COMPARISON OF SERVER LOAD BALANCING TECHNIQUES IN CLOUD SIMULATED ENVIRONMENT

A Report
in Partial Fulfilment of the Requirements
for the Course of
Minor Project - I
In
Third year – Fifth Semester of
Bachelor of Technology
Specialization in
CCVT.

Under the guidance of

Ms. Avita Katal,
Assistant professor(SS)
Department of Virtualization

Submitted By: -

Priyansh Joshi	**500068830**	**R110218111**
Rishika Bansal	**500069750**	**R110218123**
Tushar Jain	**500067385**	**R110218173**
Yashika Jain	**500069554**	**R110218191**

UNIVERSITY WITH A PURPOSE
Department of Cybernetics and Virtualization
SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
Bidholi Campus, Energy Acres, Dehradun – 248007

July-Dec,2020-21

UPES

School of Computer Science

University of Petroleum & Energy Studies, Dehradun

<u>Candidate Declaration</u>

I/We hereby certify that the project work entitled Title of Project in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering with Specialization in CCVT and submitted to the Department of Virtualization at School of Computer Science, University of Petroleum and Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from September, 2020 to December, 2020 under the supervision of Ms. Avita Katal, Assistant Professor – Senior Scale, Department of Virtualization.

The matter presented in this project has not been submitted by me/ us for the award of any other degree of this or any other University.

R110218111	**Priyansh Joshi**
R110218123	**Rishika Bansal**
R110218173	**Tushar Jain**
R110218191	**Yashika Jain**

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Date: December 5, 2020

ACKNOWLEDGEMENT

We wish to offer our profound thanks to our guide **Ms. Avita Katal**, for all exhortation, consolation and consistent help she has given us all through our undertaking work. This work would not have been conceivable without her help and important recommendations.

We sincerely thank to **our Head of the Department, Dr. Deepshikha Bhargava** for their great support in doing our project Comparison of Server Load Balancing Techniques in Cloud Simulated Environment at SoCS.

We are likewise appreciative to **Dr. Manish Prateek Professor and Dean SoCS** for giving us the important offices to do our task work effectively.

We might want to thank every one of our **companions** for their assistance and useful analysis during our project work. At long last we have no words to offer our true thanks to our folks who have demonstrated us this world and for each help they have given us.

Priyansh Joshi	**Rishika Bansal**	**Tushar Jain**	**Yashika Jain**
R110218111	**R110218123**	**R110218173**	**R110218191**

School of Computer Science

University of Petroleum & Energy Studies, Dehradun

Project Report(2020-2021)

Minor 1

Title:
Comparison of Server Load Balancing Techniques in Cloud Simulated Environment.

Abstract:
The performance of cloud services depends on the scheduling algorithms that distribute the incoming network traffic among their servers to achieve effectiveness in execution of tasks. These algorithms are assigning the tasks to various computing resources, and resources are virtual in nature. In cloud, assigning tasks to corresponding resources are NP-hard in nature. The traditional scheduling algorithms like FCFS, SJF, Round Robin etc. will not be suitable to solve NP-hard scheduling problems. Cloud scheduling considers various criteria like resource utilization, cost, make span and throughput. So, the project will implement cloud scheduling algorithms such as Max-Min Algorithm, Min-Min Algorithm, Enhanced Max-Min Algorithm and Greedy Algorithm for Server load balancing in cloud environment and will perform the comparative analysis of these algorithms.

Keywords:
Load balancing, makespan, min-min, max-min, enhanced max-min, Greedy, completion time, execution time.

TABLE OF CONTENT

TABLE OF FIGURES

1.Introduction

Computing has now become an integral part of our lives. For any assistance people require to perform some tasks with the help of any device. Since, the data in the world is hugely expanding, so is the need to manage that and handle it. Also, the need for executing tasks at an efficient rate and with minimum time elapsed has become a priority. We have limited resources in any computing network that are RAM, CPU, Bandwidth etc. So, the need of the hour becomes to manage these resources in the most efficient way that we came, termed as Load Balancing. The concept behind Load Balancing goes on to managing resources on the server side managing the incoming request in such a way that the load is equally distributed. This indeed results in a better Quality of Service while using Cloud Systems. Also, the Cloud is known for its 5’9 availability which is an obvious result of the efficient scheduling algorithms that manage the load in such a way that the new incoming requests could also be dealt with.

Fig. 1 demonstrates how the task of Load Balancing is done.

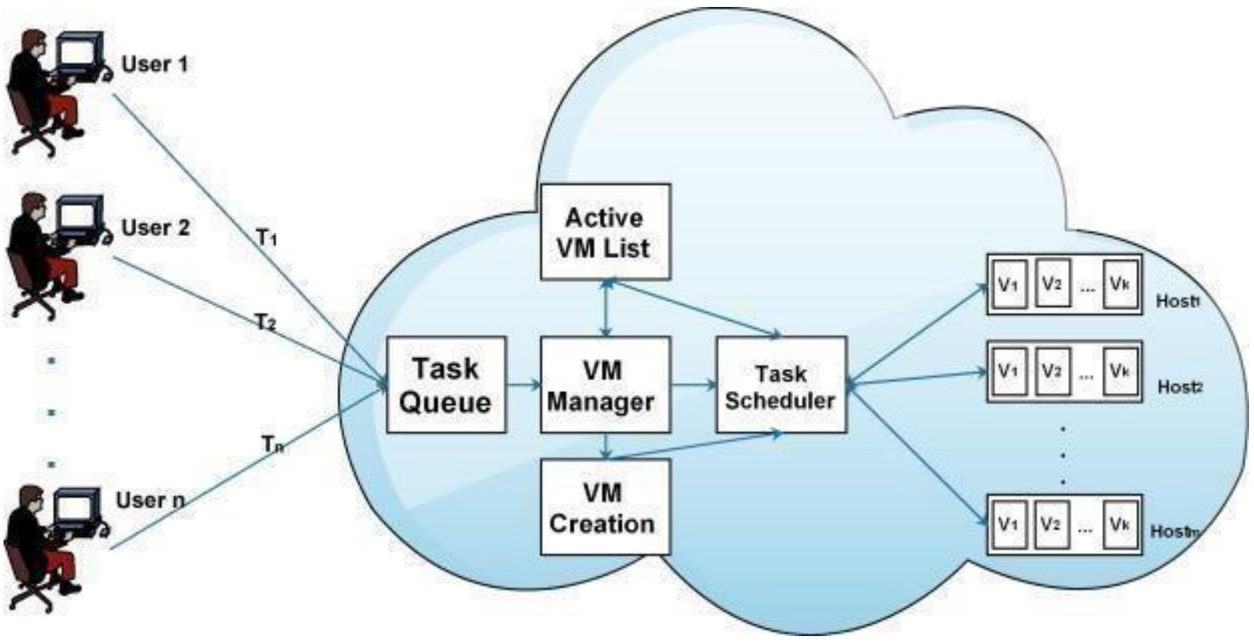


Fig.1: Load Balancing depicted in Cloud Computing Environment.[5]

Cloud Computing has gained immense popularity in the field of technology since it offers a diverse amount of IT services at meagre costs. So, more people tend to shift towards cloud. Due to this there is a need of scheduling incoming requests in such an order that the requirements of all the customers are met keeping in mind the server capacities.
The idea behind Cloud Computing is to convert a huge section of the IT industry into making more attractive services and redefining the concept of IT Hardware and Software for Resource Scheduling. This rapid growth of the cloud systems requires effective algorithms for efficient allocation of resources.
The current four algorithms i.e. Max-Min Algorithm, Min-Min Algorithm, Enhanced Max-Min Algorithm and Greedy Algorithm aim at reducing the makespan of all the tasks assigned to a cloud system with limited servers.
A Datacenter consists of many hosts. Every host takes the Virtual Machines on the basis of their hardware specifications i.e. Processing Power, Memory and Bandwidth. In Task Scheduling, the tasks are divided according to two policies which are Time shared Policy or Space shared Policy. In Time shared Policy each VM has an equal amount of time i.e. tasks are performed in parallel limiting to their capacity. In Space shared Policy, each task will be having equal share of the CPU or the processing unit.
In Task Scheduling, [6] the main focus is at the following factors that are: *Metatask*, *Makespan, Minimum Execution Time and Minimum Completion Time.*

2.Literature Review

- In [2], the authors Upendra Bhoi and Purvi N. Ramanuj told us about a distinctive modification of Improved Max-min task scheduling algorithm. The authors told us that Improved Max-min is built on expected execution time instead of completion time as a selection basis. The authors also told us that Enhanced (Proposed) Max-min is also based on the expected execution time instead of completion time as a selection basis but the only difference is that Improved Max-min algorithm allocates task with Maximum execution time (Largest Task) to resource produces least completion time (Slowest Resource) while Enhanced Max-min assign task with average execution time (average or Nearest greater than average Task) to resource produces least completion time (Slowest Resource).
- In [4], the authors Gaurang, Rutvik, et.al explains study of variety of task scheduling algorithms and depicts modification of Load balanced Min-Min (ELBMM) algorithm for Static Meta-Task Scheduling. The modified algorithm is made on catholic study of Load balanced Min-Min algorithm for Static Meta-

Task Scheduling. Enhanced Load balanced Min-Min algorithm (ELBMM) is established on Min-Min strategy and tasks rescheduling to use the unaccustomed resources effectively. The authors told that the algorithm works on the concept where the task with maximum completion time is chosen and assigned to suitable resource to fabricate better makespan and utilize resource productively.

