COMPARATIVE ANALYSIS OF TASK SCHEDULING ALGORITHMS IN CLOUD ENVIRONMENT

D. Gupta*, Harmaninder Jit Singh Sidhu, G. Kumar

1* CSE, Desh Bhagat University, Mandi Gobindgarh, Punjab, India
2 CSE, Desh Bhagat University, Mandi Gobindgarh, Punjab, India
3 IT, DAVIET, Jalandhar, Punjab, India

ABSTRACT:

Cloud computing is fast emerging as hottest high performance & distributed computing concept as it provides on-demand access to shared pool of resources over Internet in a self service, dynamically scalable and metered manner. Scheduling in cloud computing belongs to a category of problems known as NP-hard problem due to large solution space and thus it takes a long time to find an optimal solution. There are no algorithms which may produce optimal solution within polynomial time to solve these problems. In this paper, three algorithms have been implemented to schedule the incoming jobs. Two of the implemented algorithms are static algorithm and one is a stochastic meta-heuristic algorithm.

Keywords: ACO, VM, FCFS, SJFS, Data Centre, Cloud Computing, HA, MHA, SIA.

[1] INTRODUCTION

Cloud computing is still in its evolving stage. One of the important research area in cloud computing is scheduling of incoming jobs. The goal of scheduling is to map tasks to appropriate resources that optimize one or more objectives. Scheduling in cloud computing due to its large solution space belongs to a category of problems known as NP-hard problem. Objective of task scheduling in cloud computing is to find suboptimal solution in less time.
The organization of paper is as following: Section 2 Introduction to the related work. Section 3 Is the Methodology. Section 4 discusses the implementation of both static and dynamic algorithms. Results can also be seen in this section. Section 5 gives conclusion of this paper.

**Cloud Scheduling:** Scheduling problem involves tasks that must be scheduled on resources subject to some constraints to optimize some objective function. Scheduling allows optimal allocation of resources among given tasks in a finite time to achieve desired quality of service. The aim is to build a schedule that specifies when and on which resource each task will be executed.[7]

**Scheduling architecture:** There are number of users submit their tasks to Data Centre Broker but the Broker behaves like a dispatcher between user and Data Center and helps to schedule task on virtual machines. In Data Centre there is number of host on which number of virtual machines are scheduled and on those VM task are scheduled according to the scheduling polices taken by Data Centre Broker. Data Center Broker communicate with cloud controller and schedule the submitted tasks.

![Cloud scheduling architecture](image)

**Fig-1: Cloud scheduling architecture [8]**

**Optimization Problem:** In optimization problems, we have to find solutions which are optimal or near-optimal with respect to some goals. Probably it is not possible to find the optimal solution for scheduling problem is cloud. Hence, the idea is to find is to find near-to-optimal solution to the scheduling problem based on the objective function in hand.

**Classification of optimization Algorithms:** Many methods have emerged for the solution of optimization problems which can be divided into two categories based on the produced solutions namely deterministic and nondeterministic (stochastic) algorithms:[9]

**Deterministic algorithms:** Deterministic algorithms in general follow more rigorous procedures repeating the same path every time and providing the same solution in different runs.
Non-deterministic or stochastic: Non-deterministic or stochastic methods exhibit some randomness and produce different solutions in different runs. The advantage is that these methods explore several regions of the search space at the same time and have the ability to escape from local optima and reach the global optimum.

Fig-2: Classification of optimization algorithm [6]

Stochastic optimization algorithms are of two types:- Heuristic algorithms (HA) and meta-heuristic algorithms (MHA).

[2] RELATED WORK

Bio inspired algorithm plays very important role in computer networks, data mining, power system, economics, robotics, information security, control system, image processing etc. There are great opportunities of exploring or enhancing these algorithms with the help of innovative ideas or thoughts [3]. ACO performs better than scheduling algorithms like First Come First Served (FCFS) and Round-Robin (RR) in terms of Makespan [4]. ACO algorithm for scheduling tasks in cloud provides fair scheduling as compared to static algorithm like algorithm First Come First Serve (FCFS) [5]. Metaheuristic techniques are usually slower than deterministic algorithms and the generated solutions may not be optimal, most of the research done is towards improving the convergence speed and quality of the solution [1]. Cloud computing has the ability to deliver high-end computing capabilities using data centre with the help of computing infrastructure. Energy efficient algorithms are an effective approach to resolve this complication in a very effective manner [2].

[3] METHODOLOGY

Three algorithms have been implemented to schedule the incoming jobs. Two of the implemented algorithms are static algorithm and one is a stochastic meta-heuristic algorithm.
3.1 Static Algorithms

3.1.1 First Come First Serve

The Cloudlets are mapped to the virtual machine (VM) on the basis of this arrival time. No other parameter is considered for scheduling the incoming jobs. It is the simplest algorithm to implement. Following is the pseudo code for implementing the first come first serve:

Let m be the number of VM’s
Let n be the number of cloudlets
Let cloudletlist be list of all cloudlets. Size of cloudletlist is n
For each cloudlet
    Assign one cloudlet at a time to a VM based on modulus, i.e. n%m.
return the cloudletList; // this list will be sent to broker for execution.

