Multi-objective Prediction based Task Scheduling Method in Cloud Computing

Swapnil M Parikh, Saurabh A Shah, Narendra M Patel

Abstract*: Cloud Computing is Internet based computing where one can store and access their personal resources from any computer through Internet. Cloud Computing is a simple pay-per-utilize consumer-provider service model. Cloud is nothing but large pool of easily accessible and usable virtual resources. Task (Job) scheduling is always a noteworthy issue in any computing paradigm. Due to the availability of finite resources and time variant nature of incoming tasks it is very challenging to schedule a new task accurately and assign requested resources to cloud user. Traditional task scheduling techniques are improper for cloud computing as cloud computing is based on virtualization technology with disseminated nature. Cloud computing brings in new challenges for task scheduling due to heterogeneity in hardware capabilities, on-demand service model, pay-per-utilize model and guarantee to meet Quality of Service (QoS). This has motivated us to generate multi-objective methods for task scheduling. In this research paper we have presented multi-objective prediction based task scheduling method in cloud computing to improve load balancing in order to satisfy cloud consumers dynamically changing needs and also to benefit cloud providers for effective resource management. Basically our method gives low probability value for not capable and overloaded nodes. To achieve the same we have used sigmoid function and Euclidean distance. Our major goal is to predict optimal node for task scheduling which satisfies objectives like resource utilization and load balancing with accuracy.

***Keywords*: Cloud Computing, Task Scheduling, Multi-objective, Resource Management.**

I. INTRODUCTION

Because of the advancement in Information and Communication Technology (ICT) over past few years, Computing has been considered as a utility like water, electricity, gas and telephony. These utilities are available to the consumers based on their requirement at any time. Consumers pay for these services to the service providers based on their usage [1], [2], [3].

Like all the other existing utilities, Computing utility is the basic computing service that meets the day to day needs of the general community. To deliver this vision, a number of computing paradigms have been proposed, of which the latest one is known as Cloud Computing.

Revised Manuscript Received on November 19, 2019.

***** Correspondence Author

Swapnil M Parikh*, Department of Computer Science and Engineering, Babaria Institute of Technology, Vadodara, Gujarat, INDIA. Email: swapnil.parikh@gmail.com

Saurabh A Shah, Department of Computer Engineering, C U Shah University, Wadhwan, Surendranagar, Gujarat, INDIA. Email: saurabh_er@rediffmail.com

Narendra M Patel, Department of Computer Engineering, Birla Vishwakarma Mahavidydyalaya, V V Nagar, Gujarat, INDIA. Email: nmpatel@bvmengineering.ac.in

Task Scheduling is always a major issue in any computing paradigm. Any task scheduling method should fulfill objectives like load balancing and efficient resource utilization as we have availability of finite resources. Traditional task scheduling techniques are not adequate for cloud computing as cloud computing is based on virtualization technology with distributed nature. Cloud computing introduces new challenges for task scheduling due to heterogeneity in hardware capabilities, on-demand service model, pay per use model and guarantee to meet QoS [4], [5], [6], [7].

Many Authors have proposed various methods for task scheduling as a part of resource management. These methods were based on single objective or multi-objective. But there were no major multi-objective method for task scheduling in resource management for cloud computing. Many authors have proposed hybrid algorithms for task scheduling and have compared their algorithm with various other algorithms like particle swarm optimization, cuckoo etc. There were variations in parameters and these parameters were also not mathematically described for measuring accuracy [8], [9], [10].

The rest of paper is organized as follows: Section II discusses overview of cloud computing. Section III introduces various terminologies / vectors used in our proposed method. Section IV gives our proposed multi-objective prediction based method for task scheduling. Section V gives mathematical formulation and experiment results of our method. Section VI gives conclusion and future scope.

II. OVERVIEW OF CLOUD COMPUTING

Cloud computing is one of the innovative computing, which deals with storing and accessing data and programs over the Internet. Cloud delivers computing resources and services, such as storing of data on servers and databases. It also provides networking facilities and software development platforms over the Internet. Cloud computing makes access to these resources to everyone on a global scale at a very minimal cost and significantly higher speed. Cloud computing is a service provision model which provides various kinds of agile and effective services to the consumers where everything is considered as a service. Cloud is nothing but large pool of easily accessible and usable virtual resources. Cloud delivers computing as a utility as it is available to the cloud consumers on demand [1], [2], [3], [11].

