
// $pr(n_i, p_j)$ is probability of availability of processor $p_j$ for a given task $n_i$.
// $pr\_max$ is the maximum of probability of availability of all the processors for a given task $n_i$.
For each $n_i$ in N
        Calculate ranku($n_i$)
End For
RdyTskLst ← StartNode
While RdyTskLst is NOT NULL do
$n_i$ ← Node in the RdyTskLst with maximum ranku
For each processor $p_j$ in P
$EST(n_i, p_j) =$
 $\max\ (T_{avail[j]}, \max_{n_m \in pred(n_i)}(EFT(n_m, p_k) + c_{m,i}))$
$EFT(n_i, p_j) = w_{i,j} + EST(n_i, p_j)$
If $pr\_max - pr(n_i, p_j) > 0.5$
  $EFT_{pr}(n_i, p_j) = EFT(n_i, p_j) * pr\_max/pr(n_i, p_j)$
Else
  $EFT_{pr}(n_i, p_j) = EFT(n_i, p_j)$
End If
End For
Allocate node $n_i$ on processor $p_j$ with least $EFT_{pr}$
Update T_Avail[$p_j$] and RdyTskLst
End While
END PATSA
```

Figure 2: PATSA Algorithm

Execution time of a task varies with each processor due to heterogeneity of processors. Also the estimated times and actual times vary. The Computation Cost Matrix represents the expected execution time for each task on different resources. Probability of Resource Availability Vector has probability values of availability of processor for each task.
Random graphs used to test this algorithm are generated by a directed acyclic random graph generator algorithm[11]. This takes number of nodes as input and generates a weighted directed acyclic graph, where number of edges are selected randomly based on number of nodes. Communication cost weighted sparse matrix is generated randomly.

VI. RESULTS AND DISCUSSIONS

The comparative results of our algorithm PATSA and HEFT based on Performance Metrics [1] is discussed below.

a. Performance Metrics:

Makespan is the Actual Completion Time of exit node in the DAG which represents a parallel application. It is also called **Schedule Length**.
Schedule Length Ratio (SLR) is ratio of makespan to summation of minimum computation cost of critical path nodes.
Speedup is ratio of sequential execution time of parallel application to its parallel execution time. It is the ratio of minimum of summation of computation costs of tasks on processors to makespan.

b. Experimental Setup:

Our simulated framework consists of Random DAG Generator Program to construct task graphs followed by our novel task scheduling algorithm based on resource availability constraint called PATSA algorithm to generate schedule outputs and also HEFT algorithm for comparison. Various Random Directed Acyclic Graphs with 10,20,30,40,50 task nodes were generated to test our algorithm. The makespan was determined and then other performance metrics such as Schedule Length Ratio and Speedup were computed based on schedules. These performance metrics were compared with HEFT algorithm simultaneously for the same set of experiments.

c. Results:

The average makespan, average SLR and average Speedup of values obtained by considering various random DAGs of different sizes of task nodes are used to compare HEFT and PATSA. The graphs depicting the results of comparison of HEFT and PATSA based on average makespan as in Table 2, average SLR as in Table 3 and average Speedup as in Table 4 are shown in Figures 3, 4 and 5 respectively.

Table 2 : Comparison of Average Makespan

Number of Task nodes	Average Makespan	
	HEFT	PATSA
10	75.77	110.85
20	161.91	192.91
30	222.89	283.06
40	273.91	326.00
50	329.91	399.73

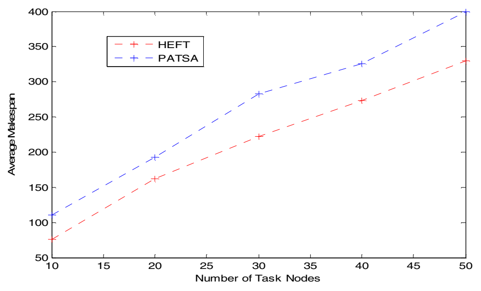

Figure 3 : Comparison of HEFT and PATSA based on Average Makespan

Table 3: Comparison of Average SLR

Number of Task nodes	Average SLR	
	HEFT	PATSA
10	0.51	0.74
20	0.48	0.58
30	0.44	0.56
40	0.46	0.55
50	0.44	0.54

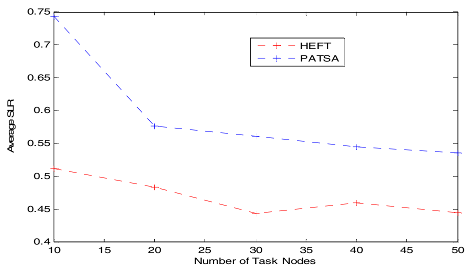

Figure 4 : Comparison of HEFT and PATSA based on Average SLR

Table 4 : Comparison of Average Speedup

Number of Task nodes	Average Speedup	
	HEFT	PATSA
10	1.22	1.01
20	1.10	0.93
30	1.23	0.98
40	1.30	1.11
50	1.38	1.15

