Symbolic Scheduling

Shihab Ud Doula
Dept. Of Electronic Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
Email: shihab-ud.doula@stud.hshl.de

***Abstract*— Scheduling, or planning, is widely recognized as a very important step in several domains such as high-level synthesis, real-time systems, and everyday applications. Given a problem described by several actions and their relationships, finding a schedule, or a plan, means to find a way to perform all the actions minimizing a specific cost function. The goal of this paper is to develop, analyze and compare different scheduling techniques on a new scheduling/planning problem. This paper talks about Symbolic scheduling techniques which is a technique for scheduling tasks in a system, where the tasks are represented symbolically. This means that the tasks are not represented as specific instructions, but rather as abstract entities that can be manipulated by the scheduler. Among several symbolic scheduling techniques, we show Heuristic based scheduling technique which is based on Cloud based computing system. It shares with previous ones the underlying problem definition, but it also unveils brand new challenging characteristics, and a different optimization target. We show how to model the problem in a suitable way, and how to solve it with different methodologies going from First-come, first-served (FCFS), to Priority scheduling** [1] [2]**. New ideas are put forward in the different domains having efficiency and scalability as main targets.**

Keywords—Symbolic Scheduling, Heuristic task, cloud computing.

I. INTRODUCTION

Symbolic scheduling is a technique for scheduling tasks in a system, where the tasks are represented symbolically. This means that the tasks are not represented as specific instructions, but rather as abstract entities that can be manipulated by the scheduler. Symbolic scheduling can be used to solve a variety of scheduling problems, including Real-time scheduling, Multiprocessor scheduling and Heterogeneous scheduling. Symbolic scheduling can be more efficient than traditional scheduling techniques, such as exhaustive searches, because it can rule out impossible schedules early in the search process. However, symbolic scheduling can also be more complex, and it may not be able to find all possible schedules. For example, it may not be able to find all possible schedules and it can also introduce overhead while representing the tasks symbolically. But by its efficiency in exhaustive search adding with its flexibility to solve real time or multiprocessor scheduling and large number of task scalability it overcomes all the complexities.

There are two main types of symbolic scheduling: exact symbolic scheduling and approximate symbolic scheduling [3].
Exact symbolic scheduling is a technique for finding all possible schedules for a given set of tasks. This is done by representing the tasks and their constraints symbolically, and then using a symbolic solver to find all possible solutions. Exact symbolic scheduling is a powerful technique, but it can be computationally expensive for large sets of tasks.

Approximate symbolic scheduling is a technique for finding good, but not necessarily optimal, schedules for a given set of tasks. This is done by representing the tasks and their constraints symbolically, and then using a heuristic to find a good solution. Approximate symbolic scheduling is less computationally expensive than exact symbolic scheduling, but it may not find the optimal solution.

Some of the most common types of symbolic scheduling [4]:

- Boolean satisfiability problem (SAT)-based scheduling: This is a type of exact symbolic scheduling that uses SAT solvers to find all possible schedules.
- Decision diagram-based scheduling: This is a type of exact symbolic scheduling that uses decision diagrams to represent the tasks and their constraints.
- Heuristic-based scheduling: This is a type of approximate symbolic scheduling that uses heuristics to find good, but not necessarily optimal, schedules.

The choice of which type of symbolic scheduling to use depends on the specific scheduling problem. For problems with a small number of tasks, exact symbolic scheduling may be a good option. For problems with many tasks, approximate symbolic scheduling may be a better option. However, the Heuristic-based scheduling has been discussed in this paper considering that it’s the most effective among above mentioned techniques.

II. HEURISTIC-BASED SCHEDULING

Heuristic-based scheduling is an approach to task scheduling that relies on heuristics or rules of thumb to make decisions about task assignment and resource allocation [5] [6]. It is a practical and efficient method for solving scheduling problems when finding an optimal solution is computationally expensive or not feasible within the available time constraints.

In heuristic-based scheduling, the scheduler uses heuristics or strategies to guide the scheduling decisions. These heuristics are typically based on domain knowledge, experience, or simple rules that prioritize certain factors or objectives. While heuristic-based approaches may not guarantee an optimal solution, they aim to find good or near-optimal solutions that meet specific objectives or criteria.