A. Cloud Service Models

Cloud Software as a Service (SaaS): In this service model, instead of using locally run applications the cloud consumer uses the cloud providers' software services running on a cloud infrastructure. It is the job of cloud provider to maintain and manage the software services that are used by the cloud consumer. The cloud provider may charge according to software quantity and time usage. Salesforce.com and Customer Relationship Management (CRM) are the examples of such service model [12], [16], [17], [18], [19], [20].

Cloud Platform as a Service (PaaS): In this service model, the cloud platform offers an environment on which developers create and deploy applications. It provides platform where applications and services can run. The consumers do not need to take care of underlying cloud infrastructure including network, servers, operating system or storage but has a control over deployed application. Google Application Engine and RightScale are the example of such model [12], [16], [17], [18], [19], [20].

Cloud Infrastructure as a Service (IaaS): In this service model, cloud providers manage large set of computing resources such as storing and processing capability. Cloud consumer can control operating system; storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls). Sometimes it is also called as a Hardware as a Service (HaaS). The cost of the Hardware can be greatly reduced here. Amazon Web Services, Microsoft Azure, Open Stack, Eucalyptus, GoGrid and Flexiscale offers IaaS [12], [16], [17], [18], [20].

B. Cloud Deployment Models

In cloud computing various deployment models have been adopted based on their variation in physical location and distribution. Regardless of the services, clouds can be classified among four models as mentioned below:

Private Cloud: It is private to the organization. It supports single tenant architecture. All the cloud services are managed by the organization people themselves or any third party vendors as well as services are not provided to the general public. Private cloud may exist on premise or off premise. Usually private cloud is used by government organizations or few large scale industries [12], [16], [17], [21].

Public or Hosted Cloud: This is most commonly used Cloud. All the cloud services managed by the cloud provider are made available as in pay as you go manner to the general public. The web browser is the way to access accounts. It supports multi-tenant architecture. The business people can adopt such cloud to save their hardware and/or software cost. Public cloud may raise number of issues like data security, data management, performance, level of control etc. Microsoft Azure is the best example of public cloud [12], [16], [17], [21].

Community Cloud: Here cloud is available to specific group of people or community. All the cloud services are shared by all these community people. Community cloud may exist on premise or off premise [12], [16], [17], [21].

Hybrid Cloud: It is a combination of two or more cloud models mentioned above so it is more beneficiary. This is most flexible cloud deployment model [12], [16], [17], [21].

C. Key features and/or technologies of Cloud Computing

- Virtualization: It hides the essential physical characteristics of the computing environment from the cloud consumers. By virtualization of the machine, one can be able to run several operating systems (and all of their applications) at the same time. This imitated computing environment for all practical purposes behaves like an independent system. But unlike a physical system, can be configured on demand, and maintained and reproduced effortlessly [13]. [22].
- Multi-tenancy: Multi-tenancy is a new unique feature in cloud computing through which cloud providers can utilize systems' resources in a better way. System resources can be processor, memory or storage. Here multiple customers are provided with single instance of software application. We call each customer as a tenant [16].
- Distributed Storage: The data storage system of cloud computing is distributed which ensures high economy and reliability of stored data. Reliability can be achieved through redundancy. Google File System (GFS) and Hadoop Distributed File System (HDFS) are the examples of Distributed Data Storage System of Cloud Computing [16].
- Data Management: From Data Management point of view cloud computing provides full availability of data. Due to distributed storage system, data management is very important in cloud computing. Cloud computing needs to analyze and process these distributed data in an efficient manner. BigTable of Google and HBase developed by Hadoop team are the data management technology in cloud computing [16].
- Service Level Agreements (SLAs): SLAs are required in cloud computing for controlling the use of computing resources. They relate cloud consumer and provider both. Service Level Agreement is nothing but a service contract between the cloud consumer and cloud provider. It formally defines the level of service. Some of the SLA parameters can be processor speed, storage and memory capabilities, availability etc [16].
- Parallel Programming Model: Cloud Computing adopts almost all the features of Parallel Computing. Job Scheduling and Parallel Execution must be transparent to the cloud consumers so that cloud computing services can be utilized efficiently. MapReduce is a parallel computing model developed by Google and has been adopted by cloud computing [16].