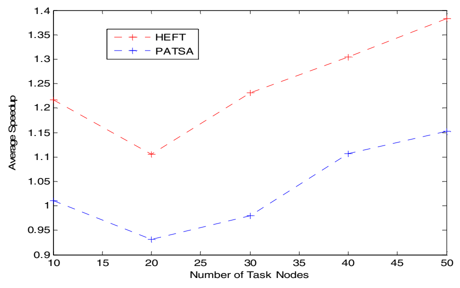

Figure 5 : Comparison of HEFT and PATSA based on Average Speedup

VII. CONCLUSIONS

In real life situations, there is no guarantee of 100% resource availability or processor availability in high performance computing environment. The global tasks to be scheduled in such situations cannot assume that there could be 100% processor availability. Reduced resource availability increases the task execution time and hence the makespan. We propose a new PATSA algorithm which considers resource availability factor and produces results comparable to HEFT. It is observed that when the algorithm is implemented with assumption of 100% resource availability, the makespan is lesser compared to when the algorithm is implemented with resource availability probability of less than 1. Between PATSA that considers resource availability probability and HEFT that does not consider resource availability, the average makespan as well as SLR vary only by 20%. Variation in Average Speedup is also about 20%. We also observe that when the number of task nodes is less, performance is better and with increase in the number of nodes, difference in makespan of both algorithms marginally increases. Since our algorithm considers resource availability, scheduling global tasks is more reliable than in HEFT.

REFERENCES

[1]Haluk Topcuoglu, Salim Hariri and Min-You Wu, “Performance Effective and Low Complexity Task Scheduling for Heterogeneous Computing", IEEE Transactions on Parallel and Distributed Systems, Vol.13, No.3, March 2002, : 260-274

[2] Prashanth C. SaiRanga: Scheduling Directed A-cyclic Task Graphs on Heterogeneous Network of Workstations to Minimize Schedule Length. ICPP Workshops 2003: 97-103

[3] D. Kondo et al. Characterizing and evaluating desktop grids: An empirical study, 2004.

[4] Sanjeev Baskiyar, Prashanth C. SaiRanga: Scheduling DAGs on Heterogeneous Network of Workstations to Minimize Finish Time. I. J. Comput. Appl. 13(4): 168-175 (2006)

[5]Brent Rood and Michael J. Lewis," Multi-State Grid Resource Availability Characterization", IEEE 8th Grid Computing Conference, 2007

[6] Farrukh Nadeem, Radu Prodan, Thomas Fahringer,"Characterizing, Modeling and Predicting Dynamic Resource Availability in a Large Scale Multi-Purpose Grid", Eighth IEEE International Symposium on Cluster Computing and the Grid, 2008

[7]Ching-Hsien Hsu; Zhan, J.; Wai-Chi Fang; Jianhua Ma; , "Towards Improving QoS-Guided Scheduling in Grids," ChinaGrid Annual Conference, 2008. ChinaGrid '08. The Third , vol., no., pp.89-95, 20-22 Aug. 2008

[8] Foad Loatfifar, Hadi Shahriar Shahhoseini,"A Low-complexity Task Scheduling Algorithm for Heterogeneous Computing Systems", Third Asia International Conference on Modelling and Simulation,2009

[9]Brent Rood and Michael J. Lewis,"Resource Availability Prediction for Improved Grid Scheduling"

[10] Chtepen, M.; Claeys, F.H.A.; Dhoedt, B.; De Turck, F.; Demeester, P.; Vanrolleghem, P.A.; , "Adaptive Task Checkpointing and Replication: Toward Efficient Fault-Tolerant Grids," Parallel and Distributed Systems, IEEE Transactions on , vol.20, no.2, pp.180-190, Feb. 2009

[11] Yinfeng Wang, Zhijing Liu, Wei Yan, “Algorithms for Random Adjacency Matrixes Generation Used for Scheduling Algorithms Test”, International Conference on Machine Vision and Human-Machine Interface (MVHI), 2010

[12] Amit Agarwal and Padam Kumar, Multidimensional QoS Oriented Task Scheduling In Grid Environments, International Journal of Grid Computing & Applications (IJGCA) Vol.2, No.1, March 2011

[13] Alexandru Iosup et al. On the dynamic resources availability in grids. In IEEE International Symposium on Cluster Computing and the Grid, Rio de Janeiro, Brazil, May 14-17, 2007.

[14] B. Rood and M. J. Lewis. Multi-state Grid Resource Availability Characterization. In International Conference on Grid Computing, Austin, TX, September 17-19, 2007.