.

Some commonly used heuristics in scheduling include [2]:

- First-Come-First-Served (FCFS): Tasks are scheduled in the order of their arrival time. It is a simple and intuitive heuristic but may not consider the characteristics of tasks or resource availability.
- Shortest Job Next (SJN): Tasks with the shortest execution time are given priority. This heuristic aims to minimize the average waiting time and can be effective when the execution times of tasks are known in advance.
- Priority Scheduling: Each task is assigned a priority value, and tasks with higher priorities are scheduled first. Priority can be based on factors such as task urgency, importance, or criticality.
- Round Robin: Tasks are assigned fixed time slices or quantum, and each task is allowed to be executed for a certain time period before being preempted. This heuristic ensures fairness in resource allocation by providing equal opportunities to all tasks.
- Deadline-based Scheduling: Tasks are scheduled based on their deadlines. Tasks with earlier deadlines are prioritized to ensure that they are completed on time.
- Genetic Algorithms: These are heuristic optimization algorithms inspired by the process of natural evolution. They use genetic operators such as selection, crossover, and mutation to iteratively search for better scheduling solutions based on fitness functions that evaluate the quality of schedules.

Heuristic-based scheduling approaches are widely used in various domains, including cloud computing, where finding an optimal solution in large-scale environments may be computationally infeasible. By employing heuristics, schedulers can quickly make scheduling decisions based on simplified rules and achieve reasonable results within practical time frames. However, it's important to note that the effectiveness of heuristics depends on the specific problem domain, characteristics of the tasks and resources, and the quality of the heuristic employed.

III. HEURUSTIC SCHEDULING ALGORITM

A flowchart for a heuristic resource constrained scheduling algorithm for project management is shown in Figure 1 [5]. This algorithm works on the principle that individual activities are to start as soon as their predecessor activities are finished if sufficient resources are available at that time. If sufficient resources are not available (i.e., the activity requires a plumber and the plumber is busy on another activity), then the activity is placed in a list of activities.

