Journal of Engineering and Applied Sciences

MEDWELL PUBLICATIONS SINCE 2000

Development of Algorithm for QoS Model in Cloud Computing

[1]Mohammed Abdullah Alfadli, [1]Mohamad Fauzan Noordin and [2]M.D. Masum Billah
[1]Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, Malaysia
[2]Universiti Kuala Lumpur, Malaysia

Key words: Trusted, QoS Model, algorithm, cloudlet, Saudi Arabia

Corresponding Author:
Mohammed Abdullah Alfadli
Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, Malaysia

Page No.: 264-272
Volume: 16, Issue 8, 2021
ISSN: 1816-949x
Journal of Engineering and Applied Sciences
Copy Right: Medwell Publications

Abstract: In this study, we will develop an algorithm for the QoS Model. It starts with studying four algorithms and explain the experimental setup for them. Then, we simulate them, get the results and discuss results for all algorithms. After that, we will develop our new algorithm for the QoS Model. Then, we simulate the new enhanced algorithm and compare to other algorithms.

INTRODUCTION

According to NIST, there are five characteristics for Cloud computing as follows[1].

On-demand self-service: Customer can get the services automatically without human interaction from service provider.

Broad network access: Services can be accessed by heterogeneous platforms (e.g., mobile phones, PC's, laptops and tablets).

Resource pooling: The resources are shared between users, resources include storage, memory and network, etc.

Rapid elasticity: Capabilities can be scaled up and down easily and quickly.

Measured service: All services are metered to allow pay as you go for customers. Also, Cloud Computing Service Delivery Models can be categorized according to NIST[1-4].

Software as a Service (SaaS): Customer can use Software and application directly from Cloud provider.

Platform as a Service (PaaS): This is mainly for developers where they will have infrastructure ready and they can apply and use their own applications.

Infrastructure as a Service (IaaS): This is only the infrastructure, so, the customer (mainly system administrators and network engineers) can use infrastructure which include network and server devices and customer can build their virtualization solution and then operate their own software on top of that.

Furthermore, Cloud Computing Deployment Models can be categorized according to NIST[1].

Private cloud: The cloud is only used by a single organization.

Community cloud: The cloud is used by a specific community who shared same interest or mission.

Public cloud: The cloud is open to be used by public.

Hybrid cloud: The cloud is consists of two or more different cloud types (private, community, or public).

The main issue behind the slow adoption is the security and privacy issues. In a recent study about higher education institutions adoption of Cloud Computing in Saudi Arabia, the researchers stated that "Non-adopters highlighted a degree of privacy concern about cloud computing adoption. There is a trust issue with respect to storing intellectual property assets"[5]. In a previous study that investigated the factors affecting cloud computing adoption decision in Saudi Arabia, it has been found that security concerns are a barrier for cloud computing adoption[6]. Another study by Alkhater *et al*.[7] concluded that "It is interesting to note that security, privacy and trust issues were the big concerns for most organizations participating in this study and were the main reasons behind their decisions not to adopt cloud services." Also, a study about Factors Affecting the Adoption of Cloud Computing in the Government Sector in Saudi Arabia, stated that "95.26% of the respondents also perceived service quality and security as an important factor in the adoption of cloud computing" The SAUDI Communications and Information Technology Commission (CITC) did many studies in this regards. All previous CITC reports on the ICT status in Saudi Arabia have showed that security concerns are considered as one of the key barriers to the adoption of many ICT services including cloud computing[8].

On the other hand, many studies conclude that cloud Computing can save the costs in Saudi Arabia. A study about Healthcare in Saudi Arabia stated that "Since, financial issues are affecting E-health projects in the country, Cloud Computing can offer economic savings by decreasing the initial and operational costs of E-health projects in Saudi hospitals"[9]. Also, in a report by Ministry of Communications and Information Technology in Saudi Arabia, it says that Saudi companies adoption of cloud computing in all services will contribute to the reduction of costs and expenses by >50%[8]. Furthermore, in a previous study about the impact of cloud computing on Saudi organizations, the researchers concluded that cloud computing can be useful for cost saving in Saudi Arabia, and one the senior management in the selected telecom company said "Decreasing cost comes from sharing resources among different locations and unifying the management of hardware and software"[10].

"Currently, there is no comprehensive framework for cloud controls that enables potential cloud customers to confidently assess and verify a cloud provider's controls and environments. This lack of reliable control framework has opened a trust gap between cloud providers and customers and that has impeded the advance of cloud computing"[11].

"There is a need for solutions that address trust and security across solutions derived from combining dedicated and shared infrastructures" and "No comprehensive framework exists to describe the business/mission needs and validate compliance of the entire solution set against open standards".

The SAUDI Communications and Information Technology Commission (CITC) has stated in a recent report about ICT in Saudi Arabia that many local Service providers don not provide QoS agreements and if they provide, they don not usually adhere to it and definitely this has a direct impact in customers trust on adoption of new services[8].

STUDY ALGORITHMS

We will study four algorithms, namely, First Come First Serve (FCFS), Round Robin (RR), Max-Min, Min-Min.

First Come First Served (FCFS): It's a simple and basic scheduling algorithm. This algorithm will process jobs based on the arrival order. The first arrived job will be fulfilled first, once its completed, the next arrived job will start processing and so on.