D. Research issues / challenges in Cloud Computing Environment

In Cloud Computing Environment, following research issues / challenges are of current interest in top conferences / journals [16], [18], [23]:

- Efficient resource management
- Information security and data integrity

- Maintaining users' privacy and trust requirements
- Storage architectures and implementations
- Interoperability
- Service Level Agreement and QoS Management
- Various models of pricing for cloud services.

Efficient resource assignment can be considered in two parts:

1. Initial Resource Assignment
2. Periodic Resource Optimization

Our major focus was on Initial Resource Assignment where we explored various traditional methods. But these methods were not appropriate for cloud as cloud has diversified challenges. So here we propose our method.

III. VARIOUS TERMINOLOGIES / VECTOTS USED IN PROPOSED METHOD

- Need Vector (N): This vector will be input to the method. It shows resource need/requirement of various tasks.
- Scheduling Vector (S): This vector shows the no of tasks currently running on a particular node. It is used during middle level computations which are so called as TieBreaker for Overloading.
- Current Availability Vector (C): This vector shows the current available resources at any particular node.

IV. PROPOSED MULTI-OBJECTIVE PREDICTION BASED METHOD FOR TASK SCHEDULING

Fig. 1 gives the algorithmic steps for Proposed Multi-objective Prediction based Method for Task Scheduling in Cloud Computing Environment. Fig. 2 represents block diagram for the proposed method.
In this proposed method, below mentioned activation functions are used:

- Sigmoid Function: A sigmoid function is a mathematical function having a characteristic "S"-shaped curve or sigmoid curve. Often, sigmoid function refers to the special case of the logistic function as shown in below Fig. \ref{sig} and defined by the formula:

$$\text{Sig} = \frac{1}{1+e^{-x}} \qquad (1)$$

The sigmoid function should only be used when positive number output is expected because the sigmoid function will only produce positive output. It will move negative numbers into the positive range.
The main reason why we use sigmoid function is because it exists between 0 and 1. Therefore, it is especially used for models where we have to predict the probability as an output. Since probability of anything exists only between the range of 0 and 1, sigmoid is the better choice.

- Euclidean Distance: In mathematics, the Euclidean distance or Euclidean metric is the "ordinary" straight-line distance between two points in Euclidean space. The Euclidean distance between points p and q is the length of the line segment connecting them.

$$\text{ED (p, q)} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \qquad (2)$$

In the first middle phase of the proposed method, sigmoid function is used to identify incapable nodes for task scheduling. Second middle phase removes overloaded nodes based on scheduling vector values for each node. Output phase uses Euclidean distance based on current availability vector and need vector.

Input: Need Vector (N), Scheduling Vector (S), Current Availability Vector (C)

Output: Predicted Node Index for Task Scheduling

Steps:

1. Input Phase: Need Vector (N), Scheduling Vector (S), Current Availability Vector is fed into the input phase.
2. 1st Middle Phase: Here we ensure that whether node is capable of handling the task. To do so, sigmoid function is used as shown below:

$$Sig = \frac{1}{1+e^{N-C}}$$

where N is Need Vector and C is Current Availability Vector.

3. 2nd Middle Phase (TieBreaker for Overloading):

Find MIN(S) → X

If S_i == x

Then return (1*input i)

Else return (0*input i)

where i denotes i^{th} node.

4. Output Phase:

For each node i,

Find ED (C,N) → d_i

Then i^{th} input * d_i

where ED stands for Euclidean Distance, C is Current Availability Vector and N is Need Vector.

5. Scheduling can be done as:

MAX (d_i) with MIN (i).

