IMPROVED MAX-MIN HEURISTIC MODEL FOR TASK SCHEDULING IN CLOUD

S.DEVIPRIYA
PG Scholar/CSE
Bannari Amman Institute of Technology
Sathyamangalam, India
Email:priyacse01@gmail.com

C.RAMESH
Asst Prof(Sr.Grade)/CSE
Bannari Amman Institute of Technology
Sathyamangalam, India
Email:rameshc70707@gmail.com

Abstract

Cloud computing is the use of computing resources that are delivered as a service over a network. It supplies a high performance computing based on protocols which allows shared computation and storage over long distances. In cloud computing, many tasks need to execute at a time by the available resources in order to achieve better performance, minimum completion time, shortest response time, resource utilization etc [4]. Because of these different factors, we need to design, develop, and propose a scheduling algorithm for the proper allocation of tasks to the resources. In this paper, a simple modification of Max-min algorithm is proposed. This algorithm is built based on RASA algorithm and the concept of Max-min strategy. An Improved Max-min algorithm is developed to outperform scheduling process of RASA in case of total complete time for all submitted jobs. Proposed Max-min algorithm is based on expected execution time instead of complete time. So the scheduling tasks within cloud environment using Improved Max-min can achieve lower makespan rather than original Max-min.

Keywords

Cloud computing,Max-min Algorithm, Min-min ,RASA, Algorithm, makespan.

1 INTRODUCTION

Cloud computing relies on sharing of resources to achieve coherence and economies of scale similar to a utility over a network. Cloud Computing is getting popular every day. Cloud service providers provide services to large scale cloud environment with cost benefits. Also, there are some popular large scaled applications like social networking and internet commerce. These applications can provide benefit in terms of minimizing the costs using cloud computing. Cloud computing is considered as internet based computing service provided by various infrastructure providers based on their need, so that cloud is subject to Quality of Service(QoS), Load Balance(LB) and other factors which have direct effect on user consumption of resources controlled by cloud infrastructure .In Cloud scheduling process need to achieve several factors. So it needs to use the effective algorithm for allocating proper task to the proper resources.Various task scheduling algorithms has been proposed, most important task scheduling algorithms are Min-min, Max-min, RASA, etc.

2 SCHEDULING PROCESS IN CLOUD

The main advantage of job scheduling algorithm is to achieve a high performance computing and the best system throughput. The available resources should be utilized efficiently without affecting the service parameters of cloud. Scheduling process in cloud can be categorized into three stages they are Resource discovering and filtering, Resource selection, and Task submission [10]. In resource discovery datacenter broker discovers the resources present in the network system and collects status information related to them. During resource selection process target resource is selected based on certain parameters of task and resource. Then during task submission task is submitted to the selected resource.

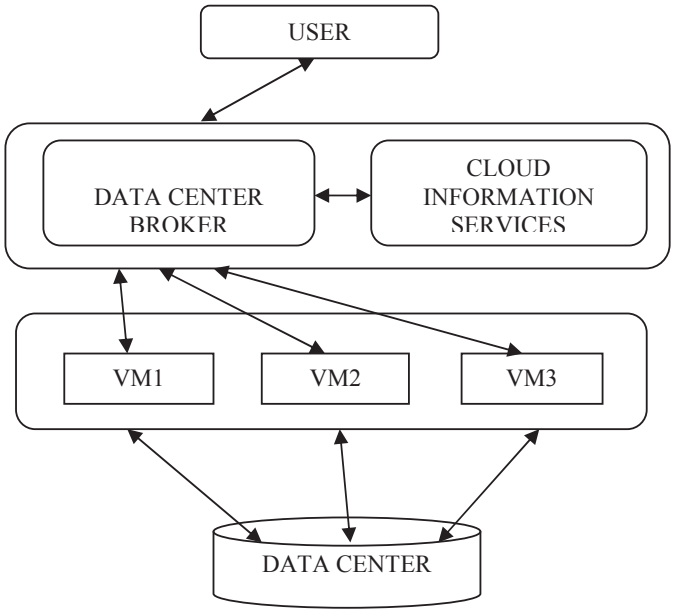

Fig 1. Scheduling in Cloud

3 EXISTING HEURISTIC ALGORITHMS FOR TASK SCHEDULING

Heuristics are applied to find an optimal solution to the scheduling. Heuristic can be both static and dynamic. Static heuristic is applied when the number of tasks to be completed are known in prior. Dynamic heuristic can be used when the task arrival is dynamic in nature. This paper focus on heuristic algorithms such as Min-min, Max-min and Improved Max-min [7].

