ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

A Partial Critical Path Based Approach for Grid Workflow Scheduling

Anagha Sharaf[1], Suguna.M[2]

PG Scholar, Department of IT, S.N.S College of Technology, Coimbatore, Tamilnadu, India [1]

Associate Professor, Department of IT, S.N.S College of Technology, Coimbatore, Tamilnadu, India [2]

ABSTRACT: Grid Computing is a technique in which the idle systems in the Network and their CPU cycles can be efficiently used by uniting pools of servers, storage systems and networks into a single large virtual system for resource sharing dynamically. Utility grids are new service models in heterogenous distributed systems. Utility grids enable users to specify the quality requirements they need. The main challenge in grid computing is the efficient workflow scheduling. For scheduling workflows considering QoS specifications of user a metaheuristic algorithm is introduced. The algorithm is based on the concept of partial critical path. The algorithm includes two phases 1.deadline phase and 2.Planning phase. User submits deadline and other QoS specifications in deadline phase. The cheapest service is allotted to tasks inorder to satisfy QoS specifications and deadline in planning phase.

Keywords: Utility grids, workflow scheduling, partial critical path

I. INTRODUCTION

Grid computing is the technique of flexible and coordinated resource sharing. With grid computing users could access geographically distributed resources. Grids use a layer of middleware to communicate with and manipulate heterogeneous hardware and data sets. In some fields—astronomy, for example—hardware cannot reasonably be moved and is prohibitively expensive to replicate on other sites. In other instances, databases which are vital for some projects could not be replicated and transferred to multiple sites. Grid computing overcomes these obstacles. Grid computing enables the virtualization of distributed computing resources such as processing, network bandwidth, and storage capacity to create a single system image, granting users and applications seamless access to vast IT capabilities.
Utility grids are the emerging service provisioning model in heterogenous distributed systems. Utility grids make the distributed resources available in market for price. The main difference between traditional grids and utility grid is the Quality of service. Utility grids enable the users to negotiate with service providers for the required Quality of service and also on the price.

\Workflow could be described as collection of tasks that can be processed on distributed resources in a well defined manner to achieve specific goal. It is a common model for describing wide range of applications in distributed systems. Workflows could be represented using Directed Acyclic Graph (DAG), where each computational task is represented by a node and dependency among tasks is represented using the edges.

A metaheuristic QoS based workflow scheduling algorithm called partial critical path algorithm is introduced. The performance of the algorithm in the parallel pipelined environment is enhanced in comparison with the heuristic approach. The problem of assigning longer sub deadlines for tasks in the pipeline
is rectified in the metaheuristic approach.

II. MATERIALS AND METHODS

2.1 ALGORITHM

In the PCP scheduling algorithm, the critical path and partial critical paths of the whole workflow is to be found. In order to find these, some idealized, notion of the start time of each workflow task are needed before scheduling of the tasks are done. This means that two notions of the start times of tasks are available, the earliest start time computed before scheduling the workflow, and the actual start time computed by the scheduling algorithm. For each unscheduled task ti, the Earliest Start Time EST (ti) is found as the earliest time ti can start its computation regardless of the actual service that will process the task which is determined during scheduling. Since grid is a heterogeneous environment and the computation time of tasks varies from service to service the EST cannot be found exactly.

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

the Minimum Execution Time MET (ti) and the Minimum Transmission Time MTT (ei,j) is defined as follows

MET (ti) = min s∈Si ET (ti, s)

MTT (ei,j) = min s∈Si,r∈Sj T T (ei,j, s, r)

From the calculated MET and MTT, EST could be calculated

EST(tentry)=0

EST(ti)=max EST(tp)+MET(tp)+MTT(tp);

where tp belongs to ti's parents

The LFT(Latest Finishing Time) could be calculated as:

LFT(texit)=D

LFT(ti)=minLFT(tc)-MET(tc)-MTT(tc);

where tc belongs to ti's children

2.1.1. THE PCP SCHEDULING ALGORITHM

Two dummy nodes *tentry* and *texit* have been added to the task graph, even if the task graph already has only one entry or exit node.Algorithm I shows overall PCP algorithm for scheduling workflows.