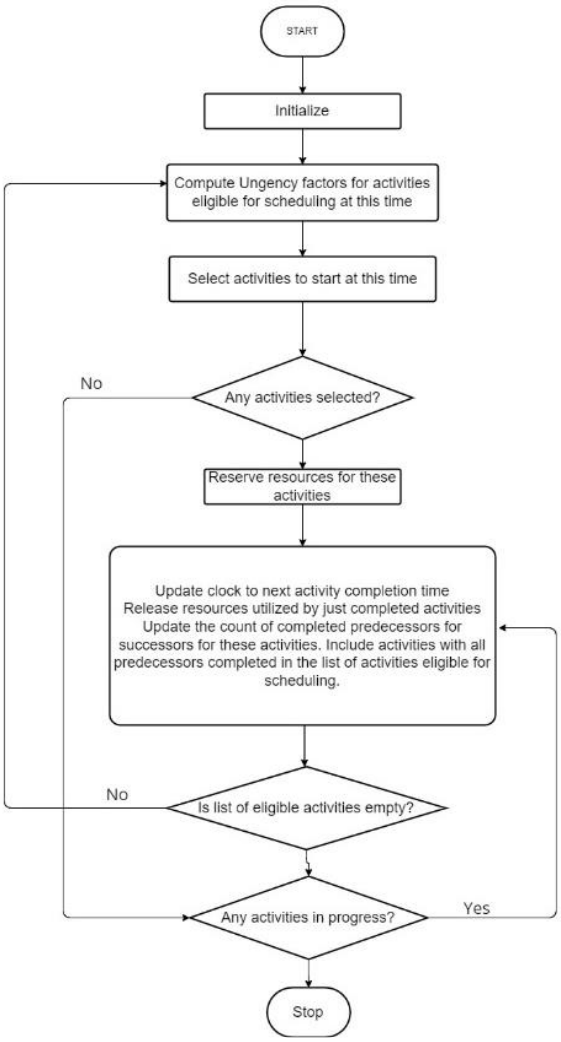

Figure 1: Overview of Heuristic Scheduling

eligible for scheduling as soon as sufficient resources become available. Later when an activity finishes and thus releases its resources, this list is reviewed to see if an activity is waiting for a released resource. Frequently there will be several activities competing for the same resource. In this case, the activities are ranked in order of their "urgency" and activities are assigned resources and started in decreasing order of their urgency. Unfortunately, there is no single best way to compute an activity's urgency. However, resource related variables and variables from the project's critical path network (such as the latest finish time (LFT) and the slack time) frequently work well. An attractive feature of the algorithm in Figure 1 is the fact that the internally computed urgencies can be easily adjusted or overridden by the user, thus facilitating considerable user control over the final schedule.

IV. The Problem and Cloud Computing

Scheduling problems occur in all the economic domains, from computer engineering to manufacturing techniques. Most scheduling problems are complex combinatorial optimization problems and very difficult to solve.Cloud computing is a platform that offers services like computing storage and many more over the internet for cloud consumers using virtualization technology. Cloud acts as a pool of infinite resources for different computation requirements . To get the best out of the cloud system the underlying distributed platform should be utilized efficiently. Resource management is a key consideration for cloud providers as it can save cost and service cloud consumers to meet Quality of Service (QoS) requirements which are mentioned in the Service Level Agreements (SLA) between both parties *[7]*. For the cloud user, cost saving, and improvement of response time will be the benefits of effective resource management in the cloud. There are many resource management techniques in cloud like job scheduling, resource migrations, resource provisioning, server optimization and e.c.t . Task or job scheduling is one of the resource management techniques that is used for efficient resource management in the cloud *[2]*.

V. Task scheduling in cloud

In task scheduling tasks are assigned to virtual machines based on different strategies considering different optimization objectives. Objectives include Makespan, Cost, Resource Utilization and etc. Different strategies have been proposed taking into consideration different factors like execution time, completion time, deadline, budget and many more. Task scheduling can be divided into two as best-effort scheduling and Quality of Service scheduling. In best-effort scheduling, the goal is to optimize one or more objectives and in QoS scheduling, the goal is to execute a set of tasks under certain predefined constraints that are set to meet QoS requirements by the cloud user. Further, there are also hybrid of which do both optimizations while ensuring that certain QoS requirements set up by the end-user are met. Our proposed algorithm falls under best-effort scheduling strategy as it tries to optimize Makespan, Degree of Imbalance and System Throughput [2].

The task scheduling in cloud environment is an NP-complete problem so it is hard to find an optimal solution in polynomial time [2]. The scheduling in cloud improves the resource utilization and reduces the overall completion time. There does not exist a standard task scheduling technique that could be extended to a large-scale environment. The main job of task scheduler is to distribute customer requests to all the present resources to execute them. Task scheduling becomes very important from the user's point of view as they must pay based on usage of resources based upon time. There are different effective resource scheduling criteria which reduces execution cost, time, energy and increases CPU utilization and productivity.

VI. Performance metrices [1]

A broad classification can be done into the following categories: static, dynamic, preemptive, non-preemptive, centralized, and decentralized scheduling. The major performance metrics used in the literature are as follows:

Cost

Cost means the total payment generated against the utilization or usage of resources, which is paid to the cloud providers by the cloud users. The main determination is to the growth of revenue and profit for cloud providers while reducing the expenses for cloud user with efficient utilization. Assume the cost of a VM varies from on another based on time substantial and VM's specification as specified by the cloud providers, then Eq 1 holds for the cost of executing task of a VM.

$$Cost = \sum_{i=1}^{n} task^{i}(Ci * Ti)$$

Where *Ci* represents the cost of i^{th} VM and *Ti* represents the execution time of i^{th} task.

Degree of imbalance

Degree of imbalance (DI) describes the amount of load distribution amongst the VMs regarding their execution competencies. Here, T_{max}, T_{min} and T_{avg} signify the maximum, minimum and average overall execution time of task among total VMs, correspondingly.

$$DI = \frac{T_{max} - T_{min}}{T_{avg}}$$

Makespan

Makespan is used to estimate the maximum completion time, by evaluating the finishing time of the latest task, when all tasks are scheduled. If the makespan of specific cloudlet or task is not minimized, then the demand will not be completed on time.

$$Makespan = max_{task^i}(Fnh_{Time})$$

where Fnh_{Time} shows the finishing time of i^{th} task.