This algorithm also called First In First Out (FIFO), as shown in Fig. 1, P1 is the first in queue, so, it will be processed first for three time units, after its finished, P2 will start the processing for four time units, and so on, till it process the last one which is P6 for five time units. FCFS provides a simple method and straightforward scheduling but it will only process the next job when the current job is completed and if the current job will take a long-time processing, the waiting time will be very high for the next jobs.

Round Robin (RR): Round Robin algorithm distributes CPU time equally between the waiting jobs till its completing processing of all jobs. This time called quantum. if the time is set too short, it will require more switching between processes which will make the whole processing is slow while if it is set to a long time, it will look likes first Come first Serve. As shown in Fig. 2, Round Robin is having quantum equal to two units of time for all jobs. In the first round, P1 will be processed partially for 2 units of time and 1 unit wait for the next round. P2 will be processed partially for 2 units of time

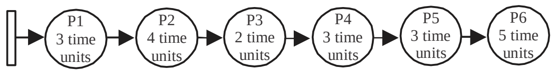


Fig. 1: Diagram representing FCFS algorithm[12]

and 2 units wait for the next round. P3 will be processed for 2 units of time, and its completed, since, all the needed time is 2 units. P4 and P5, will be processed partially for 2 units of time and 1 unit wait for the next round. P6 and P7 will be processed partially for 2 units of time and 3 and 2 units respectively will wait for the next round. P8 will be processed for 2 units of time and its completed since all the needed time is 2 units. Then, the next round will start with same scenario, so, P1, P2, P4, P5, and P7 will be completed in the second round. P6 will be completed in the third round[12].

Max-Min: Max-Min algorithm checks and identify the jobs with highest processing time to complete them. And then start processing jobs with the highest needed processing time (Algorithm 1).

Algorithm 1: For Max-Min[13]: Pseudo-Code for Max-Min algorithm

```
1: for 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 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 tasks
8:            find the maximum completion time of the task and virtual machine that obtains it
9:    end for
10:   find the task $t_p$ with maximum completion time
11:     assign task $t_p$ to the virtual machine that gives the maximum completion time
12:    delete task $t_p$ from pull of unscheduled tasks
13:     update the ready time of the machine that gives the maximum completion time
14:    end for
```

The algorithm will do two parts: calculation of the completion time of all tasks or jobs and then finding the job with maximum completion time and assign it to the virtual machine that can handle it. As depicted in Fig. 2, the first part has two for loops, M represents the number of jobs to be scheduled, N represents the number of virtual machines, C_{ij} represents the completion time of a job, E_{ij} represents the execution time of the job and R_j represents the ready time of a job i on virtual machine j. To calculate the completion time of a job (C_{ij}), we add the execution time of that task (E_{ij}) to the ready time of the task (R_j), this will be repeated N times and then repeated M times.

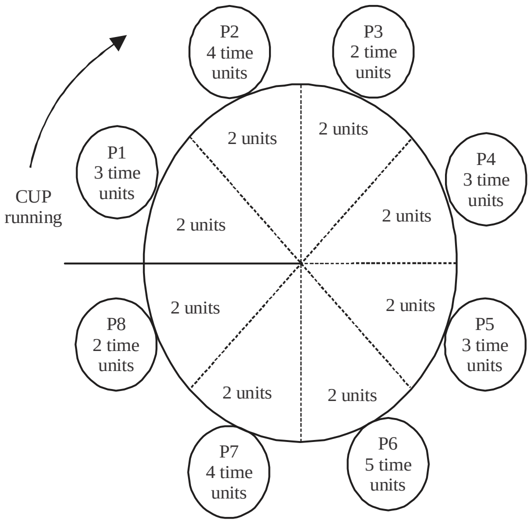


Fig. 2: Diagram representing Round Robin algorithm[12]

After that, another loop will start till scheduling all tasks or jobs. All unscheduled jobs will be checked to find the job with maximum completion time (t_p) and the suitable virtual machine for it.

Then Assign the job (t_p) to the virtual machine that gives maximum completion time and delete it from the unscheduled jobs. Then update the ready time of the machine that gives the maximum completion time. This will be repeated to schedule all jobs.

Min-Min: Min-Min algorithm will check for the needed processing time to complete each job in the queue, and start processing jobs which have lowest processing time (Algorithm 2).

Algorithm 2; Pseudo-Code for Min-Min algorithm:

```
1: for 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 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 tasks
8:            find the maximum completion time of the task and virtual machine that obtains it
9:    end for
10:   find the task $t_p$ with maximum completion time
11:     assign task $t_p$ to the virtual machine that gives the maximum completion time
12:    delete task $t_p$ from pull of unscheduled tasks
13:     update the ready time of the machine that gives the maximum completion time
14:    end do
```

The algorithm will do two parts: calculation of the completion time of all tasks or jobs and then finding the job with minimum completion time and assign it to the virtual machine most suitable for it. As depicted in Algorithm 1, the first part has two for loops, M represents the number of jobs to be scheduled, N represents the number of virtual machines, C_{ij} represents the completion time of a job, E_{ij} represents the execution time of the job and R_j represents the ready time of a job i on virtual machine j.