- In [6], the authors Swati Arti, and Dr. Sanjay Tyagi showed that in order to solve the problem of load balancing, an efficient task scheduling algorithm should be depicted in order to reduce completion time, increase resource utilization and reduce makespan etc. The authors explained that there are various types of scheduling algorithms based on different efficiency matrix. In this paper, two heuristic based algorithms: Min-Min and Max-Min have been explained. The authors discussed the key aspects and the basic terminology behind the two algorithms. The authors also gave us a comparison-based analysis based on the makespan parameter.

3. Problem Statement

To design effective task scheduling algorithms that have the ability to solve NP-Hard Problem in such a way that whenever a new task is ready to be executed, we need to assign it to a server so as to minimize the completion time of all the tasks. The objective throws a light on comparative analysis of different task scheduling algorithms on the basis of their makespan.

4.Objectives

- ➔ Implementing Server load balancing techniques:
    - o Comparing cloud scheduling algorithms such as Max-Min algorithm, Min-Min algorithm, Enhanced Max-Min Algorithm and Greedy Algorithm.
- ➔ Understanding C language and Data Structure.

5.Methodology

5.1 Software Development Model Used – Agile SDLC

A load balancing algorithm consistently attempts to answer a particular issue. In addition to other things, the idea of the assignments, the algorithmic unpredictability, the equipment design must be considered. As the traffic on the internet growing rapidly, which is about 100% annually of the present traffic. Hence, the workload on the server developing so fast which results in the overloading of servers especially for popular web server.

The Agile SDLC Model includes pair programming which decreases the quantity of blunders in the turn of events or coding stage and is better than an individual software engineer doing all the hard part. Apart from this it manages to trim down the entire development time of any project. The whole venture executes the procedures of agile software development life cycle Model.

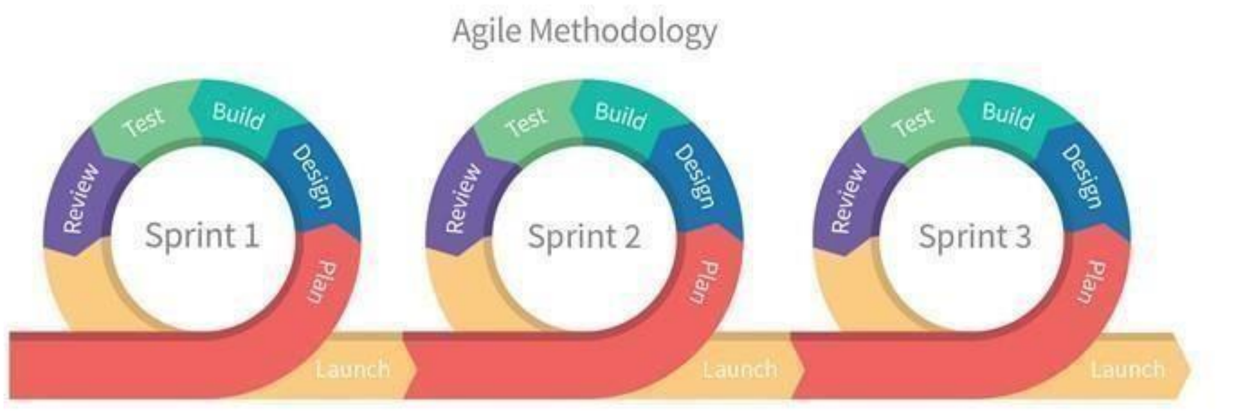


Fig 2: SDLC Agile Software Development Lifecycle [7]

5.2 Workflow

Stream of the task is as per the project:

