Allocation of Cloudlets using Solution of Job Assignment Problem

Sonam Pathak, Manish Pandey, Kanu Geete

Abstract: Resource allocation policies play a key role in determining the performance of cloud. Service providers in cloud computing have to provide services to many users simultaneously. So the job of allocating cloudlets to appropriate virtual machines is becoming one of the challenging issues of cloud computing. Many algorithms have been proposed to allocate cloudlets to the virtual machines. Here in our paper, we have represented cloudlet allocation problem as job assignment problem and we have proposed Hungarian algorithm based solution for allocating cloudlets to virtual machines. The main objective is to minimize total execution time of cloudlets. Proposed algorithm is implemented in Cloudsim-3.03 simulator. We have done comparative analysis of the simulation results of proposed algorithm with the existing First Come First Serve (FCFS) scheduling policy and Min-Min scheduling algorithm. Proposed algorithm performs better than the above mentioned algorithms in terms of total execution time and makespan time (finishing time of last cloudlet).

***Index Terms*: CloudSim, cloudlets, Hungarian algorithm, resource allocation, virtual machines**

I. INTRODUCTION

Cloud Computing is internet based computing in which services are provided on demand as per the requirements of clients. Three main services provided by cloud computing is infrastructure as a service (IAAS), platform as a service (PAAS) and software as a service (SAAS) [1]. Cloud computing is based on pay per usage model. Customers will have to pay only for what they consume. Cloud computing provide mobility, scalability and flexibility in service provisioning. In cloud, users' requests for resources are termed as cloudlets. These cloudlets need to be handled effectively with available resources. So main concern in cloud computing is to use efficient allocation algorithms in order to allocate virtual machines (VM) to the cloudlets [2].

Cloud computing environments are used for implementing scheduling algorithms. Some basic entities of cloud computing environments are Datacenter, Host, Virtual machines (VM), Tasks/Cloudlets, Datacenter Broker and Cloud Information Service (CIS) [3]. Fig 1 shows the pictorial representation for cloudlet allocation in cloud. Datacenter is group of different host. Each host may consist of different virtual machines of different specifications. CIS stores metadata of different cloud entities. CIS is kind of repository in cloud computing environment. Datacenter broker works between clients and. cloud service supplier. It gathers all the required information related to availability of resources, usage of resources and cost of communication. Then according to the used scheduling algorithm datacenter broker allocates cloudlets to virtual machines.

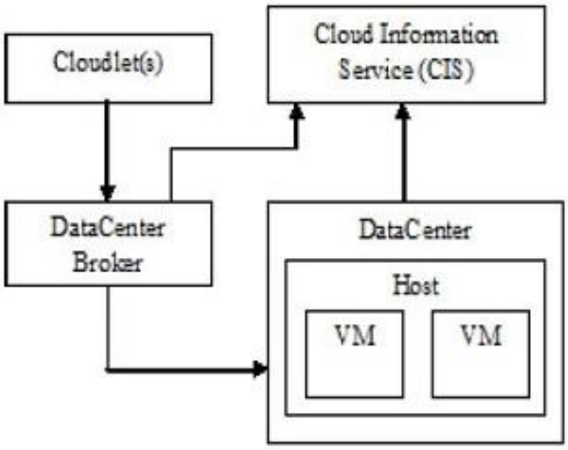

Fig.1. Cloudlet Allocation in Cloud Computing

In our paper, we have used one of the solutions of job assignment problem i.e. Hungarian algorithm for allocation of cloudlets. Our proposed algorithm minimizes total execution time of all cloudlets.

II. RELATED WORK

Cloudlet allocation is one of the most trending research topic in cloud computing. Several algorithms for cloudlet allocation have been proposed. The default policy used by Cloudsim toolkit is first come first serve policy (FCFS) [4]. FCFS simply allocates first cloudlet to first virtual machine, second cloudlet to second virtual machine and so on. Disadvantage of FCFS is that it does not consider which virtual machine will take less time for execution of particular cloudlet. Min-Min scheduling algorithm firstly computes minimum time required for execution of each cloudlet. Among all the cloudlets it selects the cloudlet with least execution time. Then it assigns this cloudlet to the virtual machine that computes minimum time for that cloudlet. This process continues for all cloudlet. [5] [6].