To calculate the completion time of a job (C_{ij}), we add the execution time of that task (E_{ij}) to the ready time of the task (R_j), this will be repeated N times and then repeated M times.

After that, another loop will start till scheduling all tasks or jobs. All unscheduled jobs will be checked to find the job with minimum completion time (t_p) and the suitable virtual machine for it. Then Assign the job (t_p) to the virtual machine that gives minimum completion time and delete it from the unscheduled jobs. Then update the ready time of the machine that gives the minimum completion time. This will be repeated to schedule all jobs.

EXPERIMENTAL SETUP

The experiment was done by using CloudSim tool. The experiment starts when an end users sends many jobs which have different configurations to simulate the heterogeneous environment. The goal is to simulate the availability factor using the four algorithms. All these algorithms are focused on how to assign cloudlets or jobs to Virtual Machines (VMs). Also, we have used a computer with 1.70 GHz CPU (Intel i3) and 4 GB RAM to do the simulation.

In every experiment, there will be a group of 10 batches. First Batch starts with 500 jobs and then each batch followed is incremented by 500 jobs, till the last batch which becomes 5000 jobs. During each batch, all jobs are submitted simultaneously. The cloudlets length are random and every cloudlet has a classification level, 1,2 or 3 where 1 denotes lowest priority and 3 denotes the highest priority. These classification levels are created randomly for every cloudlet. This classification is based on official regulation released on 6/2/2018 and enforced on 8/3/2018. It was released by SAUDI Communications and Information Technology Commission (CITC), the Commission is the only authority in charge of regulating Information and Communications Technology (ICT) sector in Saudi Arabia, which includes regulations on Cloud Computing.

To measure availability in Cloudsim and since, Cloudsim will always process all cloudlets in queue till finished, we have developed a method in Java to run the algorithms for a specific time and calculate how many successful cloudlets have been processed by an algorithm in this period of time. The same period of time is used by all algorithms, so, we can compare them later. We have found after extensive experiments that the suitable period of time is:

- 50,000 sec for 500 cloudlets
- 100,000 sec for 1000 cloudlets
- 150,000 sec for 1500 cloudlets
- 200,000 sec for 2000 cloudlets
- 250,000 sec for 2500 cloudlets
- 300,000 sec for 3000 cloudlets
- 350,000 sec for 3500 cloudlets
- 400,000 sec for 4000 cloudlets
- 450,000 sec for 4500 cloudlets
- 500,000 sec for 5000 cloudlets

It is noticed that both the time and number of cloudlets increased in the same percentage to maintain consistency and fairness in all batches of cloudlets, and this is applied for all algorithms.

We will create 4 Virtual machines (VMs) to represent 4 locations in Saudi Arabia. Every VM has 10 MIPS (million instructions per Second), 512 MB RAM, 1 Processor.

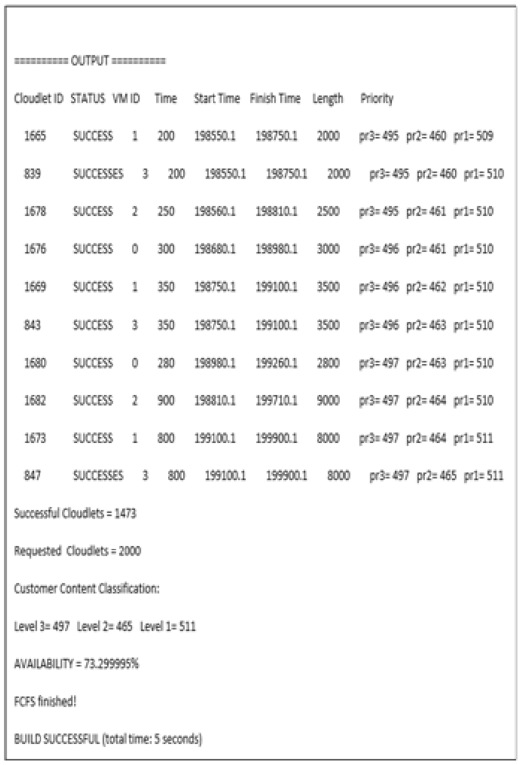
FCFS: As shown in Fig. 3, a sample result of the simulation is shown when the number of requested cloudlets is 2000. FCFS is simply processing the cloudlets by their order in the queue regardless of other parameters like the length of cloudlets. Please note that we have 4 virtual machines, so for example, VM ID No. 1 will process cloudlet No. 1665, then it will process cloudlet No.1669 and then cloudlet No. 1673.

Figure 4 is showing the availability percentage of FCFS algorithm after processing the ten batches, starting from 500 cloudlets till 5000 cloudlets. FCFS is having availability around 70% and it went up and down based on the cloudlets processed.

Round robin: As shown in Fig. 5, a sample result of the simulation is shown when the number of requested cloudlets is 2000. Round Robin will distribute the processing time equally between the cloudlets, so it will finish the similar cloudlets at the same time.

Figure 6 is showing the availability percentage of Round Robin algorithm after processing the ten batches, starting from 500 cloudlets till 5000 cloudlets. Round Robin is having availability around 65% and it went up and down based on the cloudlets processed.

