A Survey on Task Scheduling and Load Balanced Algorithms in Cloud Computing

T. Kannadasan[1], Dr. R. Pragaladan[2]

[1]Associate Professor and Head, Department of Computer Science, Thanthai Periyar Government Arts and Science College (Autonomous), Tiruchirappalli
[2]Assistant Professor and Head, Department of Computer Science, Sri Vasavi College, Erode

---***---

***Abstract*:- Cloud Computing is a recent developmental paradigm in the field of computing offering huge power to next-generation computers. The dynamic provisioning acts as a base for cloud computing facilitating and supporting the network services. It focuses on making the vision of utility computing a reality with pay-as-you-go. It offers immense potential to bloom the world with applications and products focusing on greater resource utilization and scalability. This paper presents the survey on the basics of cloud computing, the concepts of load balancing, and the scheduling of tasks in the cloud. It elaborates on the existing load scheduling algorithms with their merits and demerits, suitability in the cloud, heterogeneous computing environment, and proposes a new perspective for better results as per desired parameters.**

***Keywords:* Load balancing, Cloud Optimization, Response time, Scheduling Algorithms.**

1. INTRODUCTION

1. 1. Load balancing

Load balancing is the reapportion of the workload equally across all processors, so that no processor is overloaded. A load balancer is a physical device, running on a specialized hardware or software process and it accepts multiple requests from users and distributes them evenly across servers [20]. Load balancing increases throughput and thereby reduces response time. Load balancing in clouds, distributes the excess dynamic local workload evenly across all the nodes. It ensures better service provisioning and resource utilization ratio, and in turn, improves the overall performance of the system. Incoming tasks, which are received from different locations, are received by the load balancer and are then distributed to the data center for the proper load distribution. Figure (1) shows the typical load balancing mechanism.

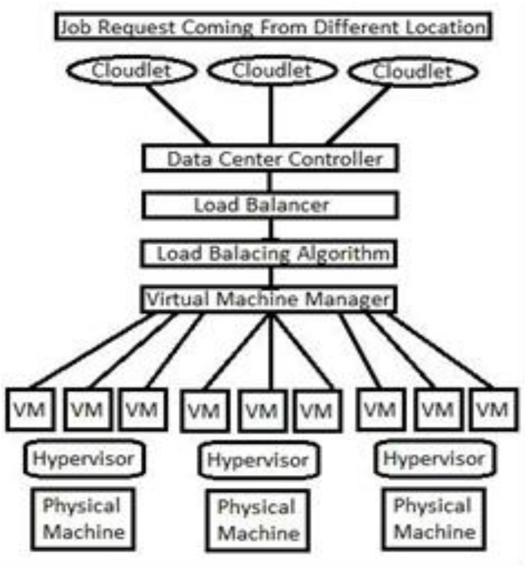

Fig. 1. The Load balancing Mechanism

1. 2. Challenges of Load Balancing Algorithm

1. 2. 1. Overhead: Any load balancing system should consider the amount of overhead involved in the implementation of a load balancing system [1]. It consists of overhead due to VM migration cost or the cost of communication. A well-designed load balancing algorithm should reduce overhead.

1. 2. 2. Performance: It denotes the efficiency of the system. Performance can be ascertained from users experience and satisfaction [1]. Ensuring optimum performance is the foremost challenge for VM load balancing algorithms. The performance includes the following perspectives:

1. 2. 2. 1. Resource utilization – It is a measure of whether a host is overloaded or underutilized [1]. Many VM load balancing algorithms prefer offloading of overloaded hosts with higher resource utilization.

1. 2. 2. 2. Scalability – The quality of service delivered is kept smooth and without lag, even when the number of users increases irrespective of the algorithm management approach, like centralized or distributed.

1. 2. 2. 3. Response time – The quantity of time to react by a load balancing algorithm upon input parameters in a cloud system. The response time should be reduced for better performance.

1. 2. 2. 4. The point of failure – One of the concerns in the system design is that a single point failure does not affect the provisioning of services. This issue prevails in centralized systems, when the central node fails, then the whole system conk out. So, load balancing algorithms should be designed with high availability to overcome this problem.