3.1 MIN-MIN ALGORITHM

Min-min scheduling is based on Minimum Completion Time (MCT) that is used to assign tasks to the resources having minimum expected completion time. It will work in two phases, in the first phase, the expected completion time will be calculated for each task in a metatask list. During the second phase, the task with the overall minimum expected completion time from metatask list is chosen and assigned to the corresponding machine [1]. Then this task is removed from metatask list and the process is repeated until all tasks in the metatask list are mapped to the corresponding resources However, the Min- min algorithm is unable to balance the load well as it usually does the scheduling of small tasks initially. [8].

1. for all submitted tasks in meta-task Ti
2. for all resource Rj
3. compute Cij = Eij + rj
4. While meta-task is not empty
5. find the task Tk consumes minimum completion time.
6. assign task Tk to the resource Rj with minimum execution time.
7. remove the task Tk from meta-tasks set
8. update rj for selected Rj
9. update Cij for all i

Fig 2. Min-min algorithm

In Fig 2 rj represents the ready time of the resource Rj to execute a task, Cij and Eij represent the expected completion time and execution time of the tasks.

3.2 MAX-MIN ALGORITHM

Similar to Min-min, a scheduler schedules tasks by expecting the Execution Time of the tasks and allocation of resources. Instead of selecting the minimum MCT, the maximum MCT is selected, that is why it is named Max-min. It focuses on giving priority to large tasks over others small. The Max-min algorithm is typical to the Min-min algorithm, except for being different in; the word "minimum" would be replaced by "maximum" [9].Officially Max-min algorithm does better than Min-min algorithm in cases when the number of short task is more than the longer ones. For example, if there is only one long task, the Max-min algorithm executes many short tasks concurrently with the long one. In order to avoid the main drawbacks of the Max-min and Min-min, the two schedulers can be executed alternatively to each other for assigning tasks to appropriate resources, for eliminating each other drawback. Such methodology, called Resource Awareness Scheduling Algorithm (RASA) which was a new grid task scheduling algorithm

1.for all submitted tasks in meta-task Ti
2. for all resource Rj
3. compute Cij = Eij + rj
4. While meta-task is not empty
5. find the task Tk consumes maximum completion time.
6. assign task Tk to the resource Rj with minimum execution time.
7. remove the task Tk from meta-tasks set
8. update rj for selected Rj
9. update Cij for all i

Fig 3.Max-min algorithm

In Fig 3 algorithm, the expected time of resource R_j is the time to become ready to execute a task after finishing the execution of all tasks assigned to it which is denoted by r_j. Also E_{ij} is the estimated execution time of task T_i on resource R_j whereas C_{ij} is the Expected Completion Time that is the estimated execution time and ready time together.

3.3 RASA ALGORITHM

RASA is a hybrid algorithm of two other ones such as Min-min and Max-min. In RASA, an estimation of the completion time of each task on the available resources is calculated then Max-min and Min-min algorithms are applied alternatively to take advantage of both algorithms and avoids their drawbacks. In RASA the Max-min and Min-min scheduling algorithms are applied alternatively [5]. For example, if the first task is assigned to a resource by Min-min strategy, in the next round the task will be assigned by Min-min and so on. Based on experimental results, if the number of available resources in grid system is odd it is highly preferred to start by Min-min algorithm in first round otherwise it will be started by Max-min algorithm. For next rounds just assign resources to task

using a strategy different from last round it reduce the delays in the execution of small tasks by Max-min algorithm and large tasks by Min-min algorithm.