Throughput

Throughput uses the consideration of total number of tasks, which are implemented successfully. In cloud computing, throughput means some tasks completed in a certain time period. Minimum throughput is required for task scheduling.

$$Throughput = \sum_{task^i}(Exe_{Time})$$

where Exe_{Time} shows the execution time of i^{th} task.

VII. MIN-MIN ALGORITHM [8]

Among the many heuristic algorithms such as MCT,MET,Max-min,Suffrage , I believe Min min is the most effective in cloud computing as it aims to minimize the makespan (total time to complete all tasks) in a scheduling problem. It follows a simple and intuitive approach by prioritizing tasks with the minimum remaining processing time and assigning them to available resources that can execute them the fastest. Scheduling of user tasks is a very complicated process in a cloud computing environment. Min-min algorithm shortcoming is the long tasks may not be scheduled. At present, Google, IBM, Amazon, Microsoft, and other IT vendors are studying and promoting cloud computing applications. Multi-tasks scheduling is the NP-Complete problems. Therefore, cloud computing task scheduling has a great significance.

- The Scheduling Model of Min-Min

The thinking of the min-min algorithm is as quickly as possible to allocate each task to resources which can complete the task in the shortest possible time. Ease of description is defined as follows:

T_{exe} = execution time; T_{start} =starting time; T_{finish} = finish time; T_{comp} = completion time

Then, $T_{finish} = T_{exe} + T_{start}$

First, calculate the T_{exe} on each node in each machine. Then assign the earliest T_{exe} of the task to the corresponding machine node. Finally, the mapped task is deleted, and the process repeats until all tasks are mapped.

Completion time= T_{comp}= max(T_{finish}= T_{exe} + T_{start})

Pseudo-Code for Min-min Algorithm

```
1:  for i = 1 to M        // M denotes the number of tasks to be scheduled
2:      for j = 1 to N    // N denotes the number of virtual machines
3:          C_ij = E_ij + R_j
                          // C_ij denotes the completion time of task
                          // E_ij denotes execution time of task
                          // R_j denotes the ready time of task i on virtual machine j
4:      end for
5:  end for
6:  do until all the unscheduled tasks are exhausted
7:      for each unscheduled task
8:          find the minimum completion time of the task and virtual machine that obtains it
9:      end for
10:     find the task t_p with earliest completion time
11:     assign task t_p to the virtual machine that gives the minimum completion time
12:     delete task t_p from pull of unscheduled tasks
13:     update the ready time of the machine that gives the minimum completion time
14: end do
```

Figure 2: Algorithm Procedure [1]

A flow chart of min min algorithm is also given below:

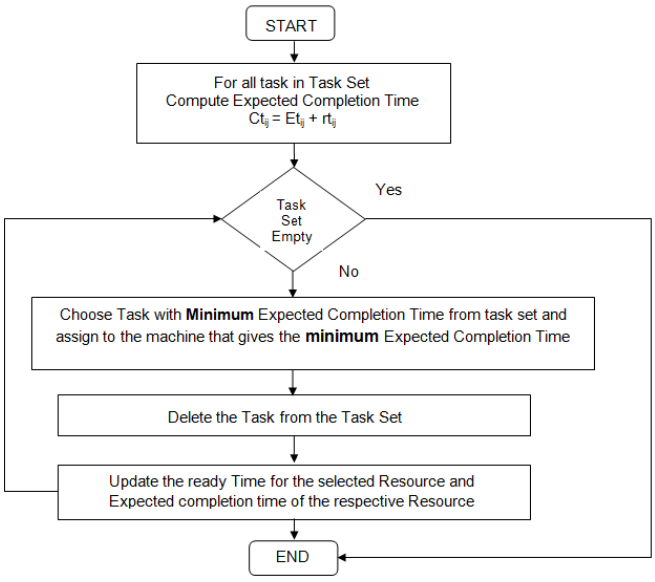

Figure 3: Flowchart Min-min Algorithm [9]