In Max-Min scheduling algorithm first step is same as Min-Min i.e. we calculate

Revised Manuscript Received on July 10, 2019.

Sonam Pathak, Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, India.
Manish Pandey, Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, India.
Kanu Geete, Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, India.

minimum execution time of each cloudlet. Then instead of selecting the cloudlet with least execution time we select the cloudlet with maximum execution time and repeat the process until last cloudlet is allocated [7]. A reliability based resource allocation has also been proposed by A B M Bodrul Alam and Mohammad Zulkernine [8]. The main goal of this allocation method is to

increase the reliability of cloud by increasing the reliability of its resources. These days, a lot of other domains including machine learning and big data utilize the capabilities of each other to give large efficient enterprise solutions in cloud. An extension of the shortest job first algorithm has been proposed by Mokhtar A. Alworafi [9]. This modified shortest job first algorithm performs better than the shortest job first algorithm by considering mean of all the cloudlets.

III. Proposed Method

We have related "Cloudlet Allocation Problem Model" with "Job Assignment Problem Model" [9]. We have proposed a linear programming model for mapping cloudlets to the virtual machines. Let us consider a set of virtual machines with n virtual machines and a set of cloudlets with m cloudlets. Assume that allocation is such that per virtual machine one cloudlet is allocated.

Let us take n=m .We will calculate the cost matrix C = [c_{ij}] by using formula which is mentioned in (1) . Here C is an n×n cost matrix and c_{ij} denotes i, j entry of matrix.

$$c_{ij} = l_j / vm_i \qquad (1)$$

c_{ij} = Execution time of jth cloudlet on ith virtual machine

l_j = length of jth cloudlet in million instructions

vm_i = capacity of ith virtual machine in million instructions per second

Let Y= [y_{ij}] is another n×n matrix where y_{ij} is

$$= \begin{cases} 1 \quad , if\ cloudlet\ j\ is\ allocated\ to\ virtual\ machine\ i \\ 0, if\ cloudlet\ j\ is\ not\ allocated\ to\ virtual\ machine\ i \end{cases}$$

Our goal is to optimize the cost q(Y). Here cost is the total time required for execution of all cloudlets. Linear model of our cloudlet allocation problem is as follows:

$$Minimize\ q(Y) = \sum_{j=1}^{m} \sum_{i=1}^{n} c_{ij}\, y_{ij} \qquad (2)$$

Subjects to

$$\sum_{i=1}^{n} y_{ij} = 1 \quad \text{for } j = 1,2,3....m \qquad (3)$$

$$\sum_{j=1}^{m} y_{ij} = 1 \quad \text{for } i = 1,2,3.....n \qquad (4)$$

$$y_{ij} = 0 \text{ or } 1 \qquad (5)$$

This linear programming model of cloudlet allocation is same as job assignment model. So we can use the solution of assignment problem to solve our cloudlet allocation problem. We have used Hungarian Method based solution for our problem. Hungarian Method is one of the standard solutions of job assignment problem which we are using for solving our cloudlet allocation problem. The objective of proposed method is to lessen the overall execution time of all cloudlets.

In the proposed method, we have considered equal number of cloudlets and equal number of virtual machines. If available virtual machines and cloudlets are not equal in number then we add dummy rows. We assume that each cloudlet needs to be allocated to just single VM and each VM will execute just one cloudlet.

A. Method Description

Let us take an example with 3 virtual machines and 3 cloudlets.

Table1. MIPS Value for Virtual Machines

Parameter	VM1	VM2	VM3
Capacity	200	500	100

Table2. Length of Cloudlets

Parameter	CL1	CL2	CL3
Length	20000	60000	40000

Step1: Compute the initial cost matrix by the mentioned formula.

Table3. Cost Matrix

	CL1	CL2	CL3
VM1	100	300	200
VM2	40	120	80
VM3	200	400	600

Step2: Subtracting minimum element of each row from all the values of its row.

Table4. Cost Matrix after performing step2

	CL1	CL2	CL3
VM1	0	200	100
VM2	0	80	40
VM3	0	200	400

Step3: Subtracting minimum element of each column from all the elements of its column.

Table5. Reduced Cost Matrix

	CL1	CL2	CL3
VM1	0	120	60
VM2	0	0	0
VM3	0	120	360

Step4: Now we find out minimum number of lines required to cover all zeroes in Reduced Cost Matrix. As we can see two lines will be needed one at column 1 and one at row 2.