1. for all tasks T_i in meta-task
2. for all resources Rj
3. Compute $Cij = E_{ij} + r_j$
4. do until all tasks in metatask are mapped to the resource
5. if the number of resources is even then
6. for each task in metatask find the expected completion time and the resource that make it.
7. find the task which gives the maximum expected completion time.
8. assign that task to the faster resource to get minimum completion time.
9. delete the task from metatask.
10. update readytime rj
11. update completion time C_{ij} for all task i.
12. else
13. for each task in metatask find the expected completion time and the resource that make it.
14. find the task which gives the minimum expected completion time.
15. assign that task to the faster resource to get minimum completion time.
16. delete task from metatask .
17. update readytime rj
18. update completion time C_{ij} for all i.
19. end do.

Fig 4.RASA algorithm

4 PROPOSED HEURISTIC ALGORITHM FOR TASK SCHEDULING

Max-min algorithm allocates larger task Ti to the resource Rj where large tasks have highest priority rather than smaller tasks [9]. For example, in Max-min we can execute many short tasks concurrently while executing the larger one. The total makespan, in this case is determined by the execution of longer task. But if the metatasks contains tasks with the different completion time then the overall task completion time is not determined by one of the submitted tasks. It would be similar to the Min-min makespan.For these cases, original Max-min algorithm losses some of its major advantages as load balance between available resources in small distributed system configuration and small total completion time for all submitted tasks in large scale distributed environment. We can't use the Max-min to decide what would be the allocation map, makespan, load balance, etc. We try to minimize the waiting time of shortest jobs by assigning largest tasks to slower resources for execution. Smallest tasks are executed by fastest resources so we can execute multiple smallest tasks while executing a single largest task by the slowest resource. We proposed an improvement of Max-min algorithm that leads to increase in Max-min algorithm efficiency. Proposed improvement increases the opportunity of concurrent execution of tasks on resources. We focuses on the Max-min to derive improved Max-min because of its advantages as load balance that is desired in small distributed system rather than larger and small makespan in large distributed system rather than small.

4.1 IMPROVED MAX-MIN ALGORITHM

The basic idea of an improved version of Max-min assigns task with maximum execution time to the resource which produce minimum completion time rather than original Max-min assigns task with maximum completion time to the resource which provides minimum execution time. It means it select largest task then assign it to slowest resource.

1. For all submitted tasks in Meta-task Ti
    - 1.1. For all resources; Rj
    - 1.2. Cij = Eij + rj
2. Find task Tk costs maximum execution time (Largest Task).
3. Assign task Tk to resource Rj which gives minimum completion time (Slowest resource).
4. Remove task Tk from Meta-tasks set.
5. Update rj for selected Rj.
6. Update Cij for all j.
7. While Meta-task not Empty
    - 7.1. Find task Tk costs maximum completion time.
    - 7.2. Assign task Tk to resource Rj which gives minimum execution time (Faster Resource).
    - 7.3. Remove Task Tk form Meta-tasks set.
    - 7.4. Update rj for Selected Rj.
    - 7.5. Update Cij for all j.

Fig 5.Improved Max-min algorithm

Where T represent task, Cij represent completion time of task

T with resource j, rj represent resource, Rj represent the ready time, Eij represent execution time of task I with resource j.