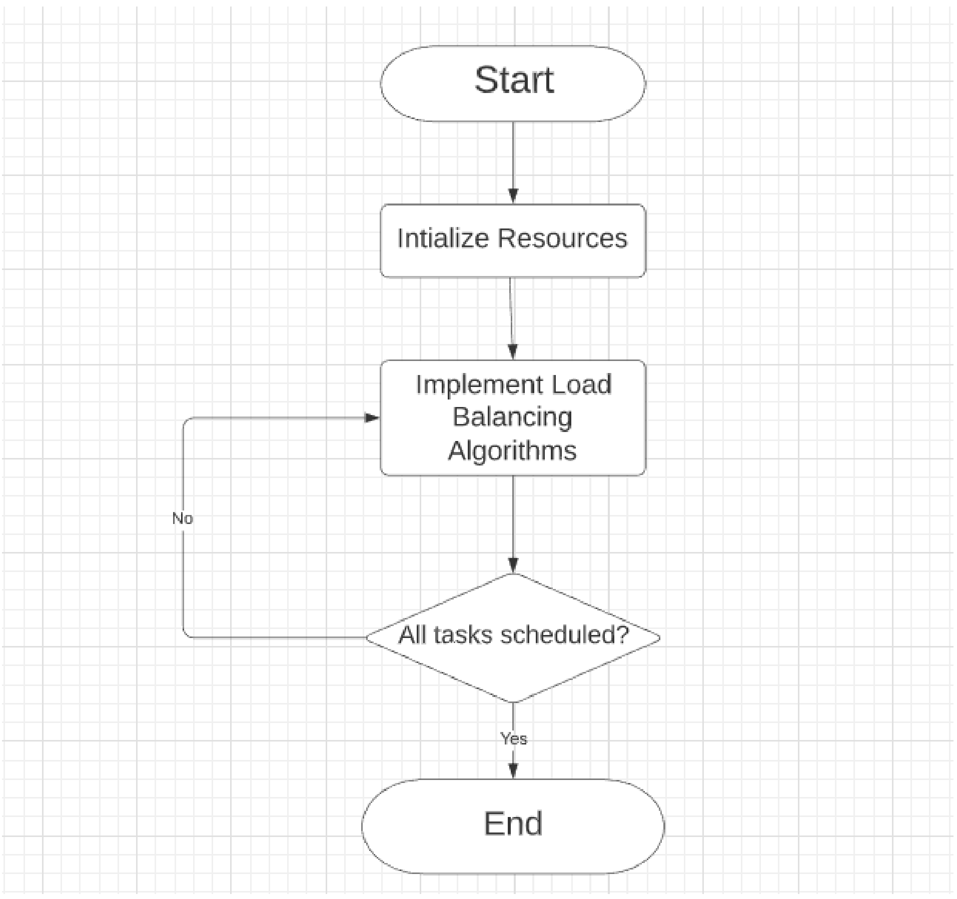


Fig 3: Flow of the project

5.3 Data Flow Diagram

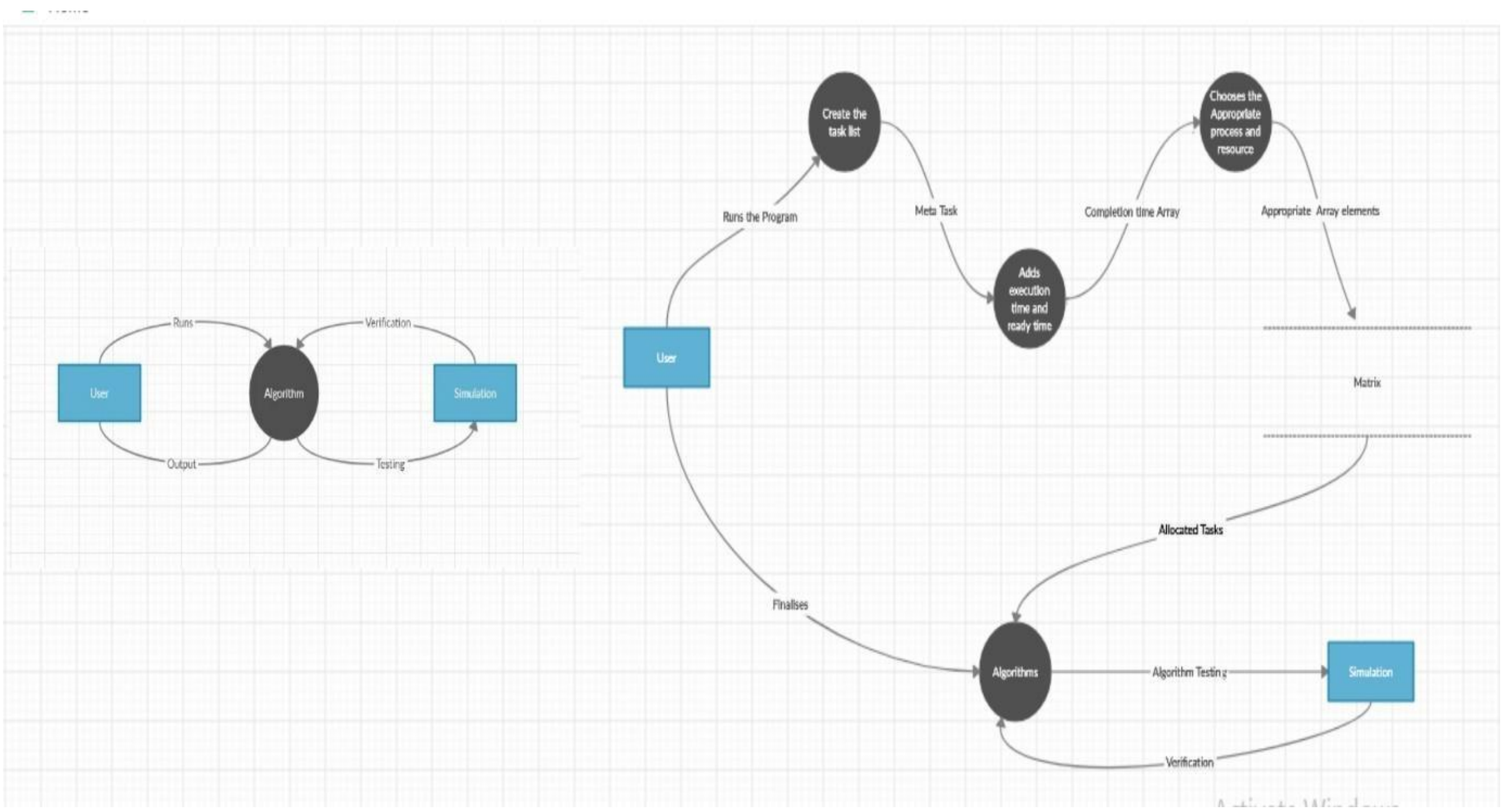


Fig 4: Data Flow Diagram

6.Implementation and Result

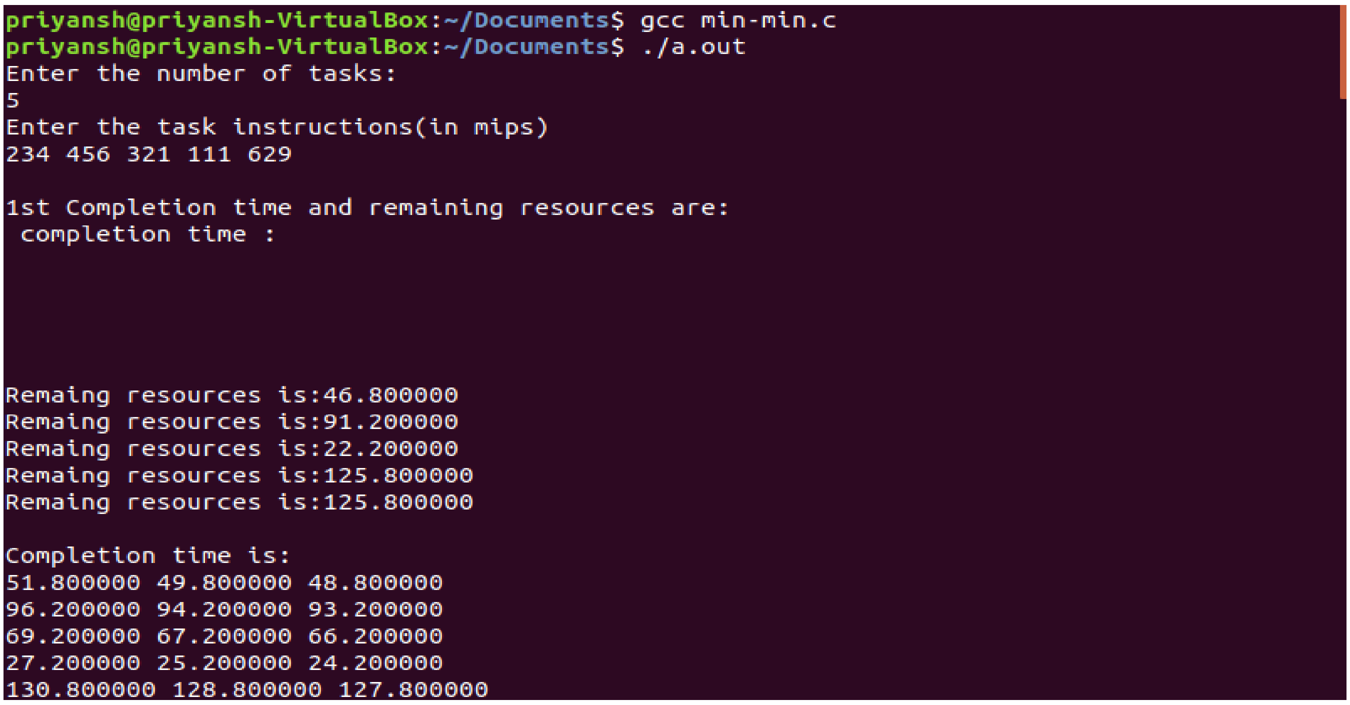
6.1 Load Balancing Algorithms

Henceforth, we're enforcing diverse load balancing algorithms which efficaciously manages the workflow and has low time complexity that include:

6.1.1Min-Min Algorithm:

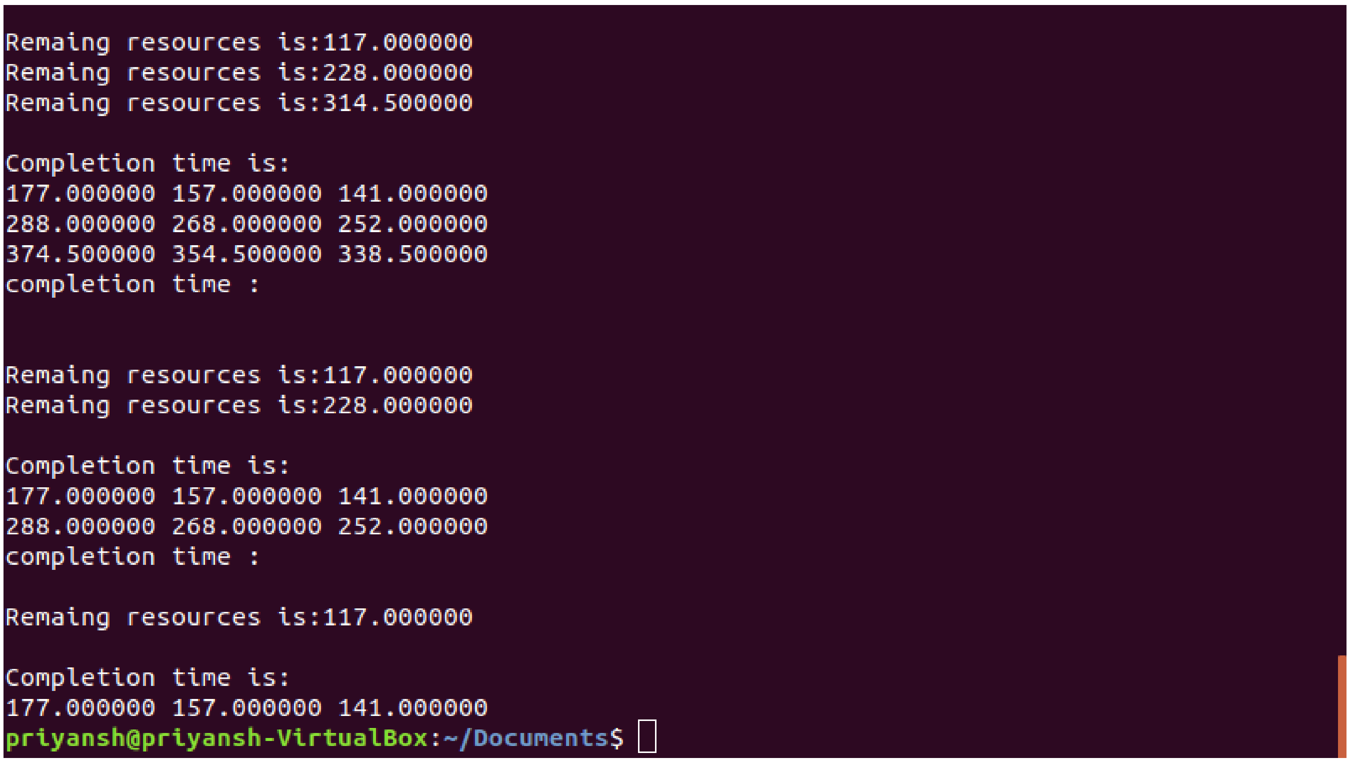
```
MIN-MIN ALGORITHM:

// Create tasks and resources array.
//Phase 1
For all submitted tasks in the set; Ti
2. For all resources; Rj
3. Ctij=Etij+rtj;
        End For; End For; //Calculate completion time of each task.
//Phase 2

4. Do while tasks set is not empty
5. Find task Tk that cost minimum execution time.
6. Assign Tk to the resource Rj which gives minimum expected complete time
7. Remove Tk from the tasks set
8. Update ready time rtj for select Rj
9. Update Cij for all Ti
10. End Do //This loop will end only when the tasks array is empty.
```