Max Min: As shown in Fig. 7, a sample result of the simulation is shown when the number of requested



```
========== OUTPUT ==========

Cloudlet ID  STATUS  VM ID   Time    Start Time   Finish Time   Length    Priority

  1665      SUCCESS     1    200    198550.1    198750.1    2000    pr3= 495  pr2= 460  pr1= 509

  839       SUCCESSES     3    200    198550.1    198750.1    2000    pr3= 495  pr2= 460  pr1= 510

  1678      SUCCESS     2    250    198560.1    198810.1    2500    pr3= 495  pr2= 461  pr1= 510

  1676      SUCCESS     0    300    198680.1    198980.1    3000    pr3= 496  pr2= 461  pr1= 510

  1669      SUCCESS     1    350    198750.1    199100.1    3500    pr3= 496  pr2= 462  pr1= 510

  843       SUCCESS     3    350    198750.1    199100.1    3500    pr3= 496  pr2= 463  pr1= 510

  1680      SUCCESS     0    280    198980.1    199260.1    2800    pr3= 497  pr2= 463  pr1= 510

  1682      SUCCESS     2    900    198810.1    199710.1    9000    pr3= 497  pr2= 464  pr1= 510

  1673      SUCCESS     1    800    199100.1    199900.1    8000    pr3= 497  pr2= 464  pr1= 511

  847       SUCCESSES     3    800    199100.1    199900.1    8000    pr3= 497  pr2= 465  pr1= 511

Successful Cloudlets = 1473

Requested  Cloudlets = 2000

Customer Content Classification:

Level 3= 497   Level 2= 465   Level 1= 511

AVAILABILITY = 73.299995%

FCFS finished!

BUILD SUCCESSFUL (total time: 5 seconds)
```

Fig. 3: Sample result of Simulation at 2000 cloudlets for FCFS

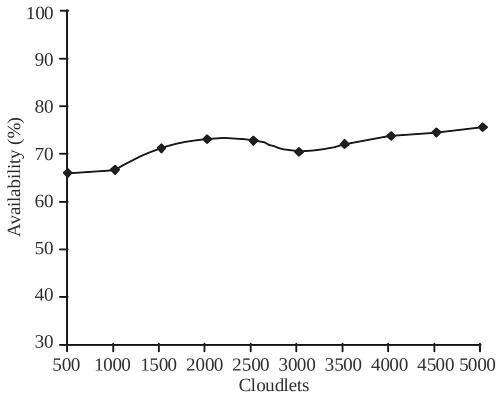
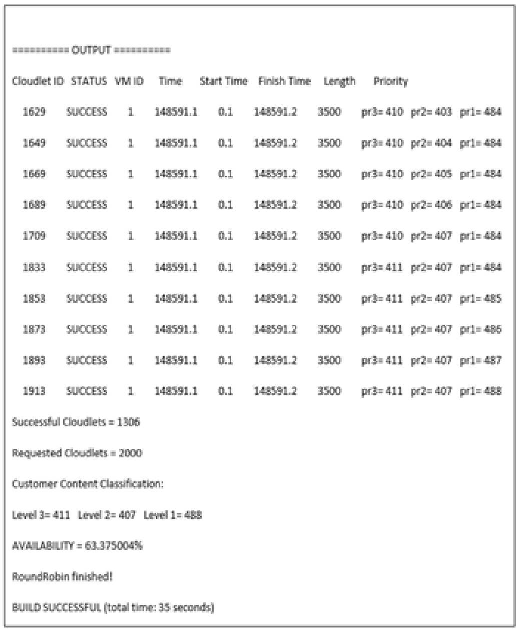


Fig. 4: Availability result related to number of cloudlets for FCFS

cloudlets is 2000. Max Min will process the longer cloudlets first, and this means the successful number of cloudlets will be low because longer cloudlets will make the processor busy for longer time.

Figure 8 is showing the availability percentage of MaxMin algorithm after processing the ten batches, starting from 500 cloudlets till 5000 cloudlets. MaxMin is having availability around 45% and it went up and down based on the cloudlets processed. Note that the batch of 1000 cloudlets, the availability is low and this is because the random cloudlets were longer than other batches.



```
========== OUTPUT ==========

Cloudlet ID  STATUS  VM ID    Time     Start Time   Finish Time    Length     Priority

  1629      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 410  pr2= 403  pr1= 484

  1649      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 410  pr2= 404  pr1= 484

  1669      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 410  pr2= 405  pr1= 484

  1689      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 410  pr2= 406  pr1= 484

  1709      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 410  pr2= 407  pr1= 484

  1833      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 411  pr2= 407  pr1= 484

  1853      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 411  pr2= 407  pr1= 485

  1873      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 411  pr2= 407  pr1= 486

  1893      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 411  pr2= 407  pr1= 487

  1913      SUCCESS    1    148591.1     0.1     148591.2     3500     pr3= 411  pr2= 407  pr1= 488

Successful Cloudlets = 1306

Requested Cloudlets = 2000

Customer Content Classification:

Level 3= 411   Level 2= 407   Level 1= 488

AVAILABILITY = 63.375004%

RoundRobin finished!

BUILD SUCCESSFUL (total time: 35 seconds)
```