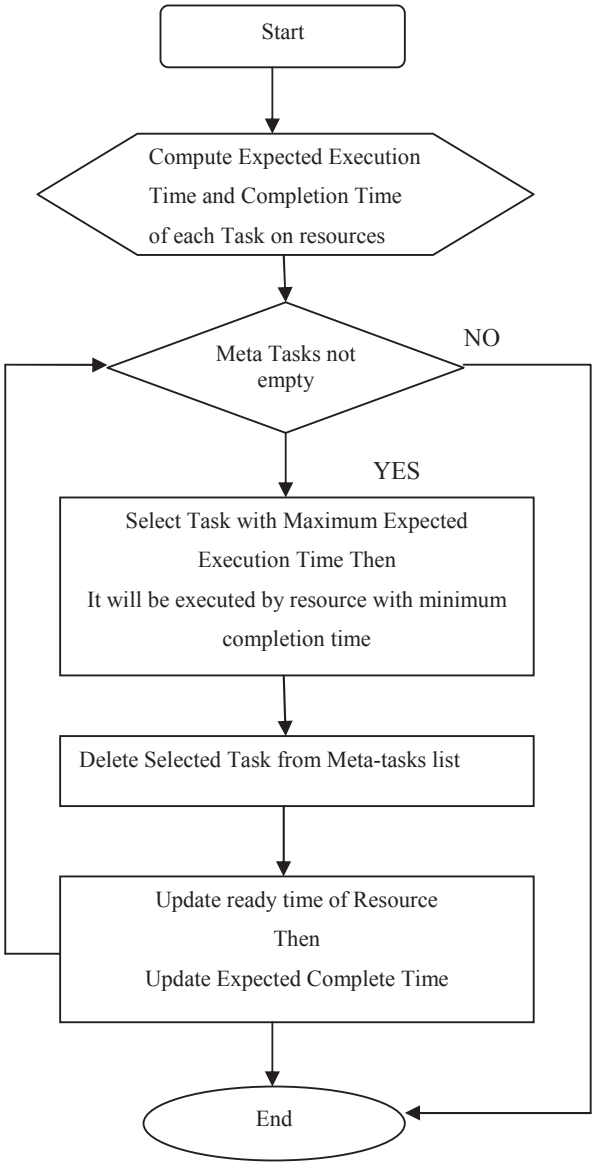

Fig 6.Flowchart

The algorithm in Fig 5 calculates the expected completion time of the submitted tasks on each resource. Then the task with the overall maximum expected execution time is assigned to a resource that has the minimum overall completion time. Finally, this scheduled task is removed from meta-tasks and all calculated times are updated and then the Max-min algorithm is applied for the remaining tasks. By selecting the task with maximum execution time means selecting larger tasks. While selecting the resource which consume minimum completion time means selecting the slowest resource among the available resources. So by allocating slowest resource to the longest task allows availability of high speed resources for finishing other small tasks concurrently. Also, we achieve shortest makespan of submitted tasks on available resources beside concurrency.

5. THEORETICAL ANALYSIS

In the theoretical analysis of improved Max-min, task selection is given below. Assume that Task scheduler has meta-tasks and resources as given below. Table 1, represents the volume of instructions and data for each task T1 to T4.

Table 1 Meta-Task Specification

Problem Sample	Task	Instruction Volume (MI)	Data Volume (MB)
P1	T1	128	44
	T2	69	62
	T3	218	94
	T4	21	59
P2	T1	256	88
	T2	35	31
	T3	327	96
	T4	210	590
P3	T1	20	88
	T2	350	31
	T3	207	100
	T4	21	50

Table 2 Resource Specification

Problem Sample	Resource	MIPS	MBBS
P1	R1	50	100
	R2	100	5
P2	R1	150	300
	R2	300	15
P3	R1	300	300
	R2	30	15

Table 1 represent metatask specification and Table 2 represent resource specification By using the data's given in table 1 and table 2 the makespan of all the tasks in all available resources will be calculated.

Table 3 Makespan of problem samples using algorithms

Problem Samples	Min-min	Max-min	RASA	Imp Max-min
P1	11	11	10	10
P2	9	9	9	8
P3	5	5	5	4

