Applied Soft Computing Journal 76 (2019) 416–424

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Resource scheduling algorithm with load balancing for cloud service provisioning

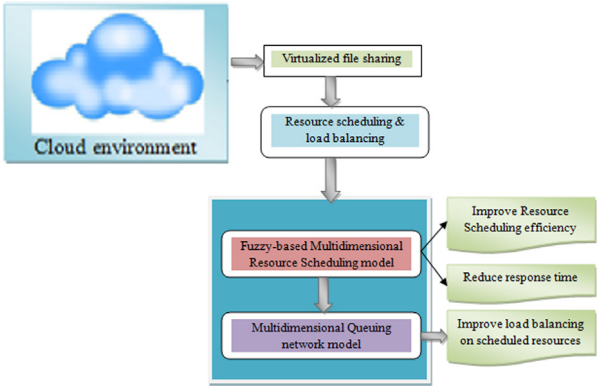
V. Priya [a,*], C. Sathiya Kumar [b], Ramani Kannan [c]

[a] Department of Computer Science Engineering, Mahendra Institute of Technology, Namakkal, 637215, India
[b] School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India
[c] Department of Electrical & Electronic Engineering, Universiti Teknologi PETRONAS, Bandar Seri Iskandar, 32610, Perak, Malaysia

HIGHLIGHTS

- FMRS algorithm for minimizing response time on handling complex query.
- Multidimensional queuing network model achieves load balancing in cloud after scheduling the resource.
- Average success rate is improved with the aid of MQLO algorithm.
- Both FMRSQN helps for efficient data sharing in cloud environment with low computational complexity.

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 16 August 2017
Received in revised form 17 October 2018
Accepted 12 December 2018
Available online 27 December 2018

Keywords:
Cloud computing
Load balancing
Multidimensional resource scheduling
Queuing load optimization scheduling
Virtual machine

ABSTRACT

Cloud computing uses scheduling and load balancing for virtualized file sharing in cloud infrastructure. These two have to be performed in an optimized manner in cloud computing environment to achieve optimal file sharing. Recently, Scalable traffic management has been developed in cloud data centers for traffic load balancing and quality of service provisioning. However, latency reducing during multidimensional resource allocation still remains a challenge. Hence, there necessitates efficient resource scheduling for ensuring load optimization in cloud. The objective of this work is to introduce an integrated resource scheduling and load balancing algorithm for efficient cloud service provisioning. The method constructs a Fuzzy-based Multidimensional Resource Scheduling model to obtain resource scheduling efficiency in cloud infrastructure. Increasing utilization of Virtual Machines through effective and fair load balancing is then achieved by dynamically selecting a request from a class using Multidimensional Queuing Load Optimization algorithm. A load balancing algorithm is then implemented to avoid underutilization and overutilization of resources, improving latency time for each class of request. Simulations were conducted to evaluate the effectiveness using Cloudsim simulator in cloud data centers and results shows that the proposed method achieves better performance in terms of average success rate, resource scheduling efficiency and response time. Simulation analysis shows that the method improves the resource scheduling efficiency by 7% and also reduces the response time by 35.5 % when compared to the state-of-the-art works.

1. Introduction

In the field of information technology (IT), cloud computing plays a vital role. Due to the distributed nature of computing, cloud

* Corresponding author.
E-mail addresses: priya.saravanaraja@gmail.com (V. Priya), csathiyakumar@yahoo.com (C. Sathiya Kumar), kreee82@gmail.com (R. Kannan).

https://doi.org/10.1016/j.asoc.2018.12.021
1568-4946/

computing environment is receiving greater attention in the area of research communities. Cloud computing is an internet-based computing model by Kaur and Luthra [1] that assists in efficient sharing of the computing resources (hardware and software) or services over a network with low cost. Cloud infrastructure enables virtualized file sharing with the aim of developing efficient load balancing and resource scheduling. Cloud users access the resources and files stored on the server through virtualization. In this scenario, an efficient resource scheduling and load balancing remain the major part to be concentrated for efficient file sharing.