If number of lines is same as number of available virtual machines then we do mapping of cloudlet and virtual machine. If number of lines is less then number of virtual machine then we find out minimum value element among all uncovered elements. We subtract this least value element from all the uncovered elements and add this element to the elements where lines intersect with each other.

Table6. Final Cost Matrix

	CL1	CL2	CL3
VM1	0	60	0
VM2	60	0	0
VM3	0	60	300

Step5: Now we require three lines to cover all zero, one line at column 1, one line at row 2 and the last one at column 3. We do mapping of cloudlet to virtual machine based on zeros.

VM1-> CL3
VM2-> CL2
VM3-> CL1

Now we get total execution time (Cost) by adding the value of corresponding mapping in initial cost matrix.
Cost = 200+120+200= 520 seconds

B. Algorithm

Input:
n: available virtual machines
m: available cloudlets
Output:
Cost: Total execution time of all cloudlets

```
/* Initialize value of cost matrix*/
for i=1 to n
  for j=1 to m
     c_ij = l_j / vm_i
  end for
end for
if n is not equal to m
 put dummy value in matrix to convert matrix in
   Square matrix
end if
minr_i = smallest element in row i
minc_i = smallest element in column i
/* calculate reduced cost matrix */
for i=1 to n
  for j=1 to m
     c_ij = c_ij - minr_i
  end for
end for
for i=1 to n
  for j=1 to m
     c_ji = c_ji - minc_i
end for
end for
/* calculate final cost matrix */
k = min_no_line()
if k<n
 for i=1 to n
  for j=1 to m
   if c_ij is not covered by any of line
      c_ij = c_ij – minimum of all uncovered elements
   else if c_ij is covered by two line
      c_ij = c_ij + minimum of all uncovered elements
   end if
  end for
 end for
end if
/* Finding mapping of cloudlets to virtual machines */
Find_mapping() for all cloudlets
adding their execution time will give total execution time
 i.e. Cost
return Cost
```

IV. Experimental Results and Comparative Analysis

We have used open source simulator named CloudSim-3.0.3 and Eclipse Java IDE for our experimental purpose. Cloudsim toolkit gives the user the flexibility to implement their own algorithms for provisioning of resources [11][12].

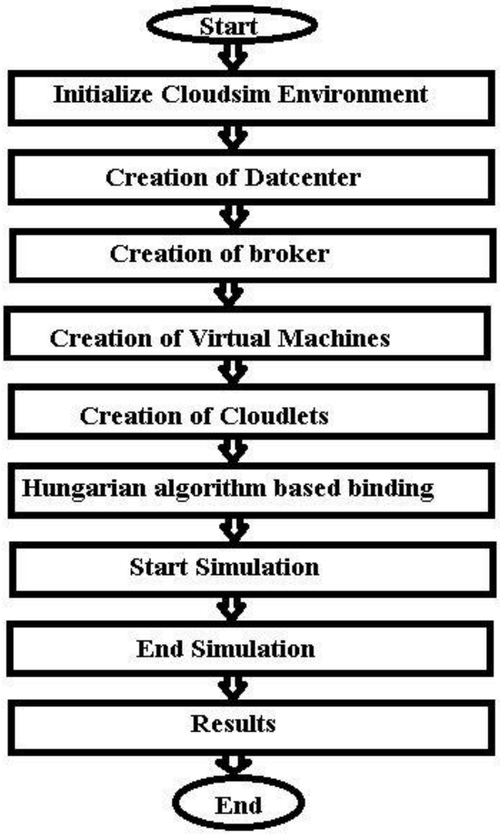

Fig.2. Cloudsim Lifecycle

Cloudsim gives classes for virtual machines, data centers, applications, clients and scheduling methods. Different stages of Cloudsim lifecycle is shown in Fig.2. Cloudsim life cycle starts with initialization of Cloudsim environment and ends with simulation results [13][14][15]. Scheduling algorithm is applied after creation of cloudlets. We have executed our proposed algorithm in Cloudsim and compared its result with conventional FCFS policy and Min-Min scheduling algorithm. Simulation results are given in tables 9, 10 and 11.