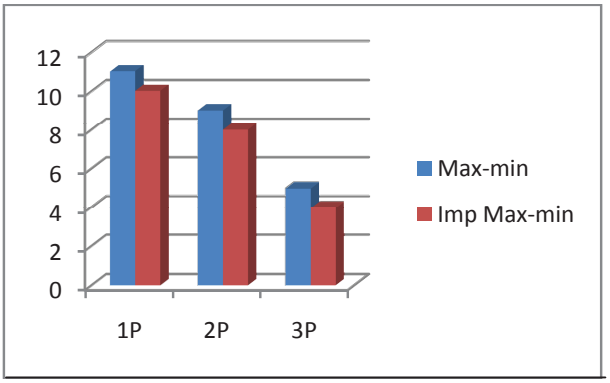

Fig 7.Comparison of Makespan

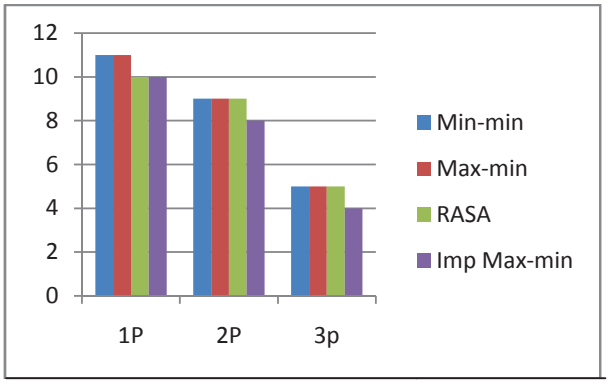

Fig 8.Comparison of Makespan

6. CONCLUSION AND FUTURE WORKS

Min-min and Max-min algorithms are most popular algorithms used in small scale distributed systems. When the number of smaller tasks is higher than number of the larger tasks in a meta-task, in that cases the Max-min algorithm works well because Min-min algorithm uses a single resource so we can't execute tasks concurrently it will increase the makespan in this cases. To overcome such limitations of Min-min algorithm, Max-min algorithm has been proposed in this we can able to execute the tasks concurrently, but if the number of larger tasks is high it will increase the completion time .So a new modification is applied for Max-min scheduling algorithm. Here instead of applying larger tasks to the faster resource in improved Max-min larger tasks are applied to slower resources and smaller tasks are executed by faster resources in order to reduce the makespan . This study is only concerned with the number of the resources and the tasks. The study can be further extended by applying the proposed algorithm on actual cloud computing environment and considering many other factors such as scalability, availability, stability and others. Also, in future we can improve the presented algorithm to be optimized and produce more efficient makespan by using one of the heuristics algorithm such as genetic algorithm (ga) and genetic programming (gp), etc.

7. REFERENCES

[1] X. He, X-He Sun and G.V. Laszewski, "QoS Guided Min-min Heuristic for Grid Task Scheduling," Journal of Computer Science & Technology, vol. 18, pp. 442-451, 2003.

[2] F. Dong F, J. Luo, L. Gao and L. Ge, "A Grid Task Scheduling Algorithm Based on QoS Priority Grouping," In the Proceedings of the Fifth International Conference on Grid and Cooperative Computing (GCC'06), IEEE,2006.

[3] E. Ullah Munir, J. Li and Sh Shi, "QoS Sufferage Heuristic for Independent Task Scheduling in Grid," Information Technology Journal, vol. 6, no. 8, pp. 1166-1170, 2007.

[4] L. Mohammad Khanli and M. Analoui, "Grid_JQA: A QoS Guided Scheduling Algorithm for Grid Computing," The Sixth International Symposium on Parallel and Distributed Computing(ISPDC'07), IEEE,2007.

[5] Saeed Parsa and Reza Entezari-Maleki, " RASA: A New Task Scheduling Algorithm in Grid Environment" in World Applied Sciences Journal 7 (Special Issue of Computer & IT): 152-160, 2009.

[6.] Salim Bitam, "Bees Life algorithms for job scheduling in cloud computing", International Conference on computing and Information Technology, 2012.

[7.] Braun, T.D., Siegel, H.J., Beck, N., Boloni, L.L.,Maheswaran, M., Reuther, A.I., Robertson, J.P., et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous

distributed computing systems", Journal of Parallel and Distributed Computing, Vol. 61, No. 6, pp.810–837, 2001.

[8.] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Minmin Heuristic for Grid Task Scheduling," Journal of Computer Science and Technology, Vol. 18, pp. 442-451, 2003.

[9.] Etminani .K, and Naghibzadeh. M, "A Min-min Max-min Selective Algorithm for Grid Task Scheduling," TheThird IEEE/IFIP International Conference on Internet, Uzbekistan, 2007.

[10] Shu-Ching Wang, Kuo-Qin Yan, Shun-Sheng Wang, Ching-Wei Chen, "A Three-Phases Scheduling in a Hierarchical Cloud Computing Network", IEEE, 2011