Fig 5: Min-Min Algorithm



```
priyansh@priyansh-VirtualBox:~/Documents$ gcc min-min.c
priyansh@priyansh-VirtualBox:~/Documents$ ./a.out
Enter the number of tasks:
5
Enter the task instructions(in mips)
234 456 321 111 629

1st Completion time and remaining resources are:
 completion time :





Remaing resources is:46.800000
Remaing resources is:91.200000
Remaing resources is:22.200000
Remaing resources is:125.800000
Remaing resources is:125.800000

Completion time is:
51.800000 49.800000 48.800000
96.200000 94.200000 93.200000
69.200000 67.200000 66.200000
27.200000 25.200000 24.200000
130.800000 128.800000 127.800000
```

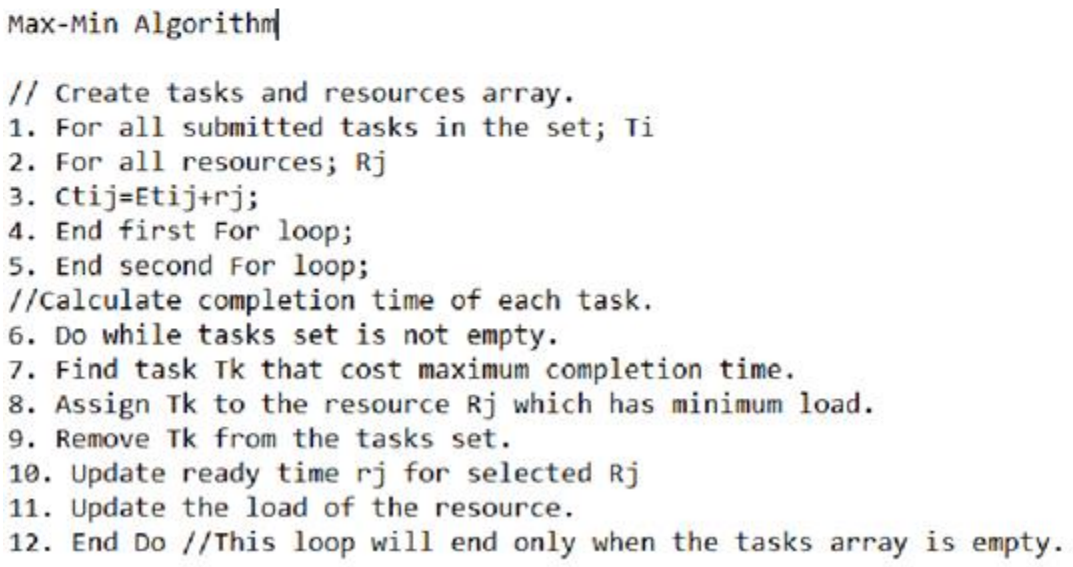
Fig 6.1: Min-Min Output



```
Remaing resources is:117.000000
Remaing resources is:228.000000
Remaing resources is:314.500000

Completion time is:
177.000000 157.000000 141.000000
288.000000 268.000000 252.000000
374.500000 354.500000 338.500000
completion time :


Remaing resources is:117.000000
Remaing resources is:228.000000

Completion time is:
177.000000 157.000000 141.000000
288.000000 268.000000 252.000000
completion time :

Remaing resources is:117.000000

Completion time is:
177.000000 157.000000 141.000000
priyansh@priyansh-VirtualBox:~/Documents$
```