Fig. 5: Sample result of Simulation at 2000 cloudlets for Round Robin

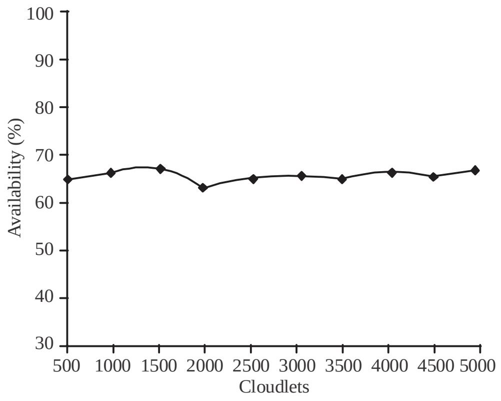
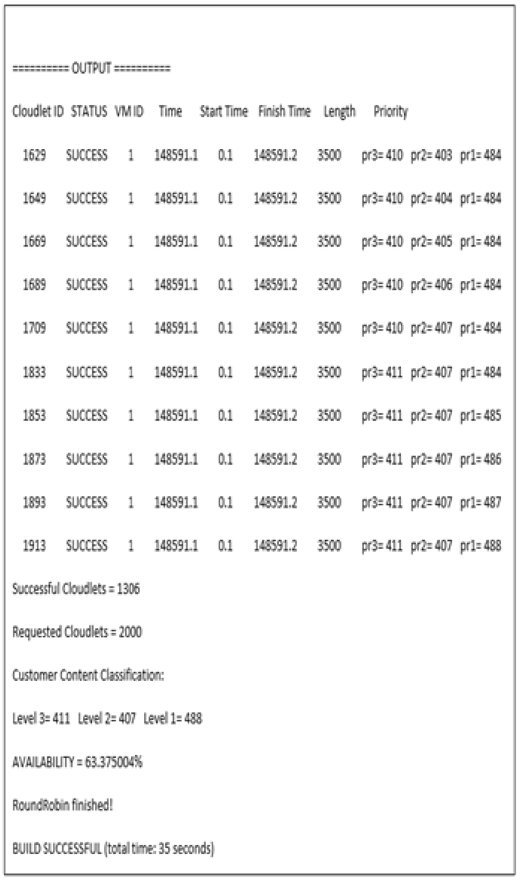


Fig. 6: Availability result related to number of cloudlets for Round Robin

Min Min: As shown in Fig. 9, a sample result of the simulation is shown when the number of requested cloudlets is 2000. Min Min will process the short cloudlets first, and this means the successful number of cloudlets will be high, because short cloudlets will take less processing time (Fig. 10).

Figure 11 is showing the availability percentage of Min Min algorithm after processing the ten batches,



```
========== OUTPUT ==========

Cloudlet ID  STATUS  VM ID    Time     Start Time   Finish Time   Length    Priority

  1629     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 410  pr2= 403  pr1= 484

  1649     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 410  pr2= 404  pr1= 484

  1669     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 410  pr2= 405  pr1= 484

  1689     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 410  pr2= 406  pr1= 484

  1709     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 410  pr2= 407  pr1= 484

  1833     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 411  pr2= 407  pr1= 484

  1853     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 411  pr2= 407  pr1= 485

  1873     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 411  pr2= 407  pr1= 486

  1893     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 411  pr2= 407  pr1= 487

  1913     SUCCESS    1     148591.1    0.1      148591.2     3500     pr3= 411  pr2= 407  pr1= 488

Successful Cloudlets = 1306

Requested Cloudlets = 2000

Customer Content Classification:

Level 3= 411   Level 2= 407   Level 1= 488

AVAILABILITY = 63.375004%

RoundRobin finished!

BUILD SUCCESSFUL (total time: 35 seconds)
```

Fig. 7: Sample result of simulation at 2000 cloudlets for Max Min

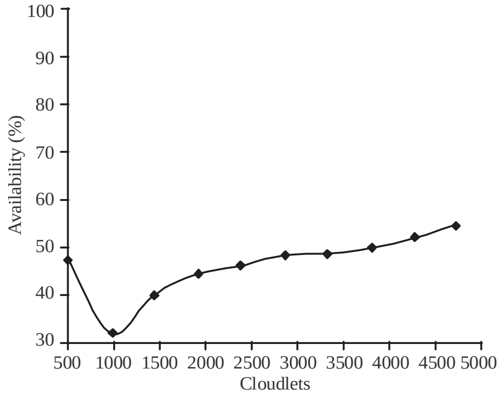


Fig. 8: Availability result related to number of cloudlets for Max Min

starting from 500 cloudlets till 5000 cloudlets. Min Min is having availability around 91% and it went up and down based on the cloudlets processed.

