**5. RESULTS**

***5.1. Basis of Analysis and Comparison:***

There can be various basis of analysis of performance for cloud scheduling algorithms. There’s no single unified method that makes the comparison on a single scale, rather, the performance of an algorithm can be evaluated based on criteria like throughput, economic cost, time consumed, etc.

For this experiment, we have considered time consumption as our metric for comparison. And for that **Makespan** is chosen as the parameter.

**5.1.1 Makespan**:

It is the maximum time taken by the entire system to execute all the cloudlets in a cloud environment. The goal is to minimize the makespan, i.e, execute all the assigned cloudlets as soon as possible on all the machines. Let T i → C j denote the task Ti is assigned to cloud Cj. Then it is mathematically defined as follows:

where,

Let’s say we have these tasks in an ETC matrix to schedule,

|  |  |  |
| --- | --- | --- |
| Tasks | Machines | M1 | M2 |
| T1 | 20 | 5 |
| T2 | 15 | 25 |
| T3 | 10 | 12 |

And, let’s say we are using Min-Min Cloud scheduling algorithm to schedule tasks and ready time for all the machines is 0 initially.

T2(15)

T1(5)

T3(10)

M1

M2

So, as visible, out of all the Machines, two in this case, machine M1 takes the maximum time to execute all the assigned tasks to it. This time is known as Makespan of the cloud environment.

***5.2. Dataset:***

For the experiment, we picked two benchmark datasets which were generated by **Buran et al**. The first dataset contains 12 instances and the second contains 24 instances respectively. The dimension of first dataset is 512x16 and that of second is 1024x32. Here, 512 and 1024 are the number of cloudlets in each of the instance; 16 and 32 are the number of VMs on which the tasks are to be scheduled, respectively. Each of the instance is named in a format as, “*u\_x\_yyzz*”, where u denoted *uniform; x* may be *c(consistent), i(inconsistent) and s(semi-consistent)*; *yy and zz* can either be *hi(high) or lo(low)* representing the task and VM heterogeneity respectively*.*

***5.3. Solution Formulation:***

Hence to formulate the solution to the proposed problem, we try to take different algorithms, and generate schedules based on them on the instances of the datasets used, calculate their makespan value, and make comparisons. We aim to find the algorithm that produces schedules that give the minimum makespan, i.e, they give minimum completion time.

***5.4. Experimental Results:***

These results are divided into two parts. One is for the dataset with instances of size 1024x32, and another is for the dataset with instances of size 512x16. To evaluate the performace of proposed algorithm it’s compared with the outcomes of Min-Min Cloud Scheduling algorithm and Max-Min Cloud Scheduling algorithm. These two are heuristic algorithms and are used as a scale to compare the performance. Along with this, the proposed algorithm which is a variation of genetic algorithm, is run with different sets of initial population.



Comparison of makespan for Min-Min, Max-Min heuristic, Genetic Algorithm using Random Population,

Min-Min Population, Max-Min Population and Both Population using benchmark Datasets of shape 1024x32

This includes, initial population containing the output of Min-Min algorithm as one of the parent; another with output of Max-Min algorithm; and a third with output of both these algorithms as parents.

A similar comparison is done for dataset with 512x16 sized instances. Which is as follows:



Comparison of makespan for Min-Min, Max-Min heuristic, Genetic Algorithm using Random Population,

Min-Min Population, Max-Min Population and Both Population using benchmark Datasets of shape 512x16