Fig. 1. Proposed Method

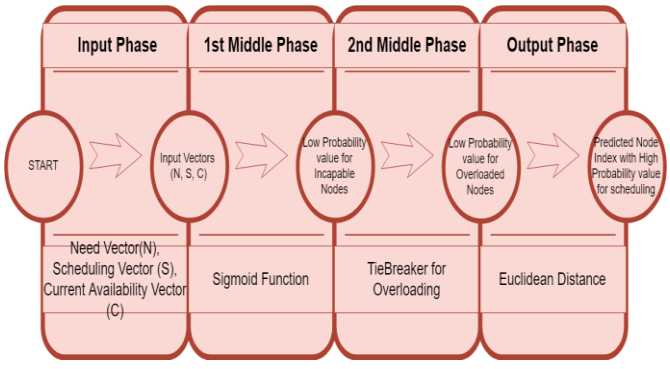

Fig. 2.Block Diagram of Proposed Method

V. EXPERIMENTAL RESULTS

A. Mathematical Formulation of Parameters

Let $T = (t_1, t_2, t_3, ..., t_n)$ be the set of tasks. Let $P = (p_1, p_2, p_3, ..., p_m)$ be the set of parameters. Let $N = (N_1, N_2, N_3, ..., N_k)$ be the set of nodes. Let S be the scheduler which works on P parameters with N no of nodes. Experimental results are based upon below mentioned parameters:

- Horizontally Scalable (P1): By increasing no of tasks, if any method works perfectly then we can say that the algorithm is horizontally scalable method.
  Let N_l be the local optimum node predicted by the scheduler S on which $t_i \in T$ will be scheduled. The scheduler S will be called Horizontally Scalable if
  $$\lim_{n\to\infty} T$$
- Vertically Scalable (P2): By increasing no of nodes, if any method works perfectly then we can say that the algorithm is vertically scalable method.
  Let N_l be the local optimum node predicted by the scheduler S on which $t_i \in T$ will be scheduled. The scheduler S will be called Vertically Scalable if
  $$\lim_{k\to\infty} N$$
  then $N_l \rightarrow C(Ng_1, Ng_2, ..., Ng_i)$ where all Ng_i are having same global optimum configuration and C is the selection function.
- Z Scalability (P3): By increasing no of decision vectors / parameters, if any method works perfectly then we can say that the algorithm is Z Scalable method.
  Let N_l be the local optimum node predicted by the scheduler S on which $t_i \in T$ will be scheduled. The scheduler S will be called Z Scalable if
  $$\lim_{m\to\infty} P$$
  then $N_l \rightarrow C(Ng_1, Ng_2, ..., Ng_i)$ where all Ng_i are having same global optimum configuration and C is the selection function.
- Biased towards Sparse Vectors (P4): If any method predicts scheduling node which is dependent of size of various vectors then it will be called as biased towards Sparse Vectors.
- Unbiased Scheduling (P5): If prediction of scheduling node is not depending up on no of decision vector then the method supports unbiased scheduling.
- Accuracy (η): An method is said to be an accurate method if it satisfied below mentioned criteria:
  - Horizontally Scalable
  - Vertically Scalable
  - Z Scalable
  - Not Biased towards Sparse Vectors
  - Unbiased Scheduling

B. Experiments on various parameters

Several experiments have been performed by taking various values in proposed multi-objective prediction based method for task scheduling. Experiments are evaluated multiple times and results are shown in various Tables from 1 to 8.

Table 1 represents need vector values of various tasks. First four columns represent various decision vectors or parameters (GPU (MB), CPU Core, RAM (MB), HDD (MB)) as a part of need vector and last column represents predicted node index (PRNI). As we can see from the table that if we increase no of tasks we get the same predicted node index, so we can say that algorithm is Horizontally Scalable. Up on changing the values of Scheduling Vector, Table 2 shows same result of accuracy.

Table 3 represents need vector values of various tasks. First five columns represent various decision vectors or parameters (GPU (MB), CPU Core, RAM (MB), HDD (MB), G-RAM (MB)) as a part of need vector and last column represents predicted node index. As we can see from the table that if we increase no of decision vectors we get the same predicted node index, so we can say that algorithm is Z Scalable. Up on changing the values of Scheduling Vector, Table 4 shows same result of accuracy.

Table 5 represents need vector values of various tasks. First four columns represent various decision vectors or parameters (GPU (MB), CPU Core, RAM (MB), HDD (MB)) as a part of need vector and last column represents predicted node index. As we can see from the table that if we increase no of nodes we get the same predicted node index, so we can say that algorithm is Vertically Scalable with Horizontally Scalable. Up on changing the values of Scheduling Vector, Table 6 shows same result of accuracy.