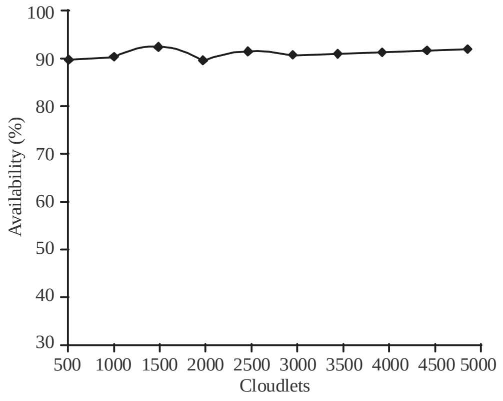
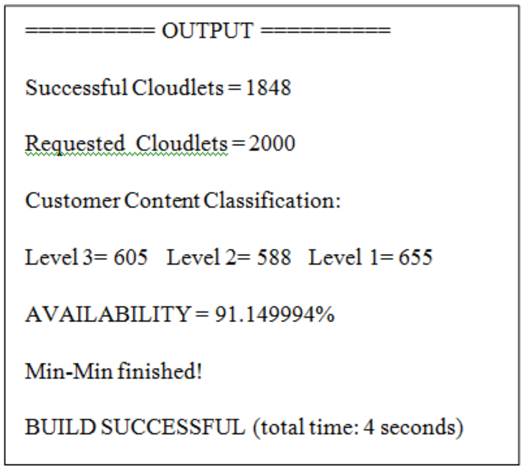


Fig. 9: Sample result of simulation at 2000 cloudlets for Min Min



```
========== OUTPUT ==========

Successful Cloudlets = 1848

Requested  Cloudlets = 2000

Customer Content Classification:

Level 3= 605   Level 2= 588   Level 1= 655

AVAILABILITY = 91.149994%

Min-Min finished!

BUILD SUCCESSFUL (total time: 4 seconds)
```

Fig. 10: Sample result of simulation

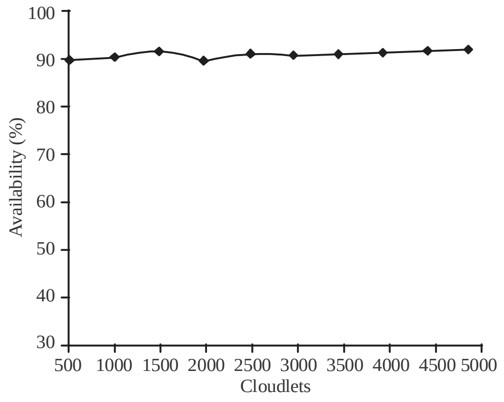


Fig. 11: Availability result related to number of cloudlets for Min Min

Comparison between the four algorithms: Figure 12 shows the availability percentage of all the four

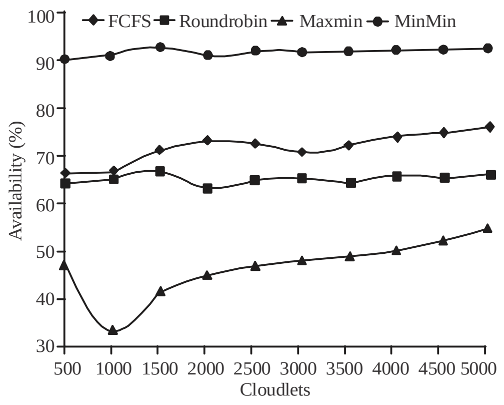


Fig. 12: Availability result related to number of cloudlets for all four algorithms

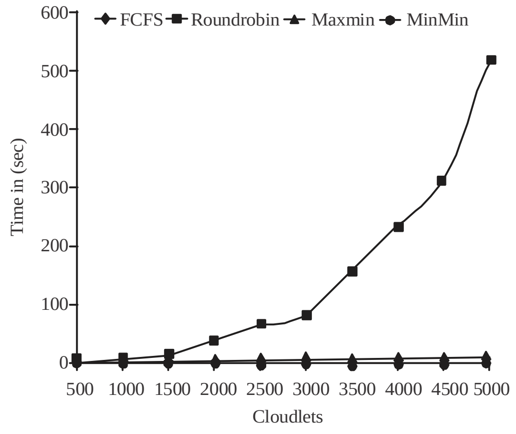


Fig. 13: Simulation time for all four algorithms

algorithms, namely FCFS, Round Robin, MaxMin, and MinMin. It is clear that MinMin has the best availability compared to other algorithms, and that is because its processing the short cloudlets first, and hence completing more cloudlets successfully. MaxMin is the worst algorithm compared to other algorithms since its processing the longer cloudlets first and this will make the successful cloudlets the lowest. FCFS has a better availability percentage than Round Robin but they are near each other.

We have noticed after doing the experiments that Round Robin algorithm is taking a very long time in simulation compared to FCFS, Max Min and Min Min. Figure 13 is showing the simulation time for all the four algorithms. It is increasing when the number of cloudlets increased, this is because the algorithm is distributing the processing time between all the cloudlets in the queue, so, if we have 5000 cloudlets, this will take much more switching time than if we have only 500 cloudlets. And the algorithm will continue distributing the time for all cloudlets, till cloudlets finish processing.

DEVELOPING A NEW ALGORITHM

Since, Min Min is the best algorithm among the four algorithms studied before, namely, FCFS, Round Robin, and MaxMin. We will develop a new algorithms based on MinMin algorithm and based on the new Regulations in Saudi Arabia that enforced any cloud provider to have three levels of quality of service. We have called this new algorithm as Enhanced Multi Level MinMin Algorithm (EMLMA) as shown in Algorithm 3.