1. 2. 3. Load balancing in cloud – The cloud is referred to as a subscription based service where computing resources and storage space in the network can easily be accessed. The clouds act as virtualized data centers [1, 22]. The platforms used in cloud computing are dynamically built using virtualized software, hardware, data, and networks. Thus, cloud computing is a new and upcoming computing paradigm that supports data and computational outsourcing [27]. Figure (2) shows the typical architecture of load balancing in cloud environment.

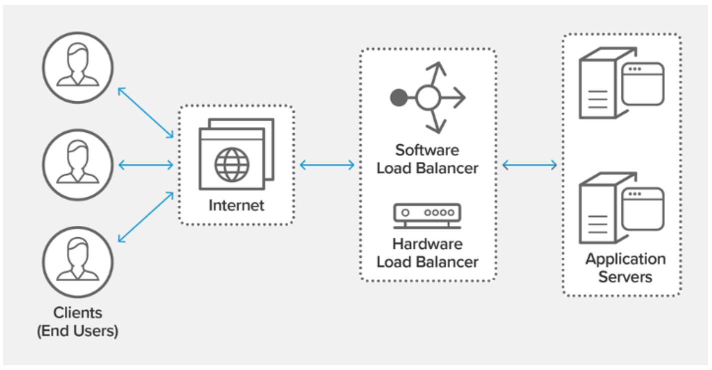

Fig. 2. The Architecture of Load Balancing in Cloud environment

Cloud computing is a scalable and distributed system requiring the allocation of resources to several users. This could lead to congestion or imbalanced allocation of the system [3]. So, a load balancing strategy is needed to deal with the imbalance in the network. Load balancing is the mechanism used for distributing the load for optimal resource utilization on the system processes or virtual machines. The load refers to the task needed to be done on the system and balancing refers to the handling or the management of the load. Load balancing is applied to the resources in the system which can be disks, drivers, memory buffers, processors, network simulators, etc. [4]. The load balancing is performed and applied to achieve minimum response time, maximum throughput, and the reduction of the overheads produced by the system. The main aim is to process the user's request by the efficient, and effective utilization of the resources present in the system [23].

3. LOAD SCHEDULING

Load scheduling thereby the load balancing is attained, is a computer networking technique to distribute workload across multiple computers or computer clusters, disk drives, networks, [28] central processing units (CPUs) and other resources incited due to scalability for achieving higher throughput, optimal resource utilization, lesser response time, and avoiding overloading of the system[1]. Using multiple components with load balancing [14] instead of a single component, increases the reliability with the help of redundancy. The task of providing the load balancing service is done by dedicated software and hardware, such as a Domain Name System server. Load balancing plays an important role in the distributed computing system. The load distribution in a network is performed among servers. Load distribution is the method of reallocating the total available load to the specific and single node available on the network for efficient utilization of resources, improving the response time of the task, and removing the situation of node overloading as well underloading in the system. The load balancer is a software service that listens on the port to avail services by connecting with external users. The load balancer intern forwards requests to the servers located at the backend and the feedback data is passed to the load balancer and then back to the user. The user receives the reply from the load balancer, but the user is unaware of the internal separation or abstraction of functions [12, 15]. It also prevents the user's direct contact with the backend servers, thereby increasing the benefits obtained from security by the inner network's structural abstraction, and preventing attacks on the service ports and network stack of the kernel.

3. 1. Major goals of the load balancing:

a) To enhance the system performance.

b) To have a backup plan ready if any failure occurs in the system.

c) To uphold the system stability.

d) To allow further modifications in the system.

e) To distribute the load effectively and obtain cost-effectiveness.

4. LOAD SCHEDULING ALGORITHMS

The load scheduling, as well as the task scheduling algorithms that are currently available in the cloud environment, are discussed below and tabularized in Table (1).

4. 1. *Max-Min Scheduling Algorithm*

In this algorithm, firstly a set of minimum execution time M is computed. The task having a total maximum completion time amongst elements of *M* is selected and allocated to the corresponding machine is called the Max-min strategy. The recently mapped task, *T* is removed from the set *M* and the procedure is repeated till the system maps the remaining tasks [1, 16, 19]. The minimizing algorithm attempts to minimize the degree of shortcomings incurred for performing tasks with longer execution times. It offers better mapping than the shorter tasks that get executed first, followed by the execution of long-running tasks while several machines remain idle waiting for resources. Thus, the Max-min heuristic provides mapping with a more nicely balanced load across machines and a better makespan (completion time).