Algorithm 1: The PCP Scheduling Algorithm

1: procedure SCHEDULEWORKFLOW(*G*(*T*,*E*), *deadline*)
2: request available services for each task in G from GMD
3: query available time slots for each service from related GSPs
4: add *tentry*, *texit* and their corresponding edges to G
5: compute *MET*(*ti*) for each task according to formula 1
6: compute *MTT*(*ei*,*j*) for each edge according to formula 2
7: compute *EST*(*ti*) for each task in G according to formula 3
8: mark *tentry* and *texit* as scheduled
9: set *AST*(*tentry*) ← 0,*AST*(*texit*) ← *deadline*
10: if (ScheduleParents(*texit*) is successfull) then
11: make advance reservations for all tasks in G according to Schedule
12: else
13: return (failure)
14: end if
15: end procedure

2.1.2. PARENT SCHEDULING ALGORITHM

This algorithm receives a scheduled node as input and tries to schedule all of its parents before the actual start time of the input node itself. On success, it returns the desired schedule, but on failure, it returns a task that causes this failure and a suggested start time for this task that hopefully makes its scheduling possible. The pseudocode of parent scheduling algorithm could be written as

Algorithm 2 Parents Scheduling Algorithm

1: procedure SCHEDULEPARENTS(*t*)
2: if (*t* has no unscheduled parent) then
3: return (Success)
4: end if
5: *ti* ← *t*
6: *CriticalPath* ← *empty*
7: while (there exists an unscheduled parent of *ti*) do
8: add *CriticalParent*(*ti*) to the beginning of *CriticalPath*
9: *ti* ← *CriticalParent*(*ti*)
10: end while
11: initialize *Constraints* to 0
12: while (*CriticalPath* is not scheduled) do
13: if (SchedulePath(CriticalPath, Constraints) is unsuccessful) then
14: set *tfailure* and *SuggestedStartTime* and return (Failure)
15: end if
16: for all (*ti* ∈*CriticalPath*) do
17: if (*ScheduleParents*(*ti*) is unsuccessful) then
18: if (*tfailure* ∈*CriticalPath*) then
19: *Constraints*[*tfailure*] ← *SuggestedStartTime*
20: break out from for loop
21: else
22: set *tfailure* and *SuggestedStartTime* and return (Failure)
23: end if
24: end if
25: end for
26: end while
27: return *ScheduleParents*(*t*)
28: end procedure

2.1.3. The Path Scheduling Algorithm

It starts from the first task in the path and moves forward to the last task, at each step selecting an untried available service for that task. If the selected service creates an admissible (partial) schedule, then it moves forward to the next task, otherwise it selects another untried service for that task. If there is no available untried service for that task left, then it backtracks to the previous task on the path and selects another service for it. After selecting a service for the current task *t*, say service *s*, the algorithm computes the start time *ST(t,s)* and the actual

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

cost *C(t,s)* of running task *t* on service *s*.The pseudocode for the path scheduling algorithm could be written as;

Algorithm 3 Path Scheduling Algorithm

1: procedure SCHEDULEPATH(*Path,Constraints*)
2: bestSchedule ← null
3: t ← first task on the path
4: while (t is not null) do
5: *s* ← next untried service ∈*St*
6: if (*s* = Ø) then
7: t ← previous task on the path and continue while loop
8: end if
9: Compute *ST*(*t, s*) and *C*(*t, s*)
10: if (*ST*(*t, s*) < *Constraintst*) then
11: *ST*(*t, s*) ← *Constraintst*
12: end if
13: for all (nodes *sc* ∈*Scheduled Children of t*) do
14: if (Actual Start Time of sc can not be met) then
15: continue while loop
16: end if
17: end for
18: if (t is the last task on the Path) then
19: if (this schedule has a better cost than bestSchedule) then
20: set this schedule as the bestSchedule
21: t ← previous task on the path
22: end if
23: else
24: t ← next task on the path
25: end if
26: end while
27: if (an admissible schedule found) then
28: mark all nodes of Path as scheduled
29: set AST(t) ← *ST*(*t, bestSchedulet*) for all tasks t in Path
30: update EST for all unscheduled children of all tasks in Path
31: return (Success)
32: else
33: determine *tfailure* and a suggested start time for it
34: return (Failure)
35: end if
36: end procedure

III. RESULTS AND DISCUSSIONS

As the first step of modifying the existing PCP algorithm, its heuristic nature is changed to metaheuristic one. Initially, only time constraint was considered for optimizing the cost of workflow scheduling. But this project resulted in the PCP algorithm which considers different parameters like storage, speed, and time etc for optimizing the cost of workflow scheduling. As first step of scheduling, prescheduling parameters of the tasks entered are found out.