Algorithm 3; Pseudo-code for EMLMA algorithm:

```
for i = 1 to M || M denotes the number of tasks to be scheduled
for j = 1 to N || N denotes the number of virtual machine
    $C_{ij} = E_{ij}+R_j$
                || $C_{ij}$ denotes the completion time of task
                || $E_{ij}$ denotes execution time of task
                || Rj denotes the ready time of task i on virtual machine j
  end for
end for
do until all the unscheduled tasks with level 3 are exhausted
for each unscheduled task
    Find the minimum completion time of task and virtual machine that
obtains it
end for
find the task $t_p$ with earliest completion time
assign task $t_p$ to the virtual machine that gives the minimum completion
time
delete task $t_p$ from pull of unscheduled tasks
Increase the counter, successful level 3 tasks by 1
update the ready time of the machine that gives the minimum completion
time
end do
do until all the unscheduled tasks with level 2 are exhausted
for each unscheduled task
find the minimum completion time of the task and virtual machine that
otains it
end for
find the task $t_p$ with earliest completion time
assign task $t_p$ to the virtual machine that gives the minimum completion
time
delete task $t_p$ from pull of unscheduled tasks
Increase the counter, successful level 3 tasks by 1
update the ready time of the machine that gives the minimum completion
time
end do
do untill all the  unscheduled tasks with level 2 are exhausted
for each unscheduled task
find the minimum completion time of the task and virtual machine that
obtains it
end for
find the task $t_p$ with earliest completion time
assign task $t_p$ to the virtual machine that gives the minimum completion
time
delete task $t_p$ from pull of unscheduled tasks
Increase the counter, successful level 3 tasks by 1
update the ready time of the machine that gives the minimum completion
time
end do
Weight of tasks = (Level 3 tasks*3)+(Level 2 tasks*2)+(Level 1*1)
Availability     = (Weight of tasks/)(Total number of tasks*3))*100
```

This algorithm will do the steps similar to normal Min Min but will process the jobs with highest level of

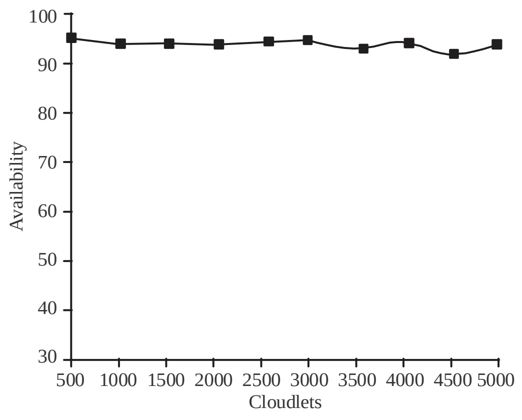


Fig. 14: Availability result related to number of Cloudlets for EMLMA

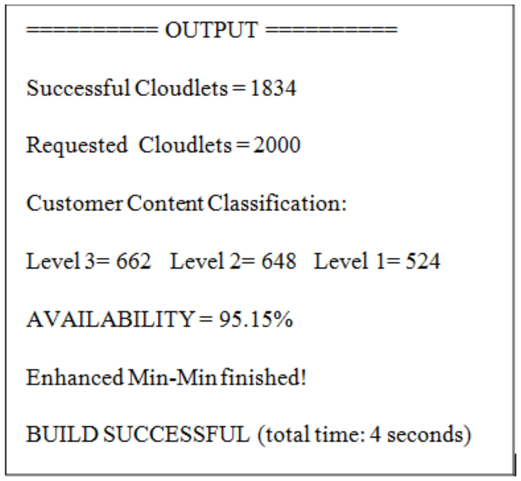
QoS requests first (Level 3), after completing that, it will process lower level (level 2) and then it will process lowest level (level 1).

The algorithm will do two parts as in Min-Min algorithm: calculation of the completion time of all tasks or jobs and then finding the with minimum completion time and assign it to the virtual machine most suitable for it. As depicted in Fig. 14, the first part has two for loops, M represents the number of jobs to be scheduled, N represents the number of virtual machines, C_{ij} represents the completion time of a job, E_{ij} represents the execution time of the job and R_j represents the ready time of a job i on virtual machine j.

To calculate the completion time of a job (C_{ij}), we add the execution time of that task (E_{ij}) to the ready time of the task (R_j), this will be repeated N times and then repeated M times.

After that, another loop will start till scheduling all tasks or jobs. All unscheduled jobs with level 3 will be checked to find the job with minimum completion time (t_p) and the suitable virtual machine for it. Then Assign the job (t_p) to the virtual machine that gives minimum completion time, and delete it from the unscheduled jobs. Then update the ready time of the machine that gives the minimum completion time. This will be repeated to schedule all jobs. After finishing all jobs with level 3 requests, another loop will do the same scenario for level 2 and finishing the all jobs. Then at last, level 1 jobs will be processed.