4. 2. *Min-Min Scheduling Algorithm*

It is the simplest algorithm enduring as a base for existing [1, 16, 21, 30] cloud scheduling algorithms. It is fast and provides higher and better performance than others by scheduling by considering the best case first. It begins using a set of completely unmapped tasks, *S*. Then, the resource *R* having a minimum completion time for all tasks is found and the task *T* with the minimum size is selected and assigned to the corresponding resource *R*. Lastly, from the set of unmapped tasks, *S* executed task, *T* is removed and the process goes on repeating using Min-min algorithm till all tasked are finished. Assuming [29] we have a set of *n* tasks (T_n) that need to be scheduled onto *m* available resources (R_m). We denote the Expected Completion Time for task *i* starting from *1* to *n* on resources *j* from *1* to *m* as CT_{ij} and RT_j represents the Ready Time of resource R_j. ET_{ij} represents the Execution Time of the task *Ti* on resource R_j. So, the completion time is $CT_{ij} = ET_{ij} + RT_j$.

4. 3. *Segmented Min-Min Scheduling Algorithm*

This algorithm sorts the tasks based on ETCs (Estimated Time to Complete) [17]. The tasks are sorted based on the sorting key using an ordered list by the trade-off factor *N* in average ETC, minimum ETC, or the maximum ETC. After this, segments with equal size using a trade-off factor *N* are created using the task list-partitioning scheme. The larger task segments are scheduled and executed first followed by the smaller tasks i.e. in decreasing order. Min-min is applied to assign tasks to machines for every segment. The task sorting is done before scheduling so that the larger tasks are scheduled earlier.

4. 4. *A* Scheduling Algorithm*

It is a search technique based on a binary tree beginning at the root node (null solution). As the tree propagates, nodes represent partial mappings where a subset of the tasks is assigned to machines [1]. The partial mapping (solution) of a child node has one more task mapped than the parent node. The children are generated by the parent node with the possible additional task's mapping (t_a) and then the parent node becomes inactive. Pruning is applied to keep track of the execution time and at a time restriction on the maximum number of tree's active nodes. Each node contains a cost function. Its children replace the node having a minimum cost function. This process continues until a complete mapping is done or a leaf node is located.

4. 5. *Heterogeneous Earliest Finish Time Algorithm (HEFT)*

This algorithm determines the average execution time of the tasks and average communication time among resources of the consecutive tasks [24]. Then, the algorithm orders the tasks, based on the rank function in the workflow i.e. higher rank value task is awarded higher priority. The scheduling of tasks occurs depending on the priority order and resources are assigned to tasks to complete them at the earliest time in the allocation phase.

4. 6. *Multiple QoS Constrained Scheduling Strategy of Multi-Workflows ((MQMW)*

This algorithm works on multiple workflows and many Quality of Service parameters [6]. This strategy involves minimizing the makespan (the time difference between the beginning and end of a job or task sequence) and the cost of workflows for a cloud computing platform. This algorithm also leads to an increase in the scheduling access rate in the system.

4. 7. *A Double Min-Min Algorithm for Task Scheduling*

In this algorithm, a priori is used in a heterogeneous environment, which is defined, by the meta-tasks size and the number of machines [4, 5]. These are static heuristics, therefore on each machine, the task expected execution time for accurate estimation is known to be priorly and stored in an ETC matrix where ETC (t_i, m_j) denotes the estimated execution time of the task *i* on machine *j*. The scheduling is accomplished using the Min-min approach. In this, we allocate a number of independent jobs to the available resources, there is a sufficient number of machines available for allocating tasks and workload and processing capability of each job and resources (in millions of instructions per second MIPS) are taken into account.

4. 8. *QoS Guided Min-Min Algorithm*

This algorithm adds QoS (Quality of Service) constraints like bandwidth, time, and memory [2, 10] to the basic Min-min heuristic. In this, some tasks require high network bandwidth, whereas others could be contented with low network bandwidth. It assigns tasks having high QoS request parameters first similar to the Min-min heuristic. In the worst case, all tasks generally need either low QoS or high QoS.