Most of the research works have been developed for resource scheduling and load balancing in the cloud. For example, scalable traffic management (STM) in cloud was addressed using a novel decomposition approach by Assi et al. [2]. With the help of STM, the maximum link load was found to be reduced and therefore ensuring load balancing between users in the network. However, this approach was not suitable for multidimensional resource scheduling. To address this issue, a scalable workload driven partitioning scheme was presented by Ahirrao and Ingle [3] aiming at improving the response time and throughput for distributed transactions. Yet another work by Dhinesh Babu L.D. et al. [4] provided an insight into task scheduling in cloud environment using Honey Bee behavior. Honey bee behavior was used to attain optimal machine utilization. Foraging behavior of honey bees are employed to effectively balance the load across VM in a cloud environment.

Cloud computing has received considerable amount of attention and is accepted as an ultimate way of managing and improving data utilization and resources and delivering various computing and IT services. The performance and power constrained load distributed methods were investigated by Cao et al. [5] to provide new insights into data center optimization. However, resource management remains unaddressed. In order to overcome this, performance evaluation for resource management was performed by Batista et al. [6] by ensuring quality of service.

A Long Term Evolution (LTE) algorithm was deliberate to provide green technology in smart grid by Hindia et al. [7]. However, integrity of virtual machine and overhead remained a major concern. Therefore, [8] presented an Advanced Cloud Protection System (ACPS) to reduce the overhead during file sharing.

Cloud computing is a new found technology delivering resources and application program to cloud users. Hesabian and Javadi [9] investigated a honey bee colony algorithm that involved a group based on search solution, improving the scheduling efficiency. However, load balancing remained unsolved. To solve this issue, Energy efficient load balancing was designed by Abdulmohson et al. [10] for virtual machine distribution across servers.

To improve the resource performance, a task scheduling heuristic was investigated by Singh and Bawa [11]. Game theoretic mechanisms were introduced by Rzadca et al. [12] using centralized model. However, the major criteria called, task scheduling was not addressed. Georgios L. Stavrinides et al. [13] introduced EDF, HLF and LSTF algorithms to schedule multiple tasks in a homogeneous distributed real-time system. The main objective of algorithm was to achieve high quality (precise) results and provide guarantee for all jobs which arrive in the system.

Under various circumstances, researchers have put forward resource scheduling with load balancing algorithms. Agent based load balancing algorithm was presented by Singh et al. [14] with the aid of an autonomous model to provide dynamic load balancing for cloud environment. Though load balancing was ensured, optimization was not said to be achieved. In order to overcome this problem, [15], budget-driven scheduling algorithms were designed using Global Greedy Budget (GGB) and Gradual Refinement (GR) ensuring performance optimizations.

With the emergence of cloud infrastructure, there has been an increasing trend toward geographically distributed data centers. Therefore, energy efficient cloud servers and electricity prices are both of highly importance in minimizing the energy cost. An online scheduling algorithm was designed by Polverini et al. [16] to optimize the energy cost and fairness among different organizations. Zhang et al. [17] presented K-Means Cluster algorithm that dynamically adjusted the number of machines to reduce energy consumption and delay. A Hyper Heuristic Scheduling algorithm was designed Chun-Wei Tsai et al. [18] aiming at improving the scheduling efficiency.

Load balancing is said to be one of the means through while power optimization is achieved. Optimal power allocation and load distribution were addressed [5] for multiple servers in cloud using queuing system. Dynamic multi-server load balancing was designed using an effective load balancing algorithm by Lin et al. [19]. The work of Ferreira et al. [20] and Neto et al. [21] designed an optimized load distribution algorithm to optimize data center. For efficient data sharing, an object-centered approach using novel highly decentralized information accountability framework was designed by Sundareswaran et al. [22] improving scheduling efficiency. Resource allocation in a heterogeneous manner was discussed by Olivier Beaumont et al. [23] using greedy heuristic method, ensuring throughput and cost.

Yet another scheduling and load balancing algorithm were designed by Chitra Devi and Rhymend Uthariaraj [24] with the abilities of all virtual machine (VM), the task length of all requested job and interdependency of many tasks. Heterogeneous resources were controlled by assigning tasks to suitable resources through static or dynamic scheduling and increase the user satisfaction level. However, security, one of the main issues in cloud remained unaddressed. A secure method was planned by Zhongma Zhu et al. [25] for key distribution without private communication channels. A fine-grained access control was also said to be carried out with any user in the group in the cloud and revoked users failed to access the cloud.