Table7. Length of Cloudlets for 3 different jobs

Cloudlet Id	Job1	Job2	Job3
	(Length)	(Length)	(length)
0	20000	200000	40000
1	60000	20000	120000
2	90000	80000	60000
3	40000	90000	20000
4	120000	10000	70000

Table8. MIPS of Virtual Machines

VM ID	0	1	2	3	4
MIPS	1000	500	200	2000	250

Table9. Result of Conventional FCFS Scheduling in Cloudsim

Cloudlet id	Job1		Job2		Job3	
	VM id	Exec. Time	VM id	Exec. Time	VM ID	Exec. Time
0	0	20	0	200	0	40
1	1	120	1	40	1	240
2	2	450	2	400	2	300
3	3	20	3	45	3	10
4	4	480	4	40	4	280
Total Exec. Time	1090 sec		725 sec		870 sec	

Table10. Result of Min-Min scheduling in Cloudsim

Cloudlet id	Job1		Job2		Job3	
	VM id	Exec. Time	VM id	Exec. Time	VM id	Exec. Time
0	0	20	1	20	0	20
1	1	120	2	160	2	180
2	4	600	0	1000	1	300
3	3	20	4	5	3	20
4	2	360	3	360	4	480
Total Exec. Time	1120 sec		1545 sec		1000 sec	

Table11. Result of our proposed Hungarian Scheduling in Cloudsim

Cloudlet id	Job1		Job2		Job3	
	VM id	Exec. Time	VM id	Exec. Time	VM id	Exec. Time
0	2	90	3	90	4	70
1	1	120	2	160	2	120
2	0	100	4	50	3	100
3	4	60	0	100	1	60
4	3	160	1	80	0	160
Total Exec. Time	530 sec		480 sec		510 sec	