Simulation and comparison: We have simulated the EMLMA in Cloudsim and have found that EMLMA has availability percentage around 95% as shown in Fig. 13. EMLMA is considering the three levels of quality stated in CITC regulations for cloud computing in Saudi Arabia.


```
========== OUTPUT ==========

Successful Cloudlets = 1834

Requested  Cloudlets = 2000

Customer Content Classification:

Level 3= 662   Level 2= 648   Level 1= 524

AVAILABILITY = 95.15%

Enhanced Min-Min finished!

BUILD SUCCESSFUL (total time: 4 seconds)
```

Fig. 15: Sample result of simulation at 2000 cloudlets for EMLMA

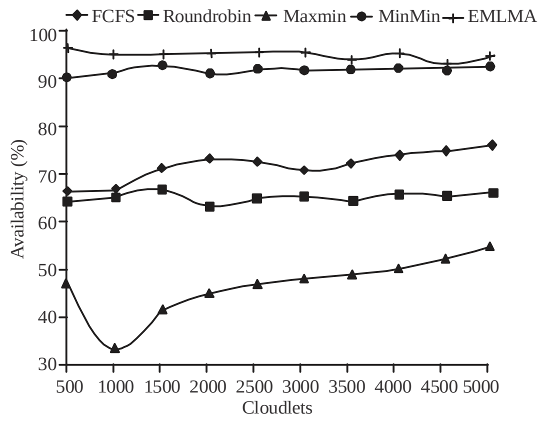


Fig. 16: Availability result for EMLMA and other algorithms

As shown in Fig. 15, a sample result of the simulation is shown when the number of requested cloudlets is 2000. EMLMA will process the short cloudlets first in level 3, and then process the short cloudlets in level 2 and then process the short cloudlets in level 1. this means the successful number of cloudlets with higher level of quality will be high.

Figure 16 is showing the availability percentage of all the four algorithms, namely FCFS, Round Robin, Max Min and Min Min in addition to our new algorithm EMLMA. It is clear that EMLMA has got the best availability compared to other algorithms including Min Min and that is because our new algorithm is considering the three levels of quality, and processing level 3 short cloudlets first, and hence completing more cloudlets successfully which have more weight.

CONCLUSION

We have studied four algorithms and explained the experimental setup for them. Then, we have simulated them, got the results and discussed results for all algorithms. After that, we have developed our new algorithm for the QoS Model. Then, we have simulated the new enhanced algorithm and compared it to other algorithms.

REFERENCES

01. Mell, P. and T. Grance, 2011. The NIST definition of cloud computing recommendations of the national institute of standards and technology. Nist Spec. Publ., 145: 1-7.
02. Pearson, S. and A. Benameur, 2010. Privacy, security and trust issues arising from cloud computing. Proceedings of the 2010 IEEE 2nd International Conference on Cloud Computing Technology and Science, November 30-December 3, 2010, IEEE, Indianapolis, Indiana, ISBN:978-1-4244-9405-7, pp: 693-702.
03. Hammadi, A.M. and O. Hussain, 2012. A framework for SLA assurance in cloud computing. Proceedings of the 2012 26th International Conference on Advanced Information Networking and Applications Workshops, March 26-29, 2012, IEEE, Fukuoka, Japan, pp: 393-398.
04. Aldakheel, E.A., 2011. A cloud computing framework for computer science education. M.Sc. Thesis, Bowling Green State University, Bowling Green, Ohio.
05. Tashkandi, A.N. and I.M. Al-Jabri, 2015. Cloud computing adoption by higher education institutions in Saudi Arabia: An exploratory study. Cluster Comput., 18: 1527-1537.
06. Alhammadi, A., C. Stanier and A. Eardley, 2015. The determinants of cloud computing adoption in Saudi Arabia. Comput. Sci. Inf. Technol. (CS&IT.), 1: 55-67.
07. Alkhater, N., G. Wills and R. Walters, 2014. Factors influencing an organisation's intention to adopt cloud computing in Saudi Arabia. Proceedings of the 6th International Conference on Cloud Computing Technology and Science (CloudCom), December 15-16, 2014, IEEE, Southampton, England, ISBN:978-1-4799-4093-6, pp: 1040-1044.
08. Saudi Ministry of Communications and Information Technology, 2015. Report: Kingdom’s enterprises adoption of cloud reduces costs, expenses by more than 50%. Saudi Ministry of Communications and Information Technology, Riyadh, Saudi Arabia.
09. Alharbi, F., A. Atkins and C. Stanier, 2015. Strategic framework for cloud computing decision-making in healthcare sector in Saudi Arabia. Proceedings of the 7th International Conference on Ehealth, Telemedicine and Social Medicine, Vol. 1, February 22-27, 2015, IARIA, Lisbon, Portugal, pp: 138-144.
10. Eman, A.T. and H. Mohammad, 2014. The impact of cloud computing on Saudi organizations: The case of a telecom company. Int. J. Comput., 3: 126-130.
11. PricewaterhouseCoopers, 2011. Cloud assurance. PricewaterhouseCoopers, London, UK.
12. Abielmona, R. and T.W. Pearce, 2000. Scheduling algorithmic research. Department of Electrical and Computer Engineering, Ottawa-Carleton Institute, Ottawa, Ontario.
13. Madni, S.H.H., M.S. Abd Latiff, M. Abdullahi, S.I.M. Abdulhamid and M.J. Usman, 2017. Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment. PloS One, Vol. 12, No. 5. 10.1371/journal.pone.0176321