4. 9. *Improved Cost-Based Algorithm for Task Scheduling*

This algorithm is used for efficient mapping of tasks to resources in the cloud [8, 10]. It measures the computational performance and the resource cost incurred in the system, thereby helping in the improvement of the computational or communication ratio.

4. 10. *Optimized Resource scheduling algorithm*

This algorithm aims to obtain complete optimization or sub-optimization in the cloud. It supports multiple instances in the cloud for processing the user requests, bewaring the cost and performance. It supports an automated scheduling policy and uses an Improved Genetic Algorithm (IGA) [14, 31] to increase the resource utilization rate and speed of execution in the system.

4. 11. *A Particle Swarm Optimization-based Heuristic for Scheduling Workflow Applications*

This algorithm uses particle swarm optimization-based heuristics to schedule applications to the resources present in cloud by taking into consideration the cost of computation and the data transmission [7, 18]. It is used for a workflow application by decreasing the applications' computational as well as communication costs with significant cost savings and optimal workload distribution of the resources in the system.

4. 12. *Resource Aware Scheduling Algorithm (RASA)*

This algorithm is based on Max-min and Min-min task scheduling algorithms and is referred to as Resource Aware Scheduling Algorithm (RASA) [8]. It utilizes the benefits of both Max-min and Min-min algorithms and hides their disadvantages by outperforming the existing scheduling algorithms in large-scale distributed systems and then using the better solution [16]. It is adjusted to exploit the conditions, with minimal overhead and higher performance either by Max-min or Min-min.

4. 13. *Round Robin Algorithm*

It is a static and decentralized algorithm [29] used on web servers. The processes are divided among the processors in a round-robin fashion using a particular timestamp. The allocation order of the resources to the processors is locally maintained irrespective of remote ones. They possess equal distribution of workload among the processors and different processing times of tasks for processors. But the pay-off is that nodes become overloaded and times become underloaded [13].

4. 14. *Weighted Round Robin Algorithm*

In this algorithm, the weights are assigned in a particular order to every task in the system to allocate resources for optimal utilization of resources [13, 29]. After assigning the weights, the allocation of the data centers occurs depending on the time quantum or time slot in the round-robin fashion.

4. 15. *Scalable Heterogeneous Earliest Finish Time Algorithm (SHEFT)*

This algorithm schedules the workflow elastically on a cloud computing environment with the benefits of the Heterogeneous Earliest Finish Time algorithm (HEFT) which also induces scalability [6, 25]. The parameters act as a means for scheduling the resources and tasks. The scalability is generally determined by adding a large number of systems for resources. It outpaces other workflow scheduling algorithms to obtain an optimized execution time and provides the ability to elastically scale resources at runtime.

4. 16. *A Compromised Time-Cost Scheduling Algorithm*

This algorithm [20] considers the characteristics of cloud computing to provide multiple instances for performing computation on the system and implementing intensive cost constraint workflows by compromising or lessening the execution time and cost with the help of the user input enabled on the fly.

4. 17. *Optimal Workflow based Scheduling (OWS) Algorithm*

This algorithm schedules the workflow in a cloud environment by finding a solution by meeting the quality of service like bandwidth, memory constraints used by the user and performs optimal workflow execution [9, 15]. Thereby, achieving an important improvement and better results in terms of CPU utilization and the costs incurred.

EXISTING LOAD BALANCING (SCHEDULING) ALGORITHMS

Table 1. The observations and strategies handled by the existing scheduling algorithms