Min-Min algorithm considers all which are not assigned tasks in each task mapping, Min-min algorithm will execute until the whole tasks set is empty. Min-min will execute short tasks in parallel and the long tasks will follow the short tasks. Min-min algorithm shortcoming is the short tasks scheduled first, until the machines are leisure to schedule and execute long tasks. Min-min can cause both the whole batch tasks executed time get longer and unbalanced load. Even long tasks cannot be executed. Compared with the traditional Min-min algorithm, improved algorithm adds the three constraints strategy which can change this condition.

VIII. Experiment and Result [10]

To justify min min algorithm to be most effective an experiment has been done on Cloudsim comparing it with Max-min scheduling techniques.

Algoritm for max-min:

The Max-Min algorithm:
1. for all submitted tasks Ti in M
2. for all resource Rj Calculate minimum turnaround time
3. Do until all task in task set M are executed,
4. find task Tk having maximum execution time (largest task)
5. assign task Tk to the resource Rj which gives the minimum completion time (Slowest resource)
6. remove task Tk from M 7. update resource status.

Here, Number of processing element – 1
Number of hosts – 2
Table I. Configuration of Host

RAM (MB)	20240
Processing Power (MIPS)	220000
VM Scheduling	Time shared

Table II shows the configuration of the system on which these results are obtained is as shown below:

Processor	Pentium® Dual-core CPU T4400@2.20GHz,2.20GHz
RAM	8.00GB
System types	32bit operating system
Operating System	Windows 7 Ultimate

Table II. System Configuration

VM	RAM	Processing Power(MIPS)	Processing Element
VM1	5024	22000	1
VM2	1048	11000	1

Table III is showing configuration of VMs for this experiment. Table III: Configuration of VMs

Performance with Time: - The experimental results show the remarkable improvement in time over the sequential approach as well as shortest Job First without fair priority.

No. of tasks	Makespan of Min-Min algorithm	Makespan of Max-Min algorithm
20	71.474	68.556
40	298.753	276.548
60	576.441	496.774
80	734.658	645.886
100	1198.745	1105.885

Table IV: Results of comparison of makespan of max-min and min-min algorithm

The results shown above are the standard mean of total execution cost obtained after several number of execution for each number of cloudlet (e.g. we have run the implementation 20 times for 100 cloudlets and calculated its mean). Results may somewhat vary according to configuration of machine used.

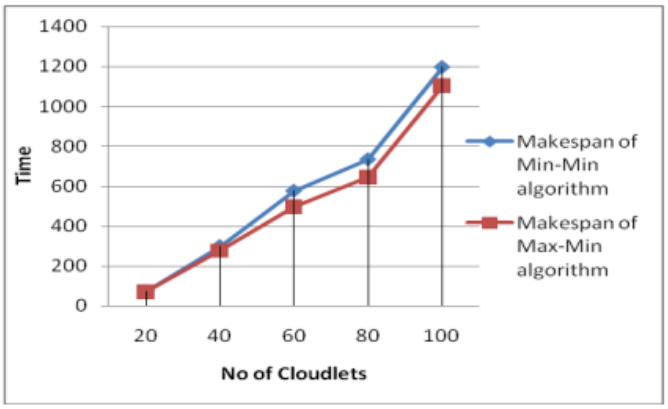

Figure 4: Comparision of total execution time i.e. makespan of min-min and max-min scheduling. [10]

Makespan means total completion time required for job execution. The above bar graph showing task completion time and comparison of min-min and max-min scheduling algorithm. It is observed from the results in Figure 3, that the max-min scheduling algorithm achieves lower makespan when compared to min-min scheduling. Thus task scheduling algorithm is for mapping jobs submitted to cloud environment onto available resources in such a way that the total response time, the makespan, is minimized as shown in Table IV. Efficient unitization of resources and task selection method helps to achieve high performance with heuristic optimization techniques like Particle swarm optimization (PSO) and Evolutionary algorithm (EA)

IX. OPTIMIZED TECHNIQUES [2]

The performance of a system is directly influenced by the efficiency of task execution schedule. To achieve this, a number of optimization algorithms for allocating and scheduling the resources proficiently in the cloud have been proposed over the years. Some of these are:

Genetic Algorithm (GA) where the term “Survival of the fittest “is employed as a strategy method of scheduling.