A workload balancing and resource management for Swift was presented by Wang et al. [26] in distributed storage system on the cloud. The workload monitoring and analysis algorithms were designed for evaluating overloaded and underloaded nodes. Resource Reallocate Algorithm was made to control the virtual machines on cloud. Resource Scheduling of Cloud Based on Improved Particle Swarm Optimization Algorithm was intended by Wang et al. [27]. Depending on the features of cloud computing resources subject to time and budget requirements of users, scheduling model of resource with particle swarm optimization algorithm were designed.

The maximum resource utilization and minimum processing time are the main concern in the cloud infrastructure. These kinds of issues are addressed by introducing Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) method in cloud services. In the proposed F-MRSQN method, fuzzy-based scheduler helps to minimize the response time of arriving job submissions. The efficiency of resource scheduling is enhanced by performing fuzzy-based multidimensional resource scheduling. The aim of load balancer is to enhance the resource scheduling efficiency and average success rate of the job allocation process. The Fuzzy-based Multidimensional Resource Scheduling is developed with Fuzzy Square inferences subject to maximum resource utilization and minimum processing time and designs an algorithm Multidimensional Queuing Load Optimization. The contribution of the research work is illustrated as follows.

- Fuzzy-based Multidimensional Resource Scheduling (MRS) algorithm is proposed with Fuzzy Square inference that associates cloud user query based on the fuzzy rule, therefore minimizing response time while handling the complex query.
- A novel multidimensional queuing network model is implemented for load balancing in cloud infrastructure. It can effectively improve resource scheduling efficiency and average success rate using Multidimensional Queuing Load Optimization algorithm.

- Finally, with the design of Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN), efficient data sharing in a cloud environment is achieved reducing the computational complexity.

The rest of the paper is organized as follows. Section 2 defines the problem and presents the system overview with the aid of block diagram and the technical details of the proposed F-MRSQN method. Section 3 describes the simulation setup and performance evaluation is provided in Section 4. Finally, Section 5 concludes the paper.

2. Fuzzy-based multidimensional resource scheduling and queuing network in cloud data centers

One of the most primary services offered by cloud providers is file sharing on dynamic cloud environment. In order to improve the resource scheduling efficiency and response time for cloud users, a new simple method called Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) is developed in cloud data centers. The structural framework of F-MRSQN is shown in Fig. 1.

The F-MRSQN method includes three stages to be performed between the users and the servers in cloud environment: (1) obtains incoming requests from the cloud users, (2) online Fuzzy-based Multidimensional Resource Scheduling by resource manager, (3) perform load optimization using Multidimensional Queuing Network.

As shown in Fig. 1, the F-MRSQN method acquires the incoming requests with the objective of maximizing resource utilization and minimizing processing time based on vector. Next, a Fuzzy-based Multidimensional Resource Scheduling is designed using the trapezoidal fuzzification and fuzzy square inference aiming at improving the resource scheduling efficiency. Finally, Multidimensional Queuing Network is designed for the scheduled resources in a queue structure for each classes reducing the response time.

2.1. Problem statement

Load balancing failed to reach the required level in cloud using the existing STM method. On the other hand, using SWDP, average success rate was not achieved easily. Both the exiting methods consume more amount of time for resource scheduling. The issues that remain unaddressed were scheduling efficiency, response time and load balancing. In order to overcome such types of issues in existing methods, F-MRSQN method is proposed in cloud environment by achieving better performance.

2.2. System model

In the proposed method, the scheduling algorithm is in charge of allocating the multidimensional resources (i.e. CPU, Memory and Bandwidth) from different cloud service providers '$CSP = CSP_1, CSP_2, \ldots, CSP_n$' to the cloud users '$CU = CU_1, CU_2, \ldots, CU_n$' in the cloud environment. Let '$VM = VM_1, VM_2, \ldots, VM_m$' be the set of '$m$' virtual machines that process 'n' tasks (i.e. files) denoted by the set '$S = S_1, S_2, \ldots, S_n$' that are running in parallel.

