Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Priority Based Workflow Handling Based on Grid Scheduling

Amit Chaudhary , Shivani Pahwa
Dept. of CSE
KUK, Kurukshetra , India

Grid allow us to solve huge and complex problems at a larger level, and kind of different computing technology in which the nodes are completely independent of each other in terms of resources. Grid computing is one of the modern tools to explain complex problems in many scientific and different applications, using the priority based resource sharing concept in the dynamic organizations. A priority-based workflow handling based on grid scheduling algorithm is discussed in this paper. The purpose of this research is to divide the assigned workflow into different tasks and after this assign priorities to these tasks .We defined the static values for workflow and create a static model which is used for workflow management. From the Results it can easily be shown that when we give workflow our algorithm provides the priorities assigned to each task. The priorities are assigned on the basis of three factors which are Average computation Cost, downlink Cost, uplink Cost .when user applies these three parameters as input and the output obtained as task order.

Keywords: Workflow Management, Directed Acyclic Graph, Load Balancing, Task Priority, cheduling

I. INTRODUCTION

Grid computing is important way to answer large-scale and complicate problems since Ian Foster proposed its concept. The term Grid generally mentioning to some form of system framework into which software or hardware components can be inserted, and which allows easy configuration and formation of new functionality from existing components. In grid computing different numbers of same types of computers clustered together. Grid can provide users with joined information and application services to realize resource sharing and teamwork in this real and in the virtual environment. Grid computing is the collection of different resources from different places to reach a common goal. The grid can be supposed of as a distributed system with interactive and non-interactive workloads that involve a large number of files

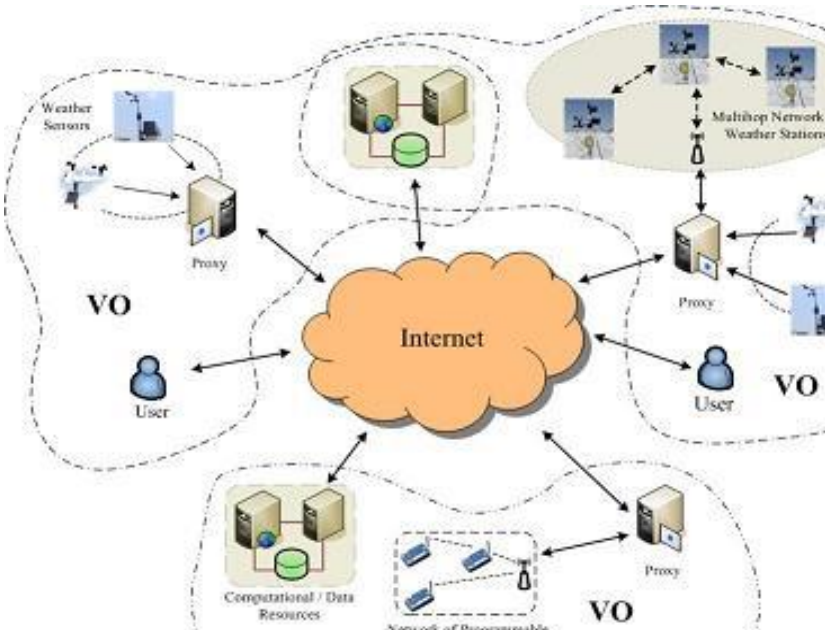

Fig 1.1: Grid Architecture

Grids are a form of distributed computing whereby a "super virtual computer" is composed of many networked loosely coupled computers acting together to perform large tasks. Load Balancing [5] is critical to computational grids. It is a planning strategy that capably equilibrates the task capacity into computational resources in the network based on the system status to increase the system performance.

A. **Grid Workflow Scheduling**

Grid workflows are an emerging research field in the Grid community. The workflow [2] is said as a set of tasks With dependencies between them. A task becomes executable when its dependencies is seen and may be scheduled by submitting it to a resource for execution. The workflow scheduling design includes many issues involved in the workflow at built time.

A workflow scheduling priority based can be composed by connecting different tasks according to their dependencies. Workflow structure, also known as workflow pattern, shows the temporal link between tasks. The workflow can be represented as Directed acyclic graph and non-direct cyclic graph. In DAG-based workflow, workflow scheduling structure can be considered into sequence, parallelism, and choice. Sequence is defined as an ordered structure of jobs, with one task starting after a recent task has been completed. Parallelism represents jobs which are performed one after another, rather than serially. In choice control pattern, a task is chosen to execute at run-time when its linked conditions is true.