A user interface was created to enable the users and service providers to enter new tasks, offer new services etc. Users could specify the requirements of the task like storage, speed etc. The prescheduling parameters like MET, EST, LFT etc are calculated.
For instance, user have entered five tasks namely t0, t1, t2, t3, t4. The MET of the tasks are 6,20,8,10,15 respectively. The EST of the first task are found using the MET as:
1.EST(t0)=0+0+6 =6
Likewise EST of the other tasks could be calculated.

LFT(t0)=70 which is the deadline likewise the LFT of the other tasks could be calculated using the formula. The EST and LFT of the tasks could be tabulated as:

Tasks	EST	LFT
t0	6	70
t1	22	37
t2	30	29
t3	40	19
t4	55	4

3.1SCREENSHOTS

Users and service providers are allowed to enter new tasks and its specifications, calculate the prescheduling parameters, and offer new services by logging into the wfms. After logging in as user, new task and its requirements could be specified as:

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

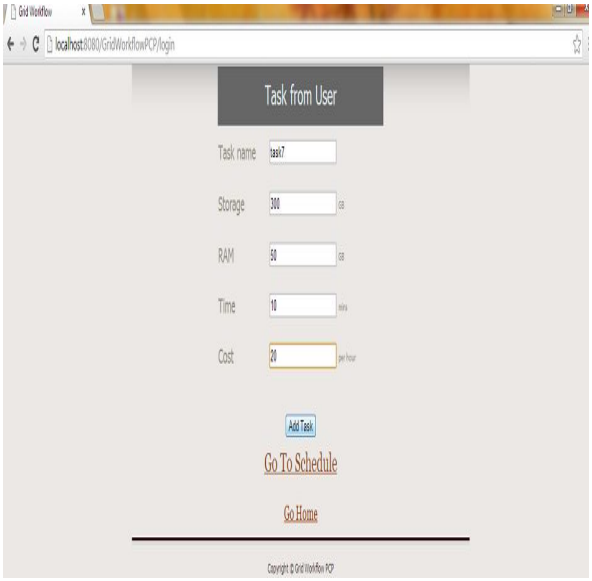
International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

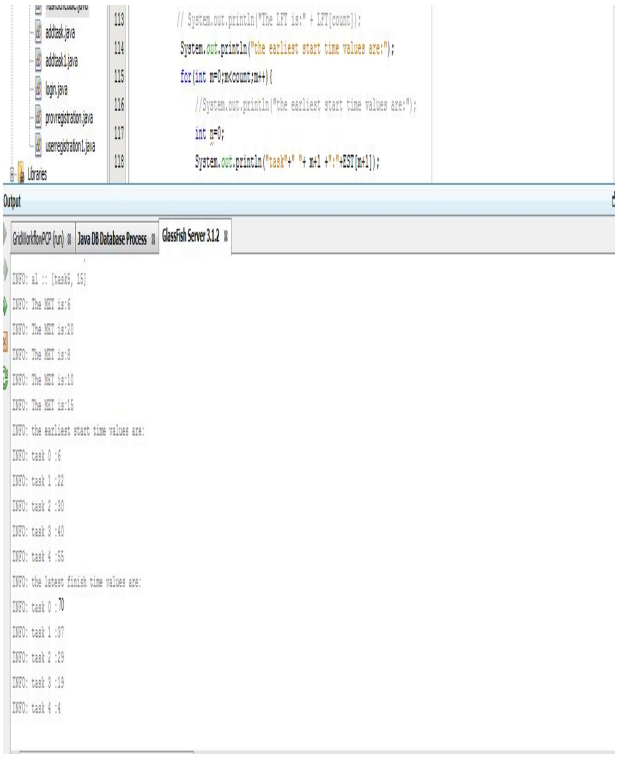
International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

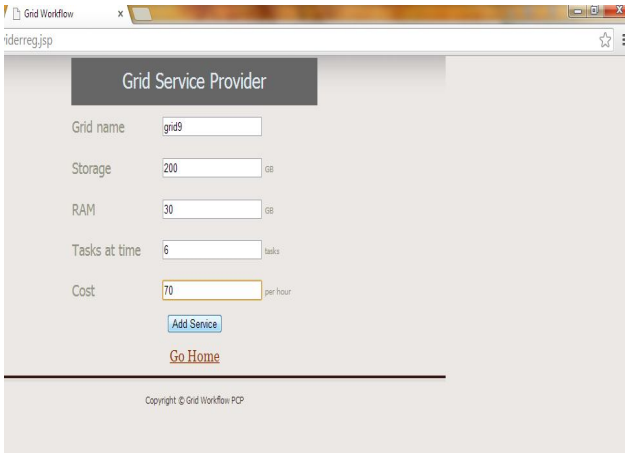
Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India