Let us further assume that the dimension of the resources be 'd', then each provider's (i.e. cloud service providers) resources is expressed in the form of a vector '$\overrightarrow{c_a} = c_a^1, c_a^2, \ldots, c_a^d$', in which '$c_a^d$' corresponds to the '$dth$' dimensional resource that the cloud service provider 'a' has with it. Therefore the resource consumption by application 'b' when executed on a cloud service provider 'a' is expressed as follows.

$$\overrightarrow{R_{ab}} = \left(R_{ab}^1, R_{ab}^2, \ldots, R_{ab}^d\right) \tag{1}$$

Table 1
Linguistic variable and their specifications.

	Fuzzy linguistic variables ($In = A$)	Fuzzy linguistic states	Meaning
Inputs	Bandwidth (BW)	Low Medium High	'L' 'M' 'H'
	Memory (Mem)	Small Medium Large	'S' 'M' 'La'
	CPU	Low Medium High	'L' 'M' 'H'
Output	Data center resource scheduling (ρ = B)	Low Moderate High	'L' 'M' 'H'

From Eq. (1) '$\overrightarrow{R_{ab}}$', corresponds the resource being consumed by 'a' cloud service provider for an application 'b'. Therefore, the resource scheduling problem is formulated as given below.

$$Max \sum_{a=1}^{n}\sum_{b=1}^{m} R_{ab}$$

$$Subject\ to \sum_{a=1}^{n}\sum_{b=1}^{m} \overrightarrow{R_{ab}} < \overrightarrow{c_a} \tag{2}$$

From Eq. (2), 'R_{ab}', corresponds to single resource being consumed by a 'a' cloud service provider for an application 'b'. From (2), the F-MRSQN is seen as a multidimensional resource scheduling problem where different applications 'b' assigned by different cloud service providers 'a'. Let 'PT_{ij}' represents the processing time for assigning a resource (i) 'R' to virtual machine (j) 'VM' is being defined as follows.

$$PT_{ij} = \begin{cases} 1, & if\ R\ is\ assigned\ to\ VM \\ 0, & Otherwise \end{cases} \tag{3}$$

Therefore the linear programming model with minimum processing time for assigning a resource 'R' to virtual machine 'VM' is expressed as given below.

$$Min = \sum_{i=1}^{n}\sum_{j=1}^{m} PT_{ij}P_{ij}$$

$$Subject\ to\ P_{ij},\ where\ i = 1, 2, \ldots, n\ and\ j = 1, 2, \ldots, m \tag{4}$$

2.3. Fuzzy-based multidimensional resource scheduling

With the objective of maximum resource utilization and minimum processing time, scheduling of resources according to user requests in a parallel fashion and load balancing of the scheduled resources is the problem to be solved in this paper. In order to solve this problem (i.e. maximum resource utilization and minimum processing time), this proposes a scheduling algorithm based on the hybrid Fuzzy MRS algorithm and Multidimensional Queuing Network in cloud infrastructure. The following part of this section presents Fuzzy-based Multidimensional Resource Scheduling (MRS) algorithm, which contains the fuzzification and defuzzification model, fuzzy inference and fuzzy rule.

To establish the fuzzy model, the F-MRSQN method initially obtains the input and output variables. There are three linguistic variables 'In', Bandwidth, Memory and CPU and one output variable 'ρ' which represents the efficient resource scheduling. The linguistic variables for the F-MRSQN method are denoted in Table 1 and are expressed as below.