Table 7 represents need vector values of various tasks. First five columns represent various decision vectors or parameters (GPU (MB), CPU Core, RAM (MB), HDD (MB), G-RAM (MB)) as a part of need vector and last column represents predicted node index. As we can see from the table that if we increase no of nodes we get the same predicted node index, so we can say that algorithm is Vertically Scalable with Z Scalable. Up on changing the values of Scheduling Vector, Table 8 shows same result of accuracy.

Table 9 shows Predicted Node Index values for different No of Node values like 50, 100, 200, 500, 1000, 2000, 5000 and 10000.

The results show that the predicted node index value is not dependent on size of vectors. By increasing size of the vector it gives the same predicted value as shown in Table 9. Also by increasing the no of decision vectors or parameters the predicted node index value remains unchanged as shown in Table 9 so it is not dependent on no of decision vectors. So proposed method is not biased towards sparse vectors and it also supports unbiased scheduling.

Table 1: Horizontally Scalable

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	PRNI
60	2	250	64	4
32	1	300	34	4
16	2	200	56	4
48	2	250	64	4
60	1	300	64	4
28	3	150	30	4
20	2	200	56	4
16	2	100	56	4
48	1	150	48	4
32	1	300	34	4

Table 2: Horizontally Scalable (Different Scheduling Vector Values)

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	PRNI
60	2	250	64	6
32	1	300	34	6
16	2	200	56	6
48	2	250	64	6
60	1	300	64	6
28	3	150	30	6
20	2	200	56	6
16	2	100	56	6
48	1	150	48	6
32	1	300	34	6

Table 3: Z Scalable (Increase No of Decision Vectors)

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	G-RAM (MB)	PRNI
60	2	250	64	12	4
32	1	300	34	10	4
16	2	200	56	8	4
48	2	250	64	16	4
60	1	300	64	14	4
28	3	150	30	12	4
20	2	200	56	10	4
16	2	100	56	8	4
48	1	150	48	16	4
32	1	300	34	14	4

Table 4: Z Scalable (Increase No of Decision Vectors with Different Scheduling Vector Values)

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	G-RAM (MB)	PRNI
60	2	250	64	12	6
32	1	300	34	10	6
16	2	200	56	8	6
48	2	250	64	16	6
60	1	300	64	14	6
28	3	150	30	12	6
20	2	200	56	10	6
16	2	100	56	8	6
48	1	150	48	16	6
32	1	300	34	14	6

Table 5: Vertically Scalable (Increase No of Nodes) with Horizontally Scalable

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	PRNI
60	2	250	64	1
32	1	300	34	1
16	2	200	56	1
48	2	250	64	1
60	1	300	64	1
28	3	150	30	1
20	2	200	56	1
16	2	100	56	1
48	1	150	48	1
32	1	300	34	1

Table 6: Vertically Scalable (Increase No of Nodes) with Horizontally Scalable (Different Scheduling Vector Values)

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	PRNI
60	2	250	64	8
32	1	300	34	8
16	2	200	56	8
48	2	250	64	8
60	1	300	64	8
28	3	150	30	8
20	2	200	56	8
16	2	100	56	8
48	1	150	48	8
32	1	300	34	8

Table 7: Vertically Scalable (Increase No of Nodes) with Z Scalable (Increase No of Decision Vectors)

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	G-RAM (MB)	PRNI
60	2	250	64	12	1
32	1	300	34	10	1
16	2	200	56	8	1
48	2	250	64	16	1
60	1	300	64	14	1
28	3	150	30	12	1
20	2	200	56	10	1
16	2	100	56	8	1
48	1	150	48	16	1
32	1	300	34	14	1

Table 8: Vertically Scalable (Increase No of Nodes) with Z Scalable (Increase No of Decision Vectors) with Different Scheduling Vectors Values

GPU (MB)	CPU Core	RAM (MB)	HDD (MB)	G-RAM (MB)	PRNI
60	2	250	64	12	8
32	1	300	34	10	8
16	2	200	56	8	8
48	2	250	64	16	8
60	1	300	64	14	8
28	3	150	30	12	8
20	2	200	56	10	8
16	2	100	56	8	8
48	1	150	48	16	8
32	1	300	34	14	8