Harmony Search Algorithm (HS) which is inspired from the process of musicians searching for perfect harmony.

Cuckoo Optimization Algorithm (COA) which is inspired from obligate brood parasitism of cuckoo which lays eggs in the host nest having Lévy flight behavior of the birds.

Artificial Bee Colony (ABC) which is meta-heuristic optimization technique, inspired from foraging conduct of honeybee colonies.

Ant Colony Optimization (ACO) which is also meta heuristic method inspired from food searching method of ants.

Gravitational Search Algorithm (GSA) which is an optimization technique method based on “Gravitational Law” [21]. This algorithm is basically a population based multi-dimensional optimization algorithm where agents are called as objects and their performance can be calculated by their masses.

X. SUMMARY AND CONCLUSION

An approach for symbolic scheduling in cloud computing has been presented in the paper [4]. It is based on heuristic-based scheduling which tends to be more effective according to our research. Algorithms and flowchart for heuristic based scheduling will help us understand better how the system works. And among various algorithm pf heuristic scheduling min-min algorithm has been discussed and experimented with the max-min algorithm. Further work concentrates the results of each algorism and alternate optimized technique in scheduling processes. Our cloud Sim Simulation verifies our that min-min algorithm is the most effective way in cloud computing as scheduling algorithms are not an easy task to know which works better until given in correct environment.

XI. REFERENCES

[1] S. H. H. Madni, M. S. Abd Latiff, M. Abdullahi, S. M. Abdulhamid and M. J. Usman, "Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment," *PloS one*, vol. 12, p. e0176321, 2017.

[2] S. K. Patel and A. Singh, "Task Scheduling in Cloud Computing Using Hybrid Meta-Heuristic: A Review," in *Proceedings of the International Conference on Paradigms of Computing, Communication and Data Sciences: PCCDS 2020*, 2021.

[3] K. Luckow, C. S. Păsăreanu, M. B. Dwyer, A. Filieri and W. Visser, "Exact and approximate probabilistic symbolic execution for nondeterministic programs," in *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering*, 2014.

[4] K. Strehl, L. Thiele, D. Ziegenbein, R. Ernst and J. Teich, "Scheduling hardware/software systems using symbolic techniques," in *Proceedings of the seventh international workshop on Hardware/software codesign*, 1999.

[5] A. Thesen, "Heuristic project scheduling," *Project Management Quarterly*, 1978.

[6] P. Fattahi, M. Saidi Mehrabad and F. Jolai, "Mathematical modeling and heuristic approaches to flexible job shop scheduling problems," *Journal of intelligent manufacturing*, vol. 18, p. 331–342, 2007.

[7] K. P. N. Jayasena, K. M. S. U. Bandaranayake and B. T. G. S. Kumara, "TRETA-A Novel Heuristic Based Efficient Task Scheduling Algorithm in Cloud Environment," in *2020 IEEE REGION 10 CONFERENCE (TENCON)*, 2020.

[8] G. Liu, J. Li and J. Xu, "An improved min-min algorithm in cloud computing," in *Proceedings of the 2012 International Conference of Modern Computer Science and Applications*, 2013.

[9] M. Haladu and J. Samual, "Optimizing Task Scheduling and Resource allocation in Cloud Data Center, using Enhanced Min-Min Algorithm," 2016.

[10] D. R. K. C. Deepika Saxena and S. Saxena., "Evaluating Makespan of Min-Min and Max-MinScheduling Techniques in Cloud Computing:Concepts and Comparision," *International Journal of Trend in Research and Development, Volume 3(6), ISSN: 2394-9333*, 2016.

[11] I. Radivojevic and F. Brewer, "A new symbolic technique for control-dependent scheduling," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 15, p. 45–57, 1996.