$$In \rightarrow \{BW, Mem, CPU\};$$

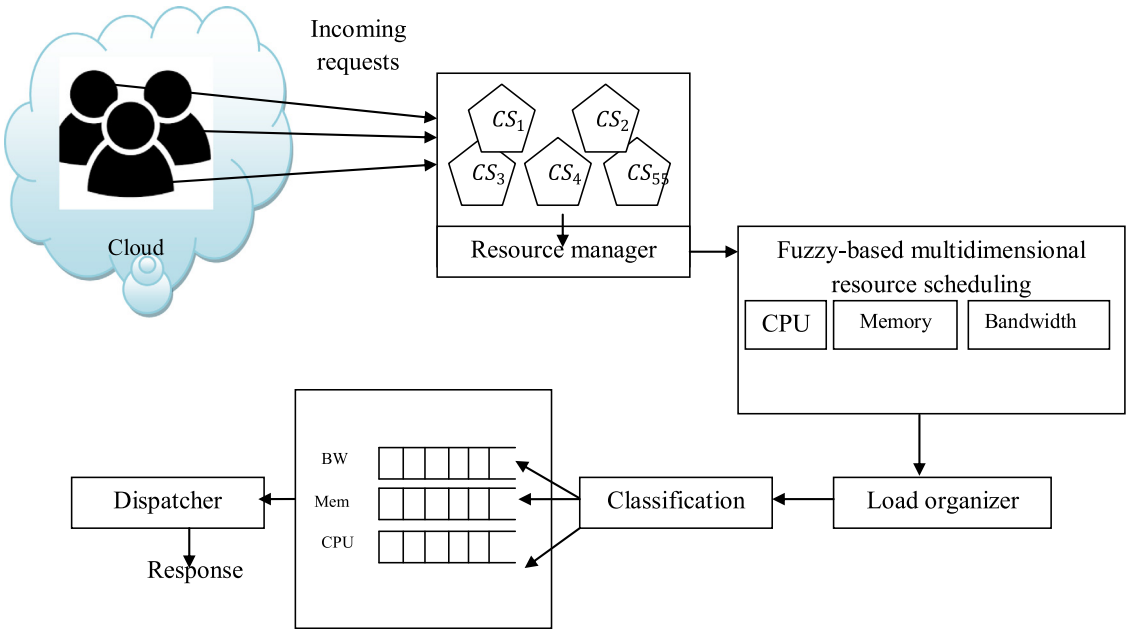


Fig. 1. Block diagram of the F-MRSQN.

$$BW \rightarrow \{L, M, H\};\ Mem \rightarrow \{S, M, La\};\ CPU \rightarrow \{L, M, H\} \quad (5)$$

Once the input linguistic variables and output variables are determined, the next step in the F-MRSQN method is to apply a trapezoidal fuzzification function for each input linguistic variables to present an associate observation. With the help of fuzzification, the real scalar value is changed into a fuzzy value. The trapezoidal fuzzification function used in F-MRSQN method is expressed as given below.

$$f_d: \{BW, Mem, CPU\} \rightarrow R \quad (6)$$

From Eq. (6), '*R*' denotes the fuzzy sets with the trapezoidal fuzzification function 'f_d' for the multidimensional resources '*BW*, *Mem*, *CPU*' to be scheduled in cloud environment. With the resultant trapezoidal fuzzification, the F-MRSQN method measures the fuzzy inferences using fuzzy square (Fig. 2).

In the proposed method, with the input being '*In*' and output being 'ρ', the fuzzy inference using fuzzy square is expressed as given below.

$$If\ In = A\ then\ \rho = B \quad (7)$$

From Eq. (7), '*A*', '*B*' represents the fuzzy numbers selected from the set of numbers and their linguistic states obtained from Table 1. From Eq. (7), '*A*' represents the linguistic variables as '*In*' and '*B*' represents the data center resource scheduling 'ρ'.

Finally defuzzification is performed in the F-MRSQN method that finds a single number compatible with membership function that forms the output in defuzzification process. The F-MRSQN method uses a centroid method to transform the output values of fuzzy inferences expressed as fuzzy set and is as given below.

$$\text{Fuzzy set}\,(y) = \frac{\mu_A\,(y)\,ydx}{\mu_A\,(y)\,dx} \quad (8)$$

From Eq. (8), let '$\mu_A\,(y)$' be the corresponding aggregated membership function, and let 'Y_L' be the low '*y*' value, attain minimum data center resource scheduling efficiency 'ρ', or 'Y_M' be the medium '*y*' value, attain medium data center resource scheduling efficiency 'ρ' or 'Y_H' be the high '*y*' value, attain high data center resource scheduling efficiency 'ρ'. Algorithm 1 given below provides the Fuzzy-based Multidimensional Resource Scheduling algorithm.