3.1.2 Shortest Job First

The Cloudlets are sorted on the basis of their length in ascending order and then will be mapped to virtual machines (VM) just like. Following is the pseudo code for SJF:

Let m be the number of VM’s
Let n be the number of cloudlets
Let cloudletlist be list of all cloudlets. Size of cloudletlist is n
The templist will be used as temporary cloudlist for sorting the cloudlets and sortlist will contain the cloudlets in the sorted order.
For each cloudlet in the cloudletlist
    Add cloudlet to the templist
Sort the templist and save the sorted list in sortlist
Send the sorted cloudlist, i.e. sortlist to broker for execution

3.2 Metaheuristic Algorithm
3.2.1 Ant Colony Optimization

Ant Colony Optimization (ACO) meta-heuristic is inspired by the behaviour of real ants finding the shortest path between their colonies and a source of food. While walking amid their colony and the food source, ants leave pheromones on the ways they move. The pheromone intensity on the passages increases with the number of ants passing through and drops with the evaporation of pheromone. As the time goes on, smaller paths draw more pheromone and thus, pheromone intensity helps ants to recognize smaller paths to the food source[1].

3.3 Parameters Setting of Cloudsim

The experiments are implemented with 10 Data Centers with 40VMs and 600-1000 tasks under the simulation platform. The length of the task is from 20000 Million Instructions (MI) to 40000 MI. The parameters setting of cloud simulator are shown in 1.
Table 1: Parameters Setting of Cloudsim

<table>
<thead>
<tr>
<th>Entity Type</th>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task(Cloudlet)</td>
<td>Length of Task</td>
<td>20000-400000</td>
</tr>
<tr>
<td></td>
<td>Total Number of Task</td>
<td>600-1000</td>
</tr>
<tr>
<td>Virtual Machine</td>
<td>Total Number of VMs</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>MIPS</td>
<td>256-512</td>
</tr>
<tr>
<td></td>
<td>VM Memory (RAM)</td>
<td>1024</td>
</tr>
<tr>
<td></td>
<td>Bandwidth</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Cloudlet Scheduler</td>
<td>Time_shared and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Space_shared</td>
</tr>
<tr>
<td></td>
<td>Number of PEs Requirement</td>
<td>1-2</td>
</tr>
<tr>
<td>Data Centre</td>
<td>Number of Datacenter</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Number of Host</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>VM Scheduler</td>
<td>Time_shared and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Space_shared</td>
</tr>
</tbody>
</table>

[4] RESULT AND DISCUSSION

The implementation has been done using two approaches. In the first approach, size of VM has been kept fixed and the size of cloudlets is changed. In the second approach, size of cloudlets is fixed and the size of VM is changed. The experiment result for the two approaches is given below:

![Fig: 3 Makespan (fixed VM=40)](image)

It can be seen that SJF performs as good as ACO. But SJF is practically not implementable in real life scenario as in dynamic environment. SJF may result in starvation of jobs with higher expected time to compute.
In the second approach, ACO performs better than the SJF. Performance of ACO is as good as SJF which is sure to produce good results, but SJF cannot be implemented practically.

[6] CONCLUSION

It is clear from the results that ACO performs much better than the two static algorithms. The gap in the Makespan widens as you increase the number of cloudlets keeping the VMs constant or when you decrease the VMs keeping the cloudlets constant. In this paper the probability function of ACO only focuses on the Makespan and does not consider the utilization of other VM parameters like bandwidth, RAM, storage. In future these parameters can also be considered and moreover migration based on current load on the VM in terms of these parameters can also be designed. Even though the SJF performs as well as ACO, SJF cannot be implemented in practical scenario.

REFERENCES


Authors Profile

Mr. Dinesh Gupta, did his M. Tech in IT from Department of CSE GNDU Amritsar, India. Currently he is pursuing his Ph. D in CSE from Desh Bhagat University. He has more than 8 years of experience in teaching. Currently he is working as Assistant Professor in department of CSE, IKGPTU, India. He has more than 8 publications in leading research Journal.

Dr. Harmaninder J. S. Sidhu is serving as Asstis-tant Professor in Department of Computer Science and Applications, Desh Bhagat University, Mandi Gobindgarh, Punjab, India. He is involved in academic activities for last 10 years. His interest areas include Open Source Software and Algorithm Analysis.

Mr. Gagan Kumar did his M. Tech in IT from Department of CSE GNDU Amritsar, India. Currently he is pursuing his Ph. D in CSE from IKGPTU, India. He has more than 7 years of experience in teaching. Currently he is working as Assistant Professor in department of IT, DAVIET, India.