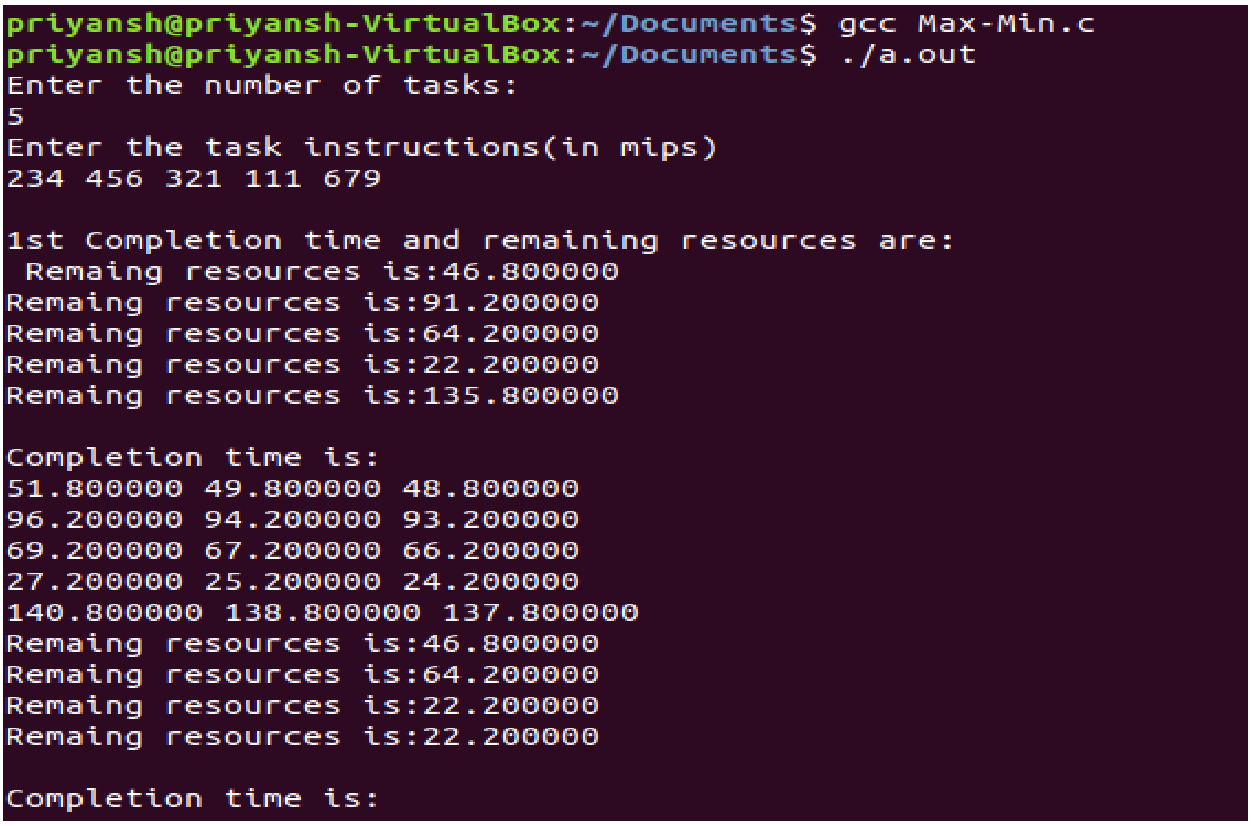
Fig 6.2: Min-Min Output

6.1.2 Max-Min Algorithm:



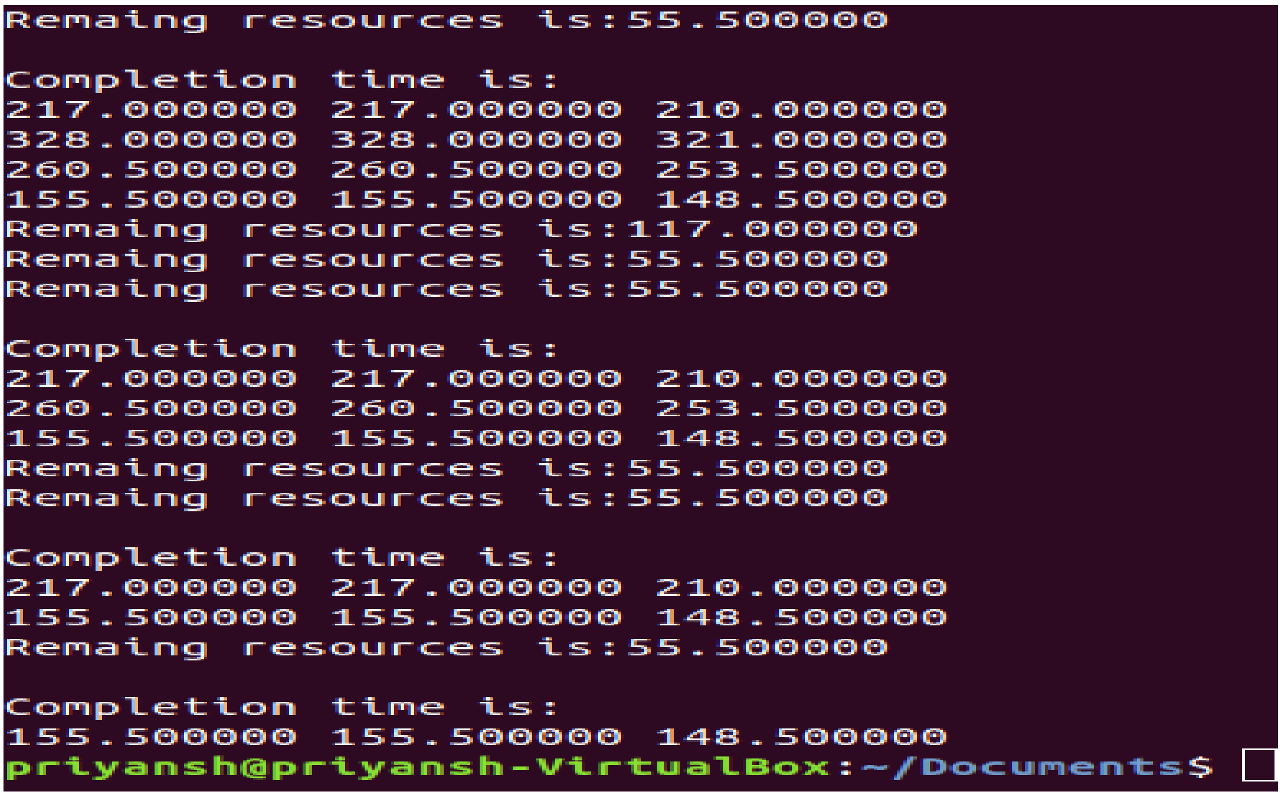
```
Max-Min Algorithm

// Create tasks and resources array.
1. For all submitted tasks in the set; Ti
2. For all resources; Rj
3. Ctij=Etij+rj;
4. End first For loop;
5. End second For loop;
//Calculate completion time of each task.
6. Do while tasks set is not empty.
7. Find task Tk that cost maximum completion time.
8. Assign Tk to the resource Rj which has minimum load.
9. Remove Tk from the tasks set.
10. Update ready time rj for selected Rj
11. Update the load of the resource.
12. End Do //This loop will end only when the tasks array is empty.
```

Fig 7: Max-Min Algorithm



```
priyansh@priyansh-VirtualBox:~/Documents$ gcc Max-Min.c
priyansh@priyansh-VirtualBox:~/Documents$ ./a.out
Enter the number of tasks:
5
Enter the task instructions(in mips)
234 456 321 111 679

1st Completion time and remaining resources are:
 Remaing resources is:46.800000
Remaing resources is:91.200000
Remaing resources is:64.200000
Remaing resources is:22.200000
Remaing resources is:135.800000

Completion time is:
51.800000 49.800000 48.800000
96.200000 94.200000 93.200000
69.200000 67.200000 66.200000
27.200000 25.200000 24.200000
140.800000 138.800000 137.800000
Remaing resources is:46.800000
Remaing resources is:64.200000
Remaing resources is:22.200000
Remaing resources is:22.200000

Completion time is:
```

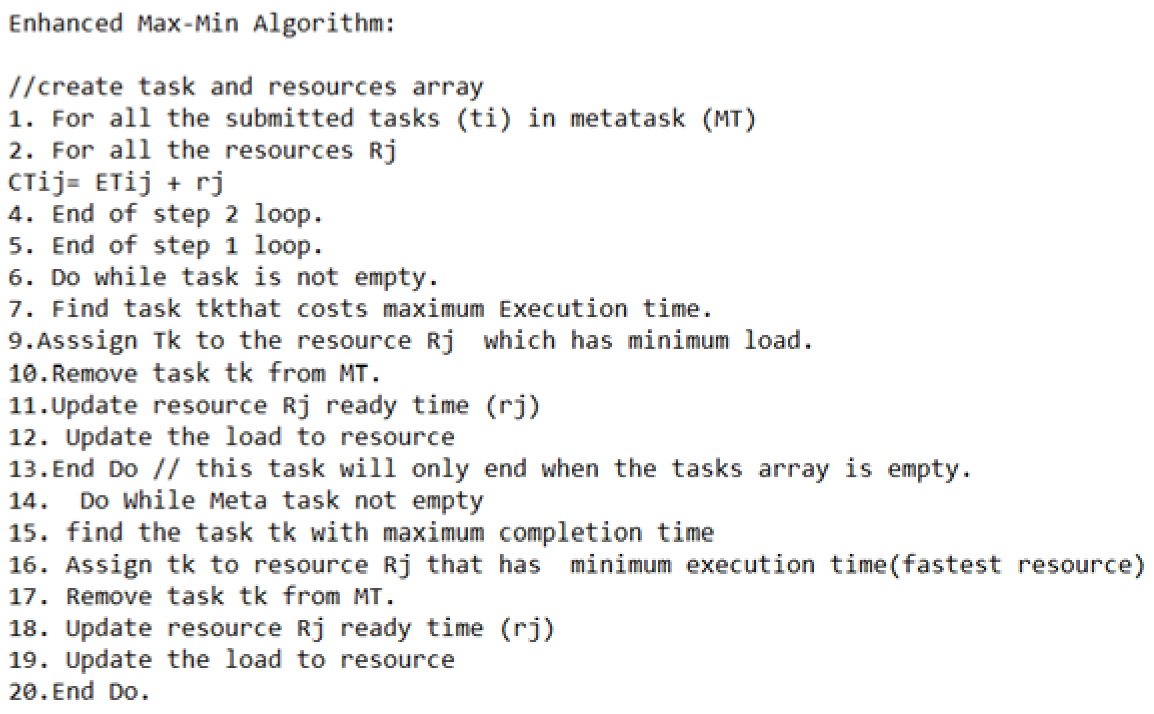
Fig 8.1: Max-Min Output



```
Remaing resources is:55.500000

Completion time is:
217.000000 217.000000 210.000000
328.000000 328.000000 321.000000
260.500000 260.500000 253.500000
155.500000 155.500000 148.500000
Remaing resources is:117.000000
Remaing resources is:55.500000
Remaing resources is:55.500000

Completion time is:
217.000000 217.000000 210.000000
260.500000 260.500000 253.500000
155.500000 155.500000 148.500000
Remaing resources is:55.500000
Remaing resources is:55.500000

Completion time is:
217.000000 217.000000 210.000000
155.500000 155.500000 148.500000
Remaing resources is:55.500000

Completion time is:
155.500000 155.500000 148.500000
priyansh@priyansh-VirtualBox:~/Documents$
```