As given in the Algorithm 1, the Fuzzy-based MRS algorithm obtains the input parameters. The parameters needed to be settled include the number of cloud users to be assigned with the required resources, the available cloud servers and the input linguistic variables bandwidth, memory and CPU respectively. In step 2 for each cloud user their requirement for resources is analyzed by the resource manager with the objective of efficient resources to be scheduled in the cloud environment. In step 3, different linguistic input variables are obtained from the cloud user, followed by which fuzzification is performed. Next, fuzzy square inference is obtained through which the defuzzification is processed to measure the resource scheduling efficiency and therefore the data center utilization.

(Requirement) Cloud user = 5 (2 – *Mem* (*S*), 2 – *CPU* (*L*), 1 – *BW* (*L*))

(Available) Cloud user = 5 (1 – *Mem* (*S*), 2 – *CPU* (*L*), 0 – *BW* (*L*))

(Assigned) Cloud user = 3 (1 – *Mem*, 2 – *CPU*)

From the above example, the requirement of 5 cloud users are 2 users with small memory, 2 users with low CPU and 1 user with low bandwidth. The available resources in the cloud environment with 5 users being assigned with resources, 1 user assigned with requested memory and 2 users assigned with requested CPU.

The following steps are performed to conduct an experiment using Cloudsim Simulator. The first step is to set up the number of cloud users '*CU*' for the current simulation. This cloud user '*CU*' count is directly proportional to a number of cloud service providers in the current simulation. The next step involved is the initialization of the simulation, provided with current time, number of cloud users, and establishing linguistic variables for different resources. The third step involved is the creation of data center and the fourth step creates a data center broker by obtaining fuzzification function for scheduling the multidimensional resources *BW*, *Mem*, *CPU* in cloud environment. Whereas fifth step then creates a virtual machine(s) obtain fuzzy square inference by the input interference 'In' and output 'ρ'. Then the sixth step is the submission of virtual machine to data center broker. The

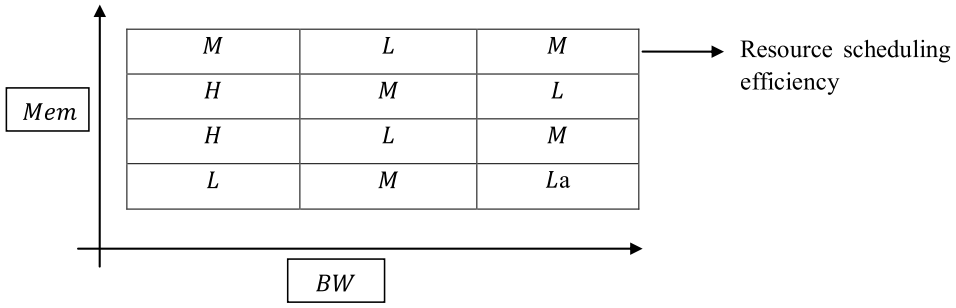


Fig. 2. Fuzzy square-based fuzzy inferences.

Input: Cloud User '$CU = CU_1, CU_2, \ldots, CU_n$', Cloud Server '$CS = CS_1, CS_2, \ldots, CS_n$', Input '$In = BW, Mem, CPU$', output Resource Scheduling 'ρ', f_d = trapezoidal fuzzification function 'A = linguistic variables', 'B = data center resource scheduling'
Output: Efficient resource scheduling
1: Begin
2: For 'CU' and resources to be assigned
3: Establish linguistic variables for different resources using Equation (5)
4: Obtain 'f_d' for scheduling the multidimensional resources BW, Mem, CPU in cloud environment using Equation (6)
5: $If\ In = A\ then\ \rho = B$
6: Perform centroid defuzzification using Equation (8) for efficient resource Scheduling
7: Else
8: Resource scheduling is unable to performed
9: End If
10: End for
11: End

Algorithm 1 Fuzzy-based Multidimensional Resource Scheduling algorithm

seventh step involves the submission of cloudlets to data center broker that performs centroid defuzzification for efficient resource scheduling continued with the eighth step to start the simulation. Upon completion of the process, the ninth step, send call to Stop Simulation. Finally, the tenth step prints the final status of the Simulation.