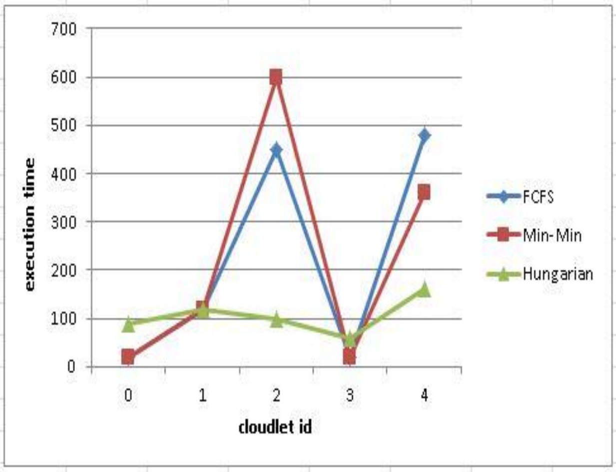

Fig.3. Execution Time of Cloudlets for job1

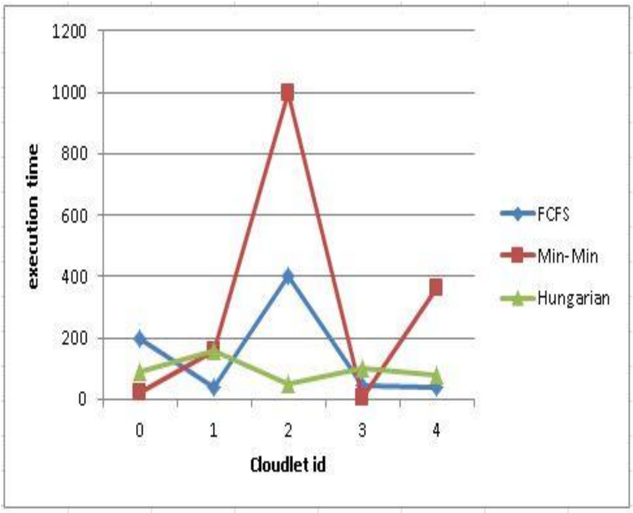

Fig.4. Execution Time of Cloudlets for job2

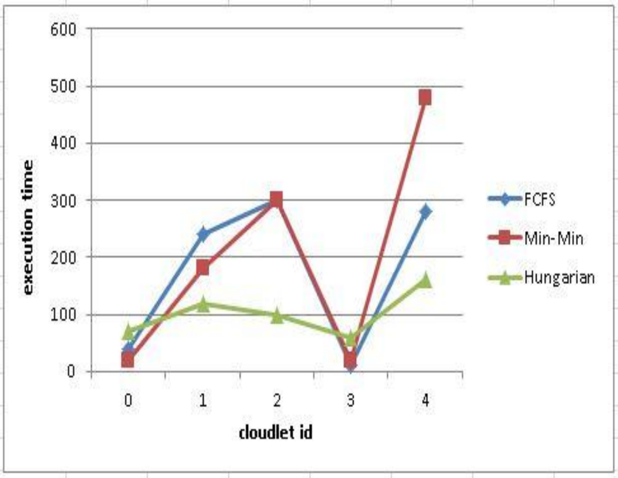

Fig.5. Execution Time of Cloudlets for job3

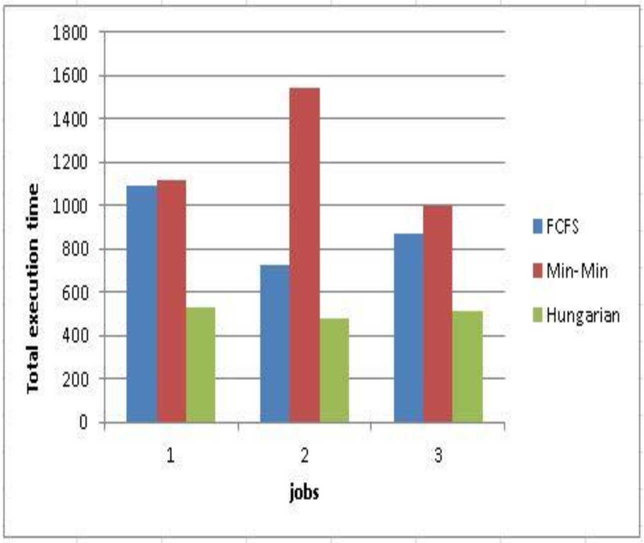

Fig.6. Comparing total execution time of proposed algorithm with FCFS and Min-Min scheduling algorithm for 3 different jobs

Fig. 3, 4 and 5 shows graphs of execution time for job1, job2 and job3 respectively. We have shown comparative analysis of total execution time of each job in Fig.6. Table 12 shows makespan time for 3 jobs using proposed method, conventional policy and min-min scheduling. Fig.7 gives graphical representation for comparing makespan time. We can clearly see from the tables and graphs that total execution time and makespan time is improved using our proposed method. Rate of improvement in makespan time for job1 using our proposed method is 66.67% over FCFS and 73.33% over Min-Min, for job2 improvement is 60% over FCFS and 84% over Min-Min and for job3 improvement is 37.5% over FCFS and 66.67% over Min-Min algorithm.

Table12. Comparing Makespan time for job1, job2 and job3 using FCFS, Min-Min and Proposed approach

Job	FCFS	Min-Min	Proposed Method
1	480	600	160
2	400	1000	160
3	300	480	160

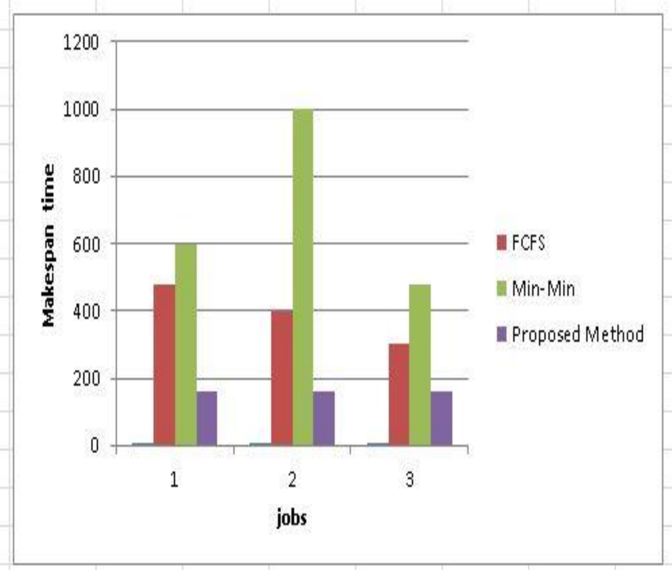

Fig.7. Comparing Makespan time of proposed method with existing ones

V. Conclusion and Future Work

In this paper, we have proposed Hungarian Algorithm based solution for cloudlet allocation problem by keeping in mind the issues of resource allocation problem. Our proposed algorithm allocates each cloudlet to appropriate virtual machine in such a way that overall execution time is minimized. Consequently makespan time is also minimized. We have implemented our proposed algorithm, first come first serve scheduling (FCFS) and Min-Min scheduling algorithm in Cloudsim tool and we have compared the simulation results of these three algorithms. Our proposed algorithm is more efficient than the existing conventional FCFS binding policy and Min-Min scheduling algorithm in terms of total execution time and makespan time.