Table 9: Predicted Node Index value for different No of Nodes

No of Nodes	Predicted Node Index (PRNI)
50	23
100	23
200	23
500	23
1000	23
2000	23
5000	23
10000	23

VI. COMPARISON WITH TRADITIONAL METHODS

We have compared our proposed multi-objective prediction based method with six other existing methods/techniques based on parameters P1 to P5. The last column is for accuracy. Table 10 shows the comparison table of various task scheduling methods namely Multi-objective Prediction based Method, First Come First Serve (FCFS), Shortest Job First (SJF), Priority, Round Robin (RR), Maxmin, Minmin methods. This table basically summarizes our results by saying that our proposed multi-objective method is Horizontally Scalable, Vertically Scalable and Z Scalable. Also our proposed method is neither dependent on size of the dataset nor on no of decision vectors.

Table 10: Comparison of our proposed method with traditional methods

Methods/Parameters	P1	P2	P3	P4	P5	Accuracy (η)
Proposed Method	YES	YES	YES	NO	YES	YES
FCFS	NO	YES	NO	NA	NO	NO
SJF	NO	YES	NO	NA	NO	NO
Priority based	NO	YES	NO	NA	NO	NO
RR	NO	YES	NO	NA	NO	NO
Maxmin	NO	YES	NO	NA	NO	NO
Minmin	NO	YES	NO	NA	NO	NO

VII. CONCLUSION

Cloud Computing is a simple pay-per-utilize consumer-provider service model. However among various research fields available, Task Scheduling is still one of the major research field in Cloud Computing where several traditional and hybrid methods/techniques were introduced. In this research paper, we designed, developed and investigated multi-objective prediction based method for task scheduling in cloud computing. Experimental results of proposed method are taken with different values of need vector, scheduling vector and current availability vector. The results show that proposed multi-objective method is Horizontally Scalable, Vertically Scalable and also Z Scalable. The results also show that proposed multi-objective method is unbiased towards sparse vectors and supports unbiased scheduling. The predicted node index value is always an optimal value which leads to accuracy of the method. The proposed method also fulfills the objectives like resource utilization and load balancing by giving low probability value for not capable and overloaded nodes. All the results are taken in ideal traffic environment. More experiments can be done on pragmatic environment for more detailed analysis.

The results show that the predicted node index remains unchanged for all need vectors even if scheduling vector and current availability vector remains same. But after changing scheduling vector and current availability vector and keeping same need vector then predicted node index value is changed but for each and every need vector it remains same. So multi-objective prediction based method gives each and every time same result which is optimal value. Basically it gives low probability value for not capable and overloaded nodes. Thus it does proper resource utilization and load balancing. The node index with highest probability value is optimal predicted value of our proposed method.