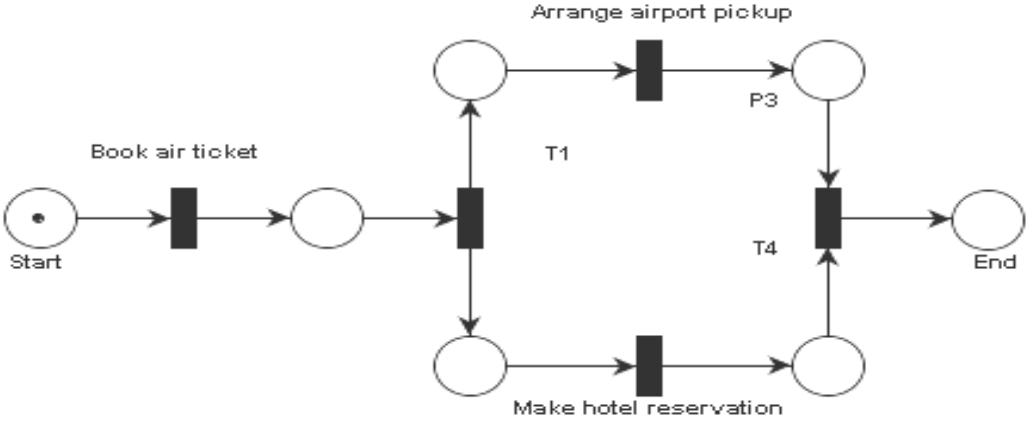

Figure 1.2 Practical model of priority workflow scheduling

In calculation to all patterns contained in a DAG-based workflow model, non-DAG workflow also contains repetition structure, in which portions of workflow tasks in iteration group is allowed to be continual. Iteration structure is fairly common in many applications, where some tasks are needed to be executed repeatedly. These types of workflow priority based models, which are sequence, parallelism, choice and iteration can be used to build many difficult workflows. Moreover, sub-workflows can also use these types of workflow structure as building blocks to form a large-scale workflow. A full or fractional concrete model can be produced just before or during workflow execution according to the current position of resources. Additionally, in some systems, every job in the workflow is concretized only at the time of task execution. Though, concrete models may be required by some end users who want to switch the execution sequence

B. **Directed Cyclic Graph (DAG)**

Direct cyclic graph mainly used to show the data dependency between subtasks, and it can be used to answer priority scheduling Problems.Essentially, the most common Grid workflow can be modelled as simple Task Directed Acyclic Graph where the direction of execution of tasks is resolute by dependencies Each direct cyclic graph node represents the processing of a component, described by a set of qualities such as an estimation of its cost and possible necessities on the target execution platform, while DAG edges represent data dependencies among specific application components.

C. **Workflow Scheduling**

Task based priority scheduling [4] in distributed computing environment into local task arrangement and global task scheduling. Local scheduling includes hold the assignment of jobs to time-slices of a single resource whereas global scheduling involves deciding where to execute a task

II. TASK BASED RIORITY SCHEDULED PROBLEM

A Problem is distributed into different tasks according to their needs and can be characterized with the help of Direct cyclic graph. A DAG is represented by figures such as G (V, E, P, T, C) where

a) Communication cost linked with edges: C
b) Number of jobs: *V* and the calculation cost matrix of the DAG: *T(using variable v*v)*
c) Number of processors in the organizations: *P*
d) Quantity of data to be relocated between the tasks: *D (v×v)*
e) Communication nodes from one task to some other task. $E(v_a,v_b)$

The model proposed here has two phases, firstly level sorting secondly Task prioritization phase .In the level sorting phase, the given DAG is traversed in a top-down fashion to sort tasks at each level in order to group the tasks that are

independent of each other. As a result, tasks in the same level can be executed in parallel. Given a DAG G = (V, E), level 0 contain entry tasks. Level I consist of all tasks vk such that for all edges (v_y, v_z), task v_y is in a level less than i and there exists at least one edge (v_y,v_z) such that vj is in first level whereas the other level comprises some of the terminating tasks. For implementation, it is assumed that there is one starting task and one terminating task for a DAG. If there are many such type of entry or exit tasks, the different tasks can be linked to a dummy task with zero computation cost and zero communication cost edges. In the task prioritization stage, priority is observed and allocated to every task of the task graph. There are many aspects used to assign the priority to a task such as Down Link Cost (DLC), the Up Link Cost (ULC), the Link Cost (LC) and the Average Computation Cost (ACC) of the task. The DLC of a task is the maximum data (input) received by a task from all its instantaneous predecessor tasks. The down link cost for all types of tasks at initial level is zero and for all other tasks at level l, the down link cost of a task v_y is computed using thee equation 1.