Algorithm	Method Used	Allocation Strategy	Observations
Min-Min Scheduling Algorithm [1, 30]	Many cases used, Minimum completion time	Tasks allocated resources for execution	1. Resource cost and computational performance is calculated. 2. Higher utilization rate of resources
Max-Min Scheduling Algorithm [1, 19]	Multiple cases, Speed, Maximum completion time	Resources allocated to tasks	1. Computational performance and resource cost are evaluated. 2. Greater resource utilization.
Segmented Min-Min Scheduling Algorithm [17]	Multiple cases, Speed, Resource Utilization	Request provisioning, Segmentation	1. Measures both resource cost and computation performance 2. The utilization rate of resources is high.
A* Scheduling Algorithm [1]	Group approach, Resource Utilization	Unscheduled task groups, Pruning is done	1. Complete the task at the earliest time. 2. Obtaining lower makespan and cost. 3. Better allocation.
Multiple QoS Constrained Scheduling Strategy of Multi Workflows [6, 15]	Segments are made, Resource Utilization	Quality of Service, Constraints are considered	1. Optimizes execution time. 2. Allows elastic scaling of resources in workflow execution.
A Double Min-Min Algorithm for Task Scheduling [4]	Various Cases, Speed, Utilization	Request allocation similar to Min-min	1. Measures both resource cost and computation performance. 2. The utilization rate of resources is high.
QoS Guided Min-Min Algorithm [2]	Segments are made, Resource Utilization	Quality of Service, Constraints considered	1. Similar to Min-min algorithm with sub-policies. 2. Resources are scaled elastically in execution with a higher utilization rate.
Optimized Resource Scheduling Algorithm [31]	Multiple instances, Speed, Resource Utilization	Allocation of resources as requested by the user	1. Speed is higher. 2. The Genetic Algorithm improves resource utilization rate.
Improved cost-based algorithm for task scheduling [10]	Multiple cases, Cost, Performance	Task groups are unscheduled	1. Measures both resource cost and computation performance. 2. Improves the computation/communication ratio.
A compromised Time Cost Scheduling Algorithm [20]	Many cases, Cost and time, Constraints of cost applied	Job cases are created by considering service level scheduling	1. Optimizes and reduces cost and time. 2. Reduces the execution time.
Heterogeneous Earliest Finish Time algorithm (HEFT) [25]	Dependency Method, Execution time, Scalability	Group of tasks ordered by rank function	1. Optimizes execution time. 2. Resources are scaled elastically during execution.

Round Robin Algorithm [13, 29]	Jobs made, Cost, Weights assigned to each task	Jobs assigned to resources based on time value	1. Reduces cost and time by round-robin fashion. 2. Reducing the debt or cost.
Resource Aware Scheduling algorithm [8, 16]	Many cases used, Uses both Min-min & Max-min Scheduling	Execution time and completion time of tasks	1. Measures both resource cost and computation performance. 2. The utilization rate of resources is high.
Weighted Round Robin Algorithm [13, 29]	Jobs created, Weights are Assigned to each task, Cost and time	Jobs assigned to resources based on time value and weights	1. Reduces cost and time due to weighted stamps. 2. Weights reduce the debt or cost.
A Particle Swarm Optimization based Heuristic Algorithm [18]	Dependency mode, Resource utilization, Time	Demand distribution strategy, Tasks are grouped	1. Offers higher cost savings. 2. Good distribution of the workload onto resources is performed.
SHEFT workflow scheduling algorithm [6, 25]	A reliable method, Execution time, Scalability	Group of tasks ordered by rank function	1. Optimizes execution time. 2. Resources are scaled elastically during execution.
Optimal Workflow Scheduling Algorithm [9]	Creation of Virtual clusters, Makespan, Cost, CPU time	Scheduling based on service level and quality of service constraints	1. Minimizes the overall running cost. 2. Optimizes the makespan and cost.

5. CONCLUSION AND FUTURE WORK

In this appraisal paper, we briefly discoursed the load and task scheduling algorithms implemented in various heterogeneous cloud servers. These algorithms are constrained by assorted scheduling parameters and strategies. For example, a higher utilization rate was achieved by using min-min, segmented min-min, double min-min, and max-min algorithms. The ETC factor completes a task at the earliest time and the weighted round-robin reduces computation cost. The exploration was served for greater resource utilization, reduced cost and debt to achieve maximum throughput, and higher performance. Even though the existing load balancing techniques doing well but not enough to fulfill the current demand from the different cloud environments. So, in future work, we have to concentrate on the existing problems to suss out these in the cloud computing environment with better acumination subjective to resource constraints. The future proposed paper must be designed to improve the optimized load balancing in cloud computing environment for efficient utilization of resources, the improved response time of tasks and refining the situation of node congestion, overloading, and underloading in the distributed cloud environment.

6. REFERENCES

[1] Braun, Tracy D, et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", in Journal of Parallel and Distributed computing, Volume 61, Issue 6, Pages 810-837, 2011.