The prescheduling parameters like EST, LFT could be calculated from already defined formulas are found. The screenshot which shows the illustration of the prescheduling parameters could be listed as:



The service providers could offer new services by adding new services and its properties in the workflow management system. This could be displayed as:



IV. CONCLUSION

Utility Grids enable users to obtain their desired QoS (such as deadline) by paying an appropriate price. In this paper, a new algorithm named PCP is proposed for workflow scheduling in utility Grids that minimizes the total execution cost while meeting a user-defined deadline. The PCP algorithm has two phases: deadline distribution and planning. In the deadline distribution phase, the overall deadline of the workflow is divided over the workflow's tasks, for which are proposed three different policies, i.e., Optimized, Decrease Cost, and Fair. In the planning phase, the best service is selected for each task according to its subdeadline. The algorithm could be evaluated by simulating it with synthetic workflows that are based on real scientific workflows with different structures and different sizes.

The heuristic nature of the existing PCP algorithm is changed to metaheuristic nature. The PCP algorithm considered only time constraint for optimizing the cost of workflow scheduling. This nature of PCP algorithm is changed in such a way that it considers other requirements of users such that storage, speed, time etc while optimizing the cost of workflow scheduling.

4.1. FUTURE ENHANCEMENT

Multiple parallel pipelines operate on distinct chunks of data. At the beginning, when the PCP finds the critical path of the whole workflow, it obviously consists of the entry task, one of the parallel pipes, plus the exit tasks of

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

the workflow. Then, PCP tries to find the best schedule for this critical path, without considering the other parallel pipes. But if the other parallel pipes are considered, it is better to assign longer subdeadlines to these, because the other parallel pipelines also benefit from this extra time and the overall cost is reduced.

In the future, i plan to modify the algorithm to improve its performance on parallel pipelines. The performance is enhanced using pipeline aggressive copy method which minimizes the waiting time of tasks while scheduling.

REFERENCES

1. D. Laforenza, "European Strategies Towards Next Generation Grids," Proc. Fifth Int'l Symp. Parallel and Distributed Computing (ISPDC '06), p. 11, 2006.

2. J. Broberg, S. Venugopal, and R. Buyya, "Market-oriented Grids and Utility Computing: The State-of-the-Art and Future Directions," J. Grid Computing, vol. 6, no. 3, pp. 255-276, 2008.

3. J. Yu and R. Buyya, "Scheduling Scientific Workflow Applications with Deadline and Budget Constraints Using Genetic Algorithms," Scientific Programming, vol. 14, nos. 3/4, pp. 217- 230, 2006.

4. E. Deelman et al., "Pegasus: A Framework for Mapping ComplexScientific Workflows Onto Distributed Systems," Science Programming, vol. 13, pp. 219-237, 2005.

5. M. Wieczorek, R. Prodan, and T. Fahringer, "Scheduling ofScientificWorkflows in the Askalon Grid Environment," SIGMODRecord, vol. 34, pp. 56-62, 2005.

6. F. Berman et al., "New Grid Scheduling and ReschedulingMethods in the Grads Project," Int'l J. Parallel Programming, vol. 33, pp. 209-229, 2005.

7. J. Yu and R. Buyya, "A Taxonomy of Workflow ManagementSystems for Grid Computing," J. Grid Computing, vol. 3, pp. 171-200, 2005.

8. M.R. Garey and D.S. Johnson, Computers and Intractability: A Guideto the Theory of NP-Completeness. W.H. Freeman, Jan. 1979.

9. Y.K. Kwok and I. Ahmad, "Static Scheduling Algorithms forAllocating Directed Task Graphs to Multiprocessors," ACMComputing Surveys, vol. 31, no. 4, pp. 406-471, 1999.

10. H. Topcuoglu, S. Hariri, and M. Wu, "Performance-Effective andLow-Complexity Task Scheduling for Heterogeneous Computing," IEEE Trans. Parallel and Distributed Systems, vol. 13, no. 3, pp. 260-274, Mar. 2002.

11.Saeid Abrishami,Mahmoud Naghibzadeh and Dick H.J.Epema,"Cost Driven Scheduling of Grid Workflows using Partial critical paths", IEEE transactions on parallel and distributed systems, vol. 23, no. 8, August 2012.

12. Reen-Cheng Wang, Su-Ling Wu, and Ruay-Shiung Chang," A Novel Data Grid Coherence Protocol Using Pipeline-based Aggressive Copy Method"