REFERENCES

1. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, A view of cloud computing, Commun. ACM 53 (4) (2010) 50–58. doi:10.1145/1721654.1721672. URL http://doi.acm.org/10.1145/1721654.1721672
2. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation computer systems 25 (6) (2009) 599–616.
3. S. S. Manvi, G. K. Shyam, Resource management for infrastructure as a service (iaas) in cloud computing: A survey, Journal of Network and Computer Applications 41 (2014) 424–440.
4. O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, D. Tsafrir, The resource-as-a-service (raas) cloud, in: Presented as part of the, 2012.
5. G. E. Gonçalves, P. T. Endo, T. Cordeiro, A. Palhares, D. Sadok, J. Kelner, B. Melander, J. Mangs, Resource allocation in clouds: concepts, tools and research challenges, XXIX SBRC-Gramado-RS.
6. L. Xu, J. Li, Building efficient resource management systems in the cloud: Opportunities and challenges, International Journal of Grid and Distributed Computing 9 (3) (2016) 157–172.
7. Y. Yuan, W.-C. Liu, Efficient resource management for cloud computing, in: System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2011 International Conference on, Vol. 2, IEEE, 2011, pp. 233–236.
8. P. Krishnadoss, P. Jacob, Ocsa: task scheduling algorithm in cloud computing environment, International Journal of Intelligent Engineering and Systems 11 (3) (2018) 271–279.
9. K. Pradeep, T. P. Jacob, Cgsa scheduler: A multi-objective-based hybrid approach for task scheduling in cloud environment, Information Security Journal: A Global Perspective 27 (2) (2018) 77–91.
10. R. Jena, Multi objective task scheduling in cloud environment using nested pso framework, Procedia Computer Science 57 (2015) 1219–1227.
11. G. Pallis, Cloud computing: the new frontier of internet computing, IEEE Internet Computing 14 (5) (2010) 70.
12. P. Mell, T. Grance, The NIST definition of cloud computing.
13. T. Mahiba, et al., Live virtual machine migration in dynamic resource management of virtualized cloud systems, IJLTET.
14. N. Sadashiv, S. D. Kumar, Cluster, grid and cloud computing: A detailed comparison, in: Computer Science & Education (ICCSE), 2011 6th International Conference on, IEEE, 2011, pp. 477–482.
15. I. Foster, Y. Zhao, I. Raicu, S. Lu, Cloud computing and grid computing 360-degree compared, in: Grid Computing Environments Workshop, 2008. GCE'08, Ieee, 2008, pp. 1–10.
16. S. Zhang, H. Yan, X. Chen, Research on key technologies of cloud computing, Physics Procedia 33 (2012) 1791–1797.
17. D. Zissis, D. Lekkas, Addressing cloud computing security issues, Future Generation computer systems 28 (3) (2012) 583–592.
18. V. K. Reddy, B. T. Rao, L. Reddy, Research issues in cloud computing, Global Journal of Computer Science and Technology 11 (11).
19. L.-J. Zhang, J. Zhang, J. Fiaidhi, J. M. Chang, Hot topics in cloud computing, IT professional (5) (2010) 17–19.
20. S. M. Parikh, A survey on cloud computing resource allocation techniques, in: Engineering (NUiCONE), 2013 Nirma University International Conference on, IEEE, 2013, pp. 1–5.
21. R. L. Grossman, The case for cloud computing, IT professional 11 (2) (2009) 23–27.
22. K. Ye, X. Jiang, D. Huang, J. Chen, B. Wang, Live migration of multiple virtual machines with resource reservation in cloud computing environments, in: Cloud Computing (CLOUD), 2011 IEEE International Conference on, IEEE, 2011, pp. 267–274.
23. R. Buyya, Introduction to the IEEE transactions on cloud computing, IEEE Transactions on cloud computing (1) (2013) 3–21.

AUTHORS PROFILE

Dr. Swapnil M Parikh is working as an Associate Professor at Department of Computer Science and Engineering, Babaria Institute of Technology, Varnama, Vadodara affiliated to Gujarat Technological University, Gujarat, India. He received his PhD (Computer Engineering) Degree from C U Shah University, Wadhwan, Surendranagar, India. He has completed his Master of Engineering in Computer Engineering from Dharmsinh Desai University, Nadiad, India and Bachelor of Engineering from Pune University, Pune, India. He authored several research papers which are presented and published by prominent publishers such as IEEE and renowned journals. His research interests include Networking, Distributed Computing, Parallel Computing, Cloud Computing, Machine Learning, Big Data etc.

Dr. Saurabh A Shah received his PhD (Computer Engineering) Degree from R. K. University, Rajkot, India in year 2016. He received his ME(Computer Engineering) Degree from B.V.M., Vidyanagar in 2007 and BE(Computer Engineering) Degree from CCET, Saurashtra University in 2001. He is currently working as a Professor (Computer Engineering) and Director at C. U. Shah University, Wadhwan City, India. His research interests include Digital Image Processing, Medical Image Analysis, Machine Learning, Pattern Recognition and Data Mining. He has authored several research papers which are published in prominent International Journals and conference proceedings like Springer and IEEE. He has also contributed as a reviewer and session chair in several Conferences and International Journals including Springer.

Dr. Narendra M. Patel received Ph.D Degree from SVNIT, Surat in 2012. He received his ME Degree from M. S. University, Baroda in 1997 and BE Degree in Electronics Engineering from M.S. University, Baroda in 1993. He is currently working as an Associate Professor in Computer Engineering Department, B.V.M. Engineering College, V.V.Nagar, India. His research interests include Digital Image Processing, Real Time Operating Systems, Distributed System and Computer Graphics. He authored more than 40 research papers which are published in prominent international journals and conference proceedings. He has guided more than 50 Masters Dissertations in Computer Engineering.