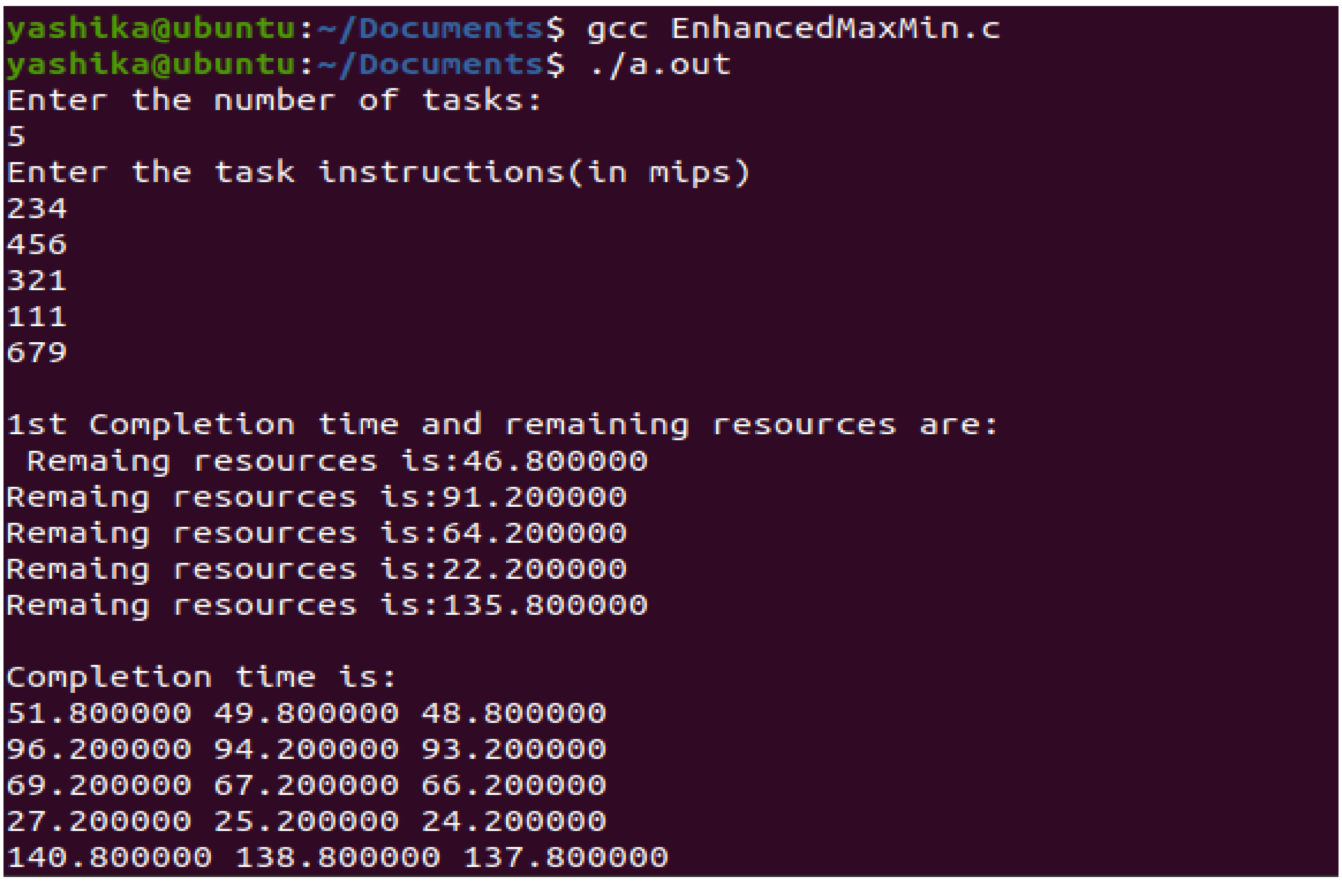
Fig 8.2: Max-Min Output

6.1.3 Enhanced Max-Min Algorithm (Proposed):



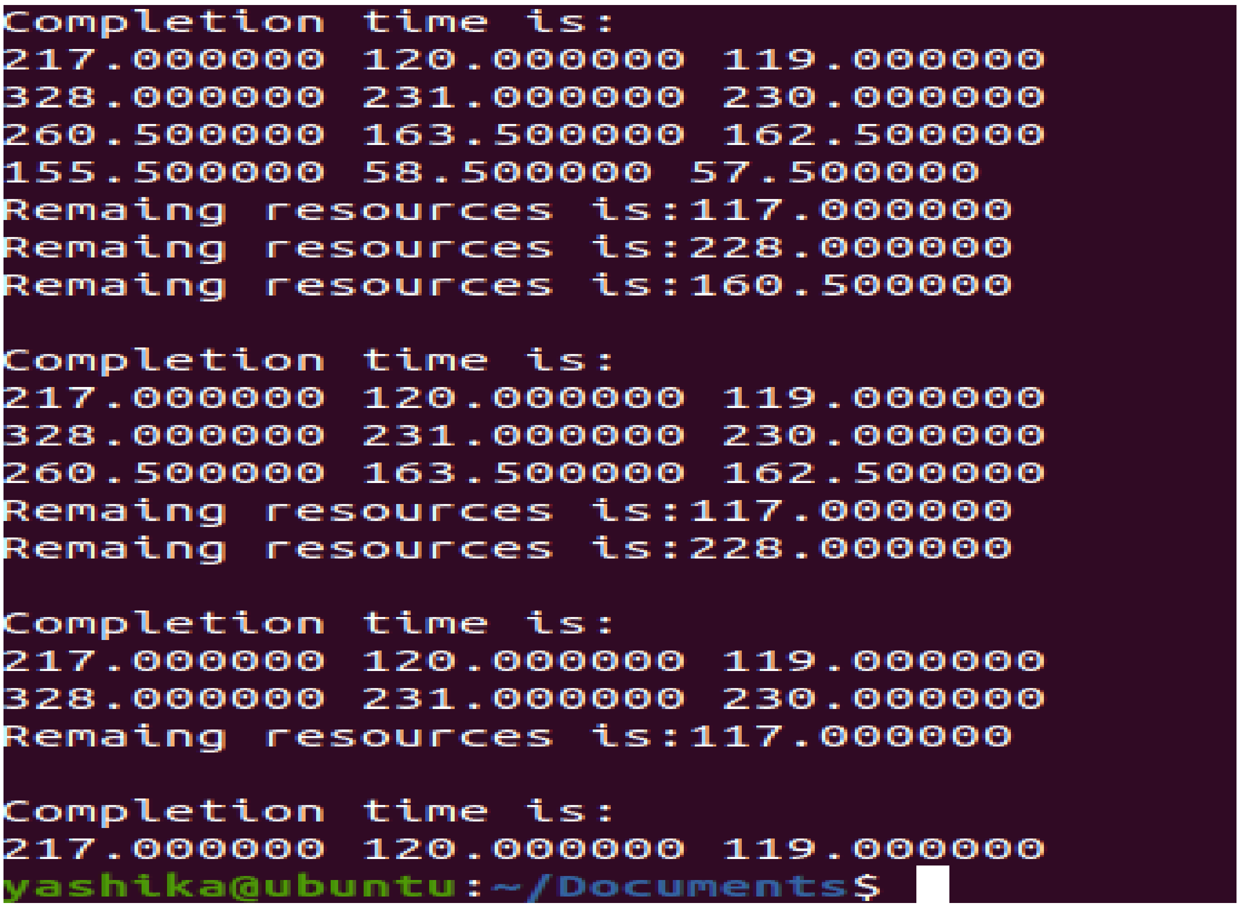
```
Enhanced Max-Min Algorithm:

//create task and resources array
1. For all the submitted tasks (ti) in metatask (MT)
2. For all the resources Rj
CTij= ETij + rj
4. End of step 2 loop.
5. End of step 1 loop.
6. Do while task is not empty.
7. Find task tkthat costs maximum Execution time.
9.Asssign Tk to the resource Rj  which has minimum load.
10.Remove task tk from MT.
11.Update resource Rj ready time (rj)
12. Update the load to resource
13.End Do // this task will only end when the tasks array is empty.
14.  Do While Meta task not empty
15. find the task tk with maximum completion time
16. Assign tk to resource Rj that has  minimum execution time(fastest resource)
17. Remove task tk from MT.
18. Update resource Rj ready time (rj)
19. Update the load to resource
20.End Do.
```

Fig 9: Enhanced Max-Min Algorithm



```
yashika@ubuntu:~/Documents$ gcc EnhancedMaxMin.c
yashika@ubuntu:~/Documents$ ./a.out
Enter the number of tasks:
5
Enter the task instructions(in mips)
234
456
321
111
679

1st Completion time and remaining resources are:
 Remaing resources is:46.800000
Remaing resources is:91.200000
Remaing resources is:64.200000
Remaing resources is:22.200000
Remaing resources is:135.800000

Completion time is:
51.800000 49.800000 48.800000
96.200000 94.200000 93.200000
69.200000 67.200000 66.200000
27.200000 25.200000 24.200000
140.800000 138.800000 137.800000
```