[2] He, XiaoShan, XianHe Sun and Gregor Von Laszewski, "QoS guided Min-min heuristic for grid task scheduling", in Journal of Computer Science and Technology, Volume 18, Number 4, Pages 442-451, 2013

[3] Wang, Shu-Ching, et al., "Towards a load balancing in a three-level cloud computing network", in 3rd IEEE International Conference Computer Science and Information Technology (ICCSIT), Vol. 1, Pages 108-113, 2019.

[4] DDH Miriam, KS Easwarakumar, "A Double Min-min Algorithm for Task Metascheduler on Hypercubic P2P Grid Systems", in International Journal of Computer Science Issues, Vol. 7, Issue 4, Pages 8-18, 2010.

[5] Kong, Xiangzhen, et al, "Efficient dynamic task scheduling in virtualized data centers with fuzzy prediction", in Journal of Network and Computer Applications, Vol. 34, Issue 4, Pages 1068-1077, 2011.

[6] Hirales-Carbajal, Adán, et al., "Multiple Workflow Scheduling Strategies with User Run Time Estimates on a Grid", in Journal of Grid Computing, Volume 10, Number 2, Pages 325-346, 2012.

[7] W. Chen, J. Zhang, "An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem with Various QoS Requirements", in IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, Vol. 39, No. 1, January 2009, Pages 29-43.

[8] Saeed Parsa, Reza Entezari-Maleki, "RASA: A New Task Scheduling Algorithm in Grid Environment", in Special Issue of Computer & IT, World Applied Sciences Journal, Vol. 7, Pages152-160, 2009.

[9] J. Yu and R. Buyya, "Workflow Scheduling Algorithms for Grid Computing", in Technical Report, GRIDS-TR-2007-10, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, May 2017.

[10] Mrs. S. Selvarani, G. and Mr. S. Sadhasivam, "Improved cost-based algorithm for task scheduling in Cloud computing", in IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Pages 1-5, 28-29 December 2010.

[11] Y. Yang, K. Liu, J. Chen, X. Liu, D. Yuan, H. Jin, "An Algorithm in SwinDeW-C for Scheduling Transaction-Intensive Cost-Constrained Cloud Workflows", in Proceedings of 4th IEEE International Conference on E-Science, Pages 374-375, USA, December 2018.

[12] Meng Xu, Lizhen Cui, et al., "A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing", in IEEE International Symposium on Parallel and Distributed Processing with Applications, 10-12 August 2009, Pages 629-634.

[13] Bhathiya Wickremasinghe, Rodrigo N. Calheiros and Raj Kumar Buyya, "CloudAnalyst: A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications", in 24th IEEE International Conference on Advanced Information Networking and Applications, 20-23 April 2010, Pages 446-452.

[14] A. Khiyaita, El Bakkali, M. Zbakh, Dafir El Kettani, "Load Balancing Cloud Computing: State of Art", in IEEE Transactions on Software Engineering, 978-1-4673-1053-6, 2012, Pages 106-109.

[15] Cui Lin, Shiyong Lu, "Scheduling Scientific Workflows Elastically for Cloud Computing", in IEEE 4th International Conference on Cloud Computing, 2011.

[16] K. Etminani and M. Naghibzadeh, "A Min-min Max-min selective algorithm for grid task scheduling", in 3rd IEEE/IFIP International Conference in Central Asia, 26-28 September 2017, Pages 1-7.

[17] M. Wu, Wei Shu and Hong Zhang, "Segmented Min-min: A static mapping algorithm for meta tasks on heterogeneous computing systems", in HCW 9th Heterogeneous Computing Workshop, Page 375, Washington, DC, USA, 2000, Pages 375-385. IEEE Comp. Society.

[18] Suraj Pandey, Rajkumar Buyya, et al., "A Particle Swarm Optimization based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments", in 24th IEEE International Conference on Advanced Information Networking and Applications, 20-23 April 2010, Pages 400-407.

[19] M. Hosaagrahara and H. Sethu, "Max-min Fair Scheduling in Input-Queued Switches", in IEEE Transactions On Parallel and Distributed Systems, Vol. 19, No. 4, Pages 462-475, April 2018.