DLC(va) = Max{Vol(v_a,v_b)}, where v_a prefixes (v_b) . (1)

The uplink cost of a task is the maximum quantity to be shifted from a task to all its instant successors. The ULC for exit task is 0 and for all other tasks at level l, it is calculated using the equation no 2.

ULC (v_b) = Max {Vol (v_a, v_b)}, where v_b suffixes (v_y).

The LC or link cost of a task is the sum of down link cost ,up link cost and maximum link cost of its immediate predecessor tasks. The LC of a task can be find using the following equation

LC(v_b) = 0, for entry task, otherwise = max{LC(v_z)}+ULC(v_b)+DLC(v_b), for all v_b prefixes (v_a). (3)

The Average Computation Cost (ACC) of any random task such as v_i is the mean value of computation cost on all the P existing processors and it is computed using the Equation (4) given below.

ACC (vi) = ∑T (v_x, pj) (4)

Then the priority is assigned to entirely the tasks at each level i, based on its LC rate. At all levels, the task with the maximum link cost value accepts the uppermost priority followed by the task with next highest LC value and so on in the identical level. Although assigning priority, if two tasks having the similar LC value, then the link is ruined based on the ACC value. The task with maximum ACC value obtains higher priority as compared to the task with the lower ACC value

Proposed Algorithm

1. S t a rt
2. Read the DAG, related aspects values, and the no. of different processors P;
3. Sort out the DAG(Level);
4. for all task V_i in the DAG do
5. Begin
6. Find DLC, ULC and ACC values for the available Task V_i ;
7. Find DLC(v_i) = Max{Data(v_a,v_b)}, where v_a prefixes v_b .
8. Find ULC (v_i) = Max {Data(v_a, v_b)where v_a suffixes vb
9. Compute LC(v_i) = max{LC(v_b)} + ULC(v_z) + DLC(v_z),
10. Placed the task into the priority queue centered on the LC values such that the tasks in Lower level is located in the priority queue than the tasks in the higher level and link if any, is shattered using the ACC value.
11. End

The figure 2.1 represents the scheduling the workflow for assigning tasks with the help of DAG. A Priority based dynamic Model is proposed to agree with the given problem. As shown in figure below name as 2.1, it contains total

ten random tasks as V1 to V10 as entry and exit task respectively. After applying the above algorithm it sort out the priority to task at all levels.

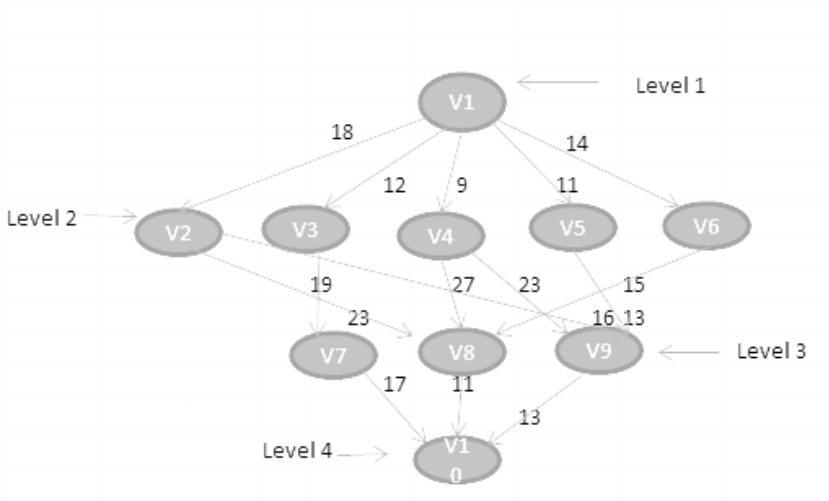

Fig 2.1 Workflow scheduling representation with help of DAG

Table 2.1 DAG REPRESENTATION

Nodes	suffixes	Prefixes
V1	V2,V3,V4,V5,V6	None
V2	V8,V9	V1
V3	V7	V1
V4	V8	V1
V5	V9	V1
V6	V8	V1
V7	V10	V3
V8	V10	V4
V9	V10	V5
V10	None	V7,V8,V9