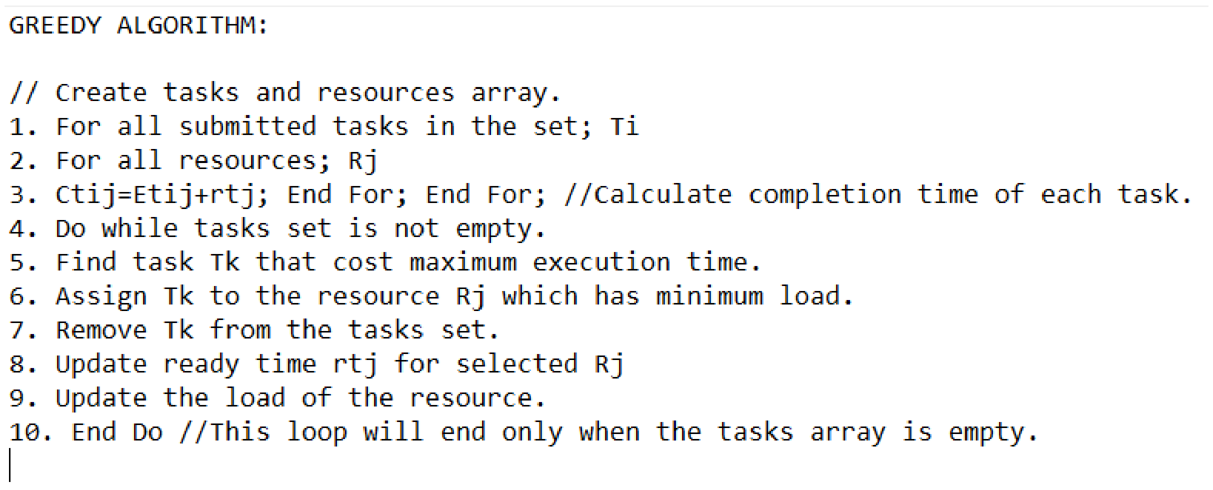
Fig 10.1: Enhanced Max-Min Output



```
Completion time is:
217.000000 120.000000 119.000000
328.000000 231.000000 230.000000
260.500000 163.500000 162.500000
155.500000 58.500000 57.500000
Remaing resources is:117.000000
Remaing resources is:228.000000
Remaing resources is:160.500000

Completion time is:
217.000000 120.000000 119.000000
328.000000 231.000000 230.000000
260.500000 163.500000 162.500000
Remaing resources is:117.000000
Remaing resources is:228.000000

Completion time is:
217.000000 120.000000 119.000000
328.000000 231.000000 230.000000
Remaing resources is:117.000000

Completion time is:
217.000000 120.000000 119.000000
yashika@ubuntu:~/Documents$
```

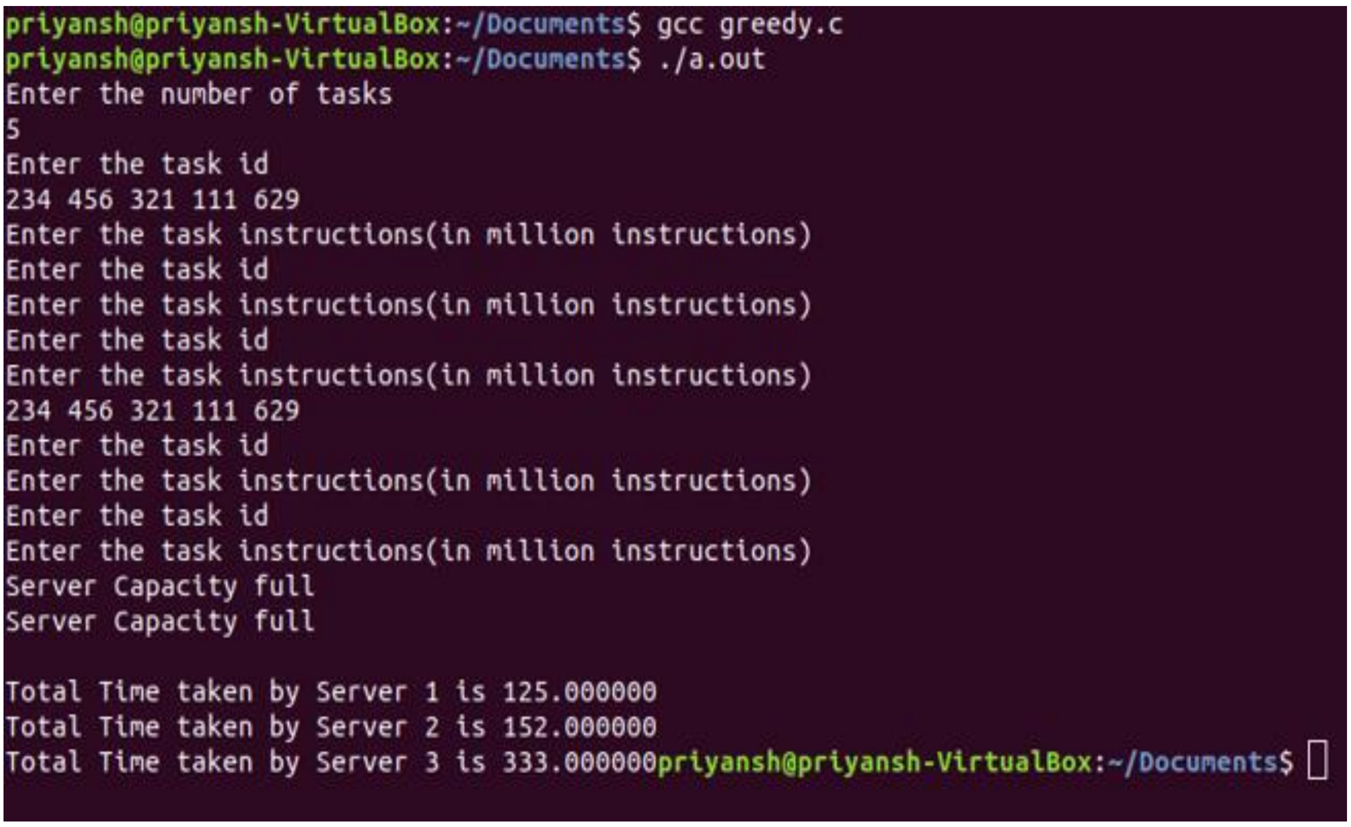
Fig 10.2: Enhanced Max-Min Output

6.1.4 Greedy Algorithm (NP Hard Problem):



```
GREEDY ALGORITHM:

// Create tasks and resources array.
1. For all submitted tasks in the set; Ti
2. For all resources; Rj
3. Ctij=Etij+rtj; End For; End For; //Calculate completion time of each task.
4. Do while tasks set is not empty.
5. Find task Tk that cost maximum execution time.
6. Assign Tk to the resource Rj which has minimum load.
7. Remove Tk from the tasks set.
8. Update ready time rtj for selected Rj
9. Update the load of the resource.
10. End Do //This loop will end only when the tasks array is empty.
```

Fig 11: Greedy Algorithm



```
priyansh@priyansh-VirtualBox:~/Documents$ gcc greedy.c
priyansh@priyansh-VirtualBox:~/Documents$ ./a.out
Enter the number of tasks
5
Enter the task id
234 456 321 111 629
Enter the task instructions(in million instructions)
Enter the task id
Enter the task instructions(in million instructions)
Enter the task id
Enter the task instructions(in million instructions)
234 456 321 111 629
Enter the task id
Enter the task instructions(in million instructions)
Enter the task id
Enter the task instructions(in million instructions)
Server Capacity full
Server Capacity full

Total Time taken by Server 1 is 125.000000
Total Time taken by Server 2 is 152.000000
Total Time taken by Server 3 is 333.000000priyansh@priyansh-VirtualBox:~/Documents$ 
```

Fig 12: Greedy Output

6.2 Comparison of Task Scheduling Algorithms

With respect to **Makespan**: -

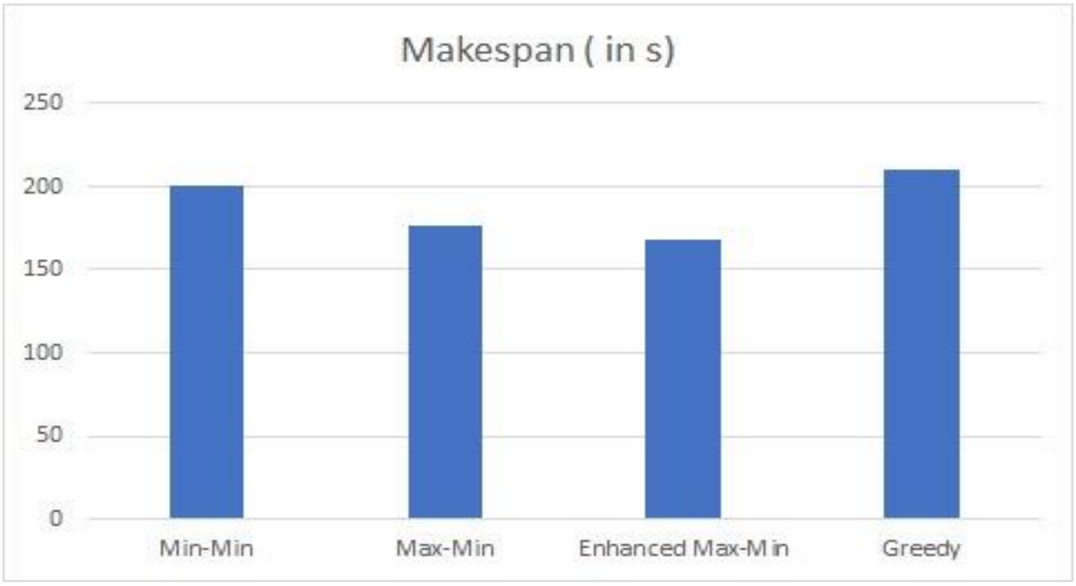


Fig 13: Graph with input (No. of Tasks=5)