In future, our proposed method may be used for allocation of cloudlets in real time. By keeping in mind the efficiency of the proposed approach, same method can be used for fog computing and mobile computing applications.

.

REFERENCES

1. Saurabh Kumar Garg, Steeve Versteeg and Rajkumar Buyya, "A Framework for ranking of cloud computing services", S.K. Garg et al. / Future Generation Computer Systems 29 (2013) 1012-1023.
2. MR.Manan D. Shah, MR.Amit A. Kariyani and MR.Dipak L. Agrawal, "Allocation of Virtual Machines In Cloud ComputingUsing Load Balancing Algorithm" *IRACST - International* Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol. 3, No.1, February *2013*.
3. Tarun Goyal, Ajit Singh and Akansha Goyal, "Cloudsim:Simulator for cloud computing infrastructure and modeling" International Conference on Modeling, optimization and computing (ICMOC-2012).
4. Dr. Amit Agarwal and Saloni Jain, "Efficient Optimal Algorithm of Task Scheduling in Cloud Computing Environment" International Journal of Computer Trends and Technology (IJCTT) – volume 9.
5. Gaurang Patel, Rutvik Mehta and 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).
6. Neha Sharma and 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.
7. Upendra Bhoi1 and Purvi N. Ramanuj2, "Enhanced Max-min Task Scheduling Algorithm in Cloud Computing", Web Site: www.ijaiem.org Email: editor@ijaiem.org,editorijaiem@gmail.com Volume 2, Issue 4, April 2013.
8. A B M Bodrul Alam, Mohammad Zulkernine and Anwar Haque, "A Reliability-Based Resource Allocation Approach for Cloud Computing", 2017 IEEE 7th International Symposium on Cloud and Service Computing
9. Mokhtar A. Alworafi, Atyaf Dhari and Asma A. Al-Hashmi, "An Improved SJF Scheduling Algorithm in Cloud Computing Environment" 2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT).
10. "Linear Programming Applications-nptel" https://nptel.ac.in/courses/110106059/Module%201/Lecture%202.pdf
11. Gibet Tani Hicham and El Amrani Chaker, "Cloud Computing CPU Allocation and Scheduling Algorithms Using CloudSim Simulator" International Journal of Electrical and Computer Engineering IJECE)Vol. 6, No. 4, August 2016, pp. 1866~1879 ISSN: 2088.
12. Pradeep Singh Rawat, G. P. Saroha and Varun Barthwal"Quality of Service Evaluation of SaaS Modeler (Cloudlet) Running on Virtual Cloud Computing Environment using CloudSim"*International Journal of Computer Applications (0975– 8887) Volume 53– No.13, September 2012"*.
13. Suchintan Mishra and Manmath Narayan Sahoo, "On using CloudSim as a Cloud Simulator: The Un-official Manual", Research · December 2017 DOI: 10.13140/RG.2.2.30215.91041
14. Arezoo Khatibi and Omid Khatibi, "Criteria for the CloudSim Environment", arXiv:1807.03103v1 [cs.DC] 14 Jun 2018
15. Ranjan Kumar and G. Sahoo, "Cloud Computing Simulation Using CloudSim", International Journal of Engineering Trends and Technology (IJETT) – Volume8 Number 2-Feb 2014

AUTHORS PROFILE

First Author: Sonam Pathak, Research Scholar in Computer Science and engineering at Maulana Azad National Institute of Technology, Bhopal Email- 22pathaksona@gmail.com

Second Author : Dr Manish Pandey, Assistant Professor at Maulana Azad National Institute of Technology, Bhopal, India,.
Email- contactmanishpandey@yahoo.co.in

Third Author: Kanu Geete, Phd scholar at Maulana Azad National Institute of Technology, Bhopal, India.
Email- kanu.geete29@gmail.com