Table 2.2 SORTED TASK LIST BY LEVEL

level	nodes	DLC	ULC	LC	ACC	Priority order
1	V1	0	18	18	13	1
2	V2	18	19	55	16	1
2	V3	12	23	53	14	3
2	V4	9	27	54	12	2
2	V5	11	13	42	11	5
2	V6	14	15	47	16	4
3	V7	23	17	93	11	1
3	V8	27	11	93	10	2
3	V9	23	13	91	16	3
4	V10	17	0	110	14	1

Table 2.3: SORTED TASK LIST BY LEVEL AND PRIORITY

Level	nodes	DLC	ULC	LC	ACC	Priority level
1	V1	0	18	18	13	1
2	V2	18	19	55	16	1
2	V4	9	27	54	12	2
2	V3	12	23	53	14	3
2	V6	14	15	47	16	4

2	V5	11	13	42	11	5
3	V7	23	17	93	11	1
3	V8	27	11	93	10	2
3	V9	23	13	91	16	3
4	V10	17	0	110	14	1

Table above 2.2 & 2.3 shows Sorted task List by level and then by priority .The priority at each level of DAG can be found on bases on parameters such as DLC, ULC & LC. As shown in table above , task sorted by leve l then assign priority at each level so now order of execution of workflow will be priority based.

III. Conclusions

In this paper, we implements t a s k scheduling workflow management based on priority over a grid network. Actually, workflow is difficult to be distributed in tasks in case of grid computing as the size of task to be execute is not known a priori and the selection principles is also a type of bottleneck problem . Also the assignment of priorities to different jobs is a matter as the level of each task is not known and also to distinguish among entry and exit task is difficult. So, Workflow has been distributed into different jobs and priorities are assigned to these jobs. After applying the algorithm the results shows that when we input workflow, output obtain as the priorities assigned to each job. These priorities are allocated on the basis of different costs such as Uplink Cost, Downlink Cost and Average computation Cost.After applying these three parameters as input and the output comes as task order to specify the task and assign the priorites.

Refrences

1. Ian Foster. (2002) "*what is the grid*"? A three point checklist, Grid Today, vol.1, pp.6-12, 2002
2. Bharadwaj, V., Ghose , D. , G. Robertazzi, T.(2003) *"Divisible Load Theory: A New Paradigm for Load Scheduling in Distributed Systems"* Springer Berlin Heidelberg. January
3. Talukder, A. K. and Yavagal , R.R. (2006) "*Mobile Computing: Technology, applications and service creation*", Tata McGraw Hill.
4. Ehsan Ullan Munir and Jian-Zhong Li (2007) *"Performance Analysis of Task Scheduling Heuristics in grid."* Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 19- 22 August.
5. Gizem, Aksahya & Ayese, Ozcan (2009) "*Communications & Networks"*, Network Books, ABC Publishers.
6. Suri, P.K. , and Singh, M . (2010)" *An Efficient Decentralized Load Balancing Algorithm for Grid*," IEEE 2nd International Advance Computing Conference, March.
7. Sharma,R, , Kant Soni ,V,; Mishra,M.K (2010)" *A New Resource Scheduling Model with Bandwidth aware Job Grouping Strategy in Grid Computing systems*," Computer Science and Information Technology (ICCSIT), 3rd IEEE International Conference
8. Pandey,S.,Gupta,K., Barker,A. (2011)" *Minimizing Execution Cost when using Globally Distributed Cloud Services*" 24th IEEE International Conference on Advanced Information Networking and Applications (AINA),Australia
9. Feng,G., Garg,S.K. , Buyya, R. ,(2012)" *Revenue Maximization Using Adaptive Resource Provisioning in Cloud Computing Environments*", Proceedings of the 13th IEEE/ACM International Conference on Grid Computing
10. Marton, I, Karoczkal , K. , Kozlovzsky. , M.(2012) "*Enabling Generic Distributed Computing Infrastructure Compatibility for Workflow Management"* International Journal of Computer Science.
11. Yadav,K., Jindal, D. and Singh, R.(2013) "*Job Scheduling in Grid Computing*" *International Journal of Computer Applications"* 69(22):13-16, May.
12. Long, W., Rubing, D., Xiaorong Li,(2013)" *An Iterative Optimization Framework for Adaptive Workflow Management in Computational Clouds"*, Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications