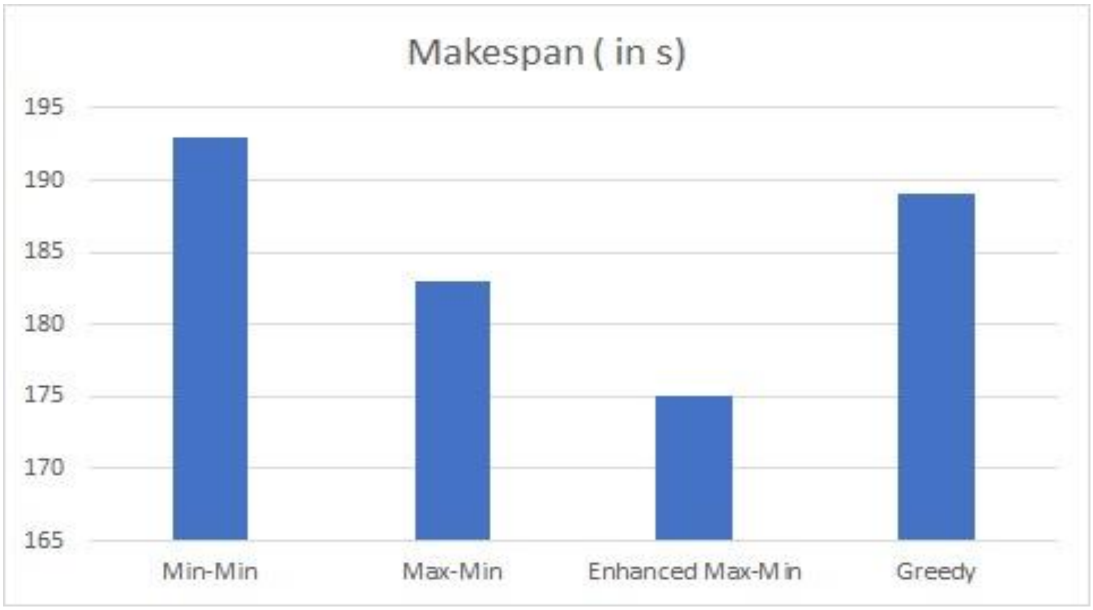


Fig 14: Graph with Input (No. of Tasks=10)

7. System Requirements

Hardware Interface:

- Minimum RAM requirement for proper functioning is 4 GB.
- Required Linux Operating System.
- Processor required of i3 5th generation or above.

Software Interface:

- This system is developed in C programming language.
- C editor and compiler

8. Pert Chart

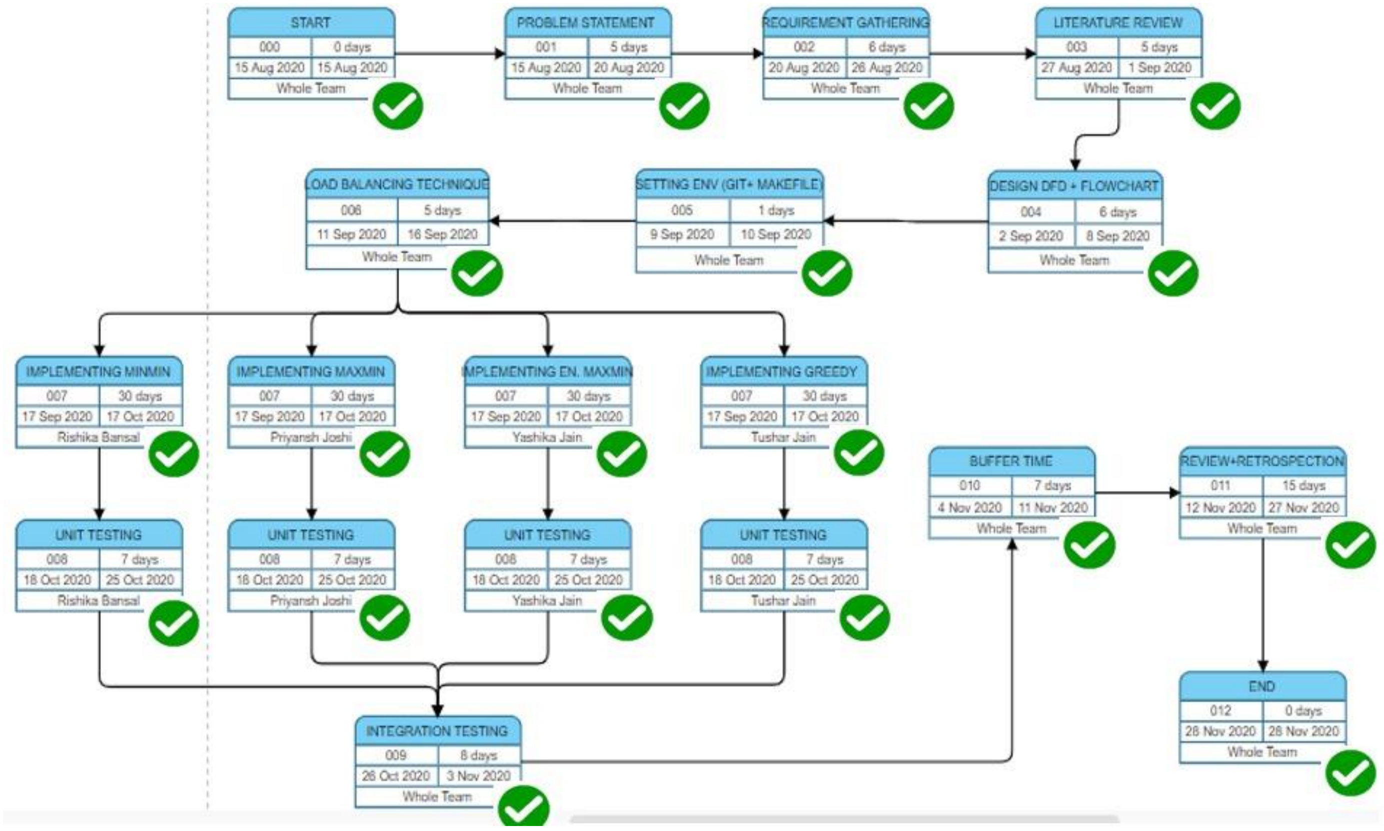


Fig 15: PERT Chart for the Project

9. Code Link

GitHub - Priyansh247Joshi/Server-Load-Balancing: Minor 1 Project

10. REFERENCES

[1]J. Kok Konjaang, Y. Maipan-uku, Kumangkem Kennedy Kubuga
"An Efficient Max-Min Resource Allocator and Task Scheduling Algorithm in Cloud Computing Environment" International Journal of Computer Applications (0975 – 8887) Volume 142 – No.8, May 2016.

[2]Upendra Bhoi1, Purvi N. Ramanuj2, "Enhanced Max-min Task Scheduling Algorithm in Cloud Computing" International journal of Application or Innovation in Engineering of Management, Web Site: www.ijaiem.org Volume 2, Issue 4, April 2013.

[3]J. Kok Konjanag, Fahrul Hakim Ayob, Abdullah Muhammed, "An Optimized Max-Min Scheduling In Cloud Computing" Journal of Theoretical and Applied Information Technology 15th May 2017. Vol.95. No 9 © 2005 – ongoing JATIT & LL

[4] Gaurang Patel, Rutvik Mehta, Upendra Bhoi "Enhanced Load Balanced Min-Min algorithm for Static Meta Task Scheduling in Cloud Computing" 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

[5] https://ars.els-cdn.com/content/image/1-s2.0-S1319157817303361-gr2.jpg (9/19/2020)

[6] Neha Sharma, Swati Arti, Dr. Sanjay Tyagi "A Comparative Analysis of Min-Min and Max-Min Algorithms based on the Makespan Parameter" International Journal of Advanced Research in Computer Science, Volume 8, No. 3, March – April 2017

[7] https://miro.medium.com/max/695/0*iDT-e3lTcGR0CSy8. (9/19/2020)

Report verified by

Ms. Avita Katal

Assistant Professor-Senior Scale
Department of Virtualization
School of Computer Science

Project Guide
(Ms. Avita Katal)

Dr. Deepshikha Bhargava

Head of Department
Department of Virtualization
School of Computer Science

HOD
(Dr. Deepshikha Bhargava)