[20] Sen Su, Jian Li, Qingjia Huang, et al., "Cost-efficient task scheduling for executing large programs in the cloud", in Journal of Parallel Computing, Elsevier, Vol. 39 Pages 177-188, 2013.

[21] Doulamis et al., "Fair Scheduling Algorithms in Grids", in IEEE Transactions on Parallel and Distributed Systems, Vol. 18, No. 11, November 2007, Pages 1630-1648.

[22] Luiz F. Bittencourt, Edmundo R. M. Madeira and Nelson L. S. da Fonseca, "Scheduling in Hybrid Clouds", in IEEE Communications Magazine, September 2012, Pages 42-47.

[23] Chun-Wei Tsai and Joel J. P. C. Rodrigues, "Metaheuristic Scheduling for Cloud: A Survey", in IEEE Systems Journal, Vol. 8, No. 1, March 2014, Pages 279-291.

[24] Haluk Topcuoglu et al., "Performance Effective and Low Complexity Task Scheduling for Heterogeneous Computing", in IEEE Transactions on Parallel and Distributed Systems, Vol. 13, No. 3, Pages 260-274.

[25] Dr. A. Shaik Abdul Khadir, V. Manochitra (2021). Decision making of milk processing system using automated rule mining technique. D*esign engineering*, 4775-4787 retrieved from http://www.thedesignengineering.com/index.php/de/article/view/2917

[26] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey", in IEEE Journal of Computing, Vol. 27, No. 6, June 1994, Pages 17-26.

[27] http://en.wikipedia.org/wiki/Cloud_computing

[28] http://en.wikipedia.org/wiki/Load_balancing_(computing)

[29] Divya Chaudhary and RS Chhillar, "A New Load Balancing Technique for Virtual Machine Cloud Computing Environment", in International Journal of Computer Applications 69 (23) Pages 37-40, May 2013.

[30] Huankai Chen, Wang, F., Helian, N., Akanmu, G, "User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing", in 2013 IEEE National Conference on Parallel Computing Technologies, 21-23 Feb. 2013, Pages 1-8, ISBN 978-1-4799-1589-7.

[31] Dr. A. Shaik Abdul Khadir, V. Manochitra. (2021). A survey on optimized milk distribution process using hybrid rule generation techniques in milk big data servers. *Design engineering*, 308-313 retrieved from http://www.thedesignengineering.com/index.php/de/article/view/2277.

[32] Dr. K. Saravanan, T. Thamaraiselvan (2021). An efficient clustering on hybrid item dependency using SCFCM and SVM techniques. *Design engineering*, 2275-2286 retrieved from http://www.thedesignengineering.com/index.php/de/article/view/2596.

[33] Cui Lin and Shiyong Lu, "Scheduling Scientific Workflows Elastically for Cloud Computing", in 2011 IEEE 4th International Conference on Cloud Computing, Pages 746-747, 2011.

[34] Hai Zhong, Kun Tao, Xuejie Zhang, "An Approach to Optimized Resource Scheduling Algorithm for Open-source Cloud Systems", in Fifth Annual China Grid Conference, Pages 124-129, 16-18 July 2010.

BIOGRAPHIES

[1]Mr. T. Kannadasan, designated as Associate Professor in Computer Science in the Tamil Nadu State Government Collegiate Education Service from October 1998. Pursuing the Ph.D. Degree in Computer Science (Part-Time) in Bharathiar University, Coimbatore under the guidance and supervision of Dr.R.Pragaladan in Cloud Computing.

[2]Dr. R. Pragaladan conferred the Ph.D. Degree in Computer Science by Bharathiar University, Coimbatore through Erode Arts and Science College under Faculty Development fellowship Programme (FDP) for Teacher by UGC, Erode Tamilnadu, India in the year 2018. Currently, he is designated as Assistant Professor in Computer Science and Head of the Department of Computer Science, Sri Vasavi College, Erode affiliated with Bharathiar University. He has been a supervisor and mentor for several students in B.Sc. (CS), M.Phil., and Ph.D. programs all around for the past 14 years. He has vast experience and published 5 papers at Conferences and more than 15 papers in reputed Journals. His principal domain of fascination includes Cloud Computing, Internet of Things, Bio-Informatics, Cyber Security, and Computer Networks.