2.4. Multidimensional queuing network

Based on observations from previous sections, for each user requests, a multidimensional queuing network model for load balancing in cloud infrastructure is presented for the scheduled resources. The multidimensional queuing network (MQN) model allows considering different classes of requests of resources (i.e. Bandwidth, Memory and CPU) from several cloud users aiming at optimizing the load in cloud infrastructure.

The MQN model of a cloud server consists of '*R*' resources and '*C*' different classes of cloud user requests. In a MQN model, let '*CSS*' be an aggregate cloud server state which describes the cloud user requests from each class for each resource. Then the cloud server state, '$CSS = (z_1, z_2, \ldots, z_n)$' where each '$z_R$ ($R = 3$ *in our proposed method*)' is represented by a vector '$z_R = (n_{R1}, n_{R2}, \ldots, n_{Rm})$', with '$n_{Rm}$' being the total number of cloud users requests in class '*m*' at resource '*R*'. Therefore the resource utilization rate '$Util_R(t)$' is the product of total demands of one class of user requests at time interval '$\lambda_{Rj}(t)$'and the demand 'Dem_{Rs}' that is expressed as below.

$$Util_R(t) = Dem_{Rm} * \lambda_{Rm}(t) \tag{9}$$

The maximum utilization of each resource must be less than Resource Threshold Factor '*RTF*', so we have (as given in Eq. (10))

$$Max(Dem_{Rm} * \lambda_{Rm}(t)) \leq RTF \tag{10}$$

From Eq. (10), the condition states that, no resource will receive more user requests than it can handle. With this, the average latency of a cloud server for every class of user request is computed

as follows:

$$LT_R = LT_{Rm}(t) = \sum_{j=1}^{n} \frac{D_{Rj}}{1 - Util_R(t)} \quad (11)$$

From Eq. (11), 'LT_R', represent the average latency time of class 'm', with resources being 'R'. Also the average latency time for all classes of user requests in cloud infrastructure is evaluated as follows:

$$LT = \frac{LT_R(t) * \lambda_R(t)}{\lambda_R(t)} \quad (12)$$

As it can be seen in above mentioned analysis, for a cloud server model one queue for CPU resource, one queue for memory and one queue for bandwidth, the resource utilization '$Util_R(t)$' and average latency time for each class of cloud user requests, 'LT_R', is determined based on the number of cloud user requests at time '$\lambda_R(t)$'. In this way, load is said to be optimized for the resources being scheduled. Algorithm 2 given below provides the Multidimensional Queuing Load Optimization algorithm.

As shown in the above Algorithm 2, for each cloud user, the MQLO algorithm measures the resource utilization rate, if it satisfies the resource requirement of the cloud users. Followed by this, the average processing time of a cloud server and all cloud servers in the data center is evaluated. Accordingly, the allocation of resources is being performed ensuring load optimization.

The above said algorithm is then simulated using Cloudsim simulator and is as given below. The first step is to set up the number of cloud users 'CU' for current simulation, this cloud user 'CU' count is directly proportional to number of cloud service providers in the current simulation. Next step involved in Multidimensional Queuing Load Optimization algorithm is initialization of the simulation, provided with current time, number of cloud users, and establishing linguistic variables for different resources. It helps for the building of data center and it creates a Data center broker for measuring resource utilization rate in cloud environment in the second step. After that, the algorithm develops a Virtual Machine(s) for the submission to Data center broker. The fourth step involves submission of Cloudlets to Data center broker by checking a condition If(Max [(Dem)]_Rm* λ_Rm (t) ≤RTF), measuring average processing time of a cloud server and measuring average processing time of all cloud servers. The fifth step is to starts the simulation for completion of the process, the ninth step send call to the virtual machine(s) to stop simulation. Finally, the algorithm prints final status of the simulation.

3. Experimental settings

Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) is used in cloud data centers is implemented using JAVA language to perform simulation work. Cloudsim simulator uses Amazon Dataset to simulate different parameters. Number of user requests considered for experimentation ranges from 5 to 35.

The Cloudsim goal is to provide a global and extensible simulation framework that compares the proposed Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) in cloud data centers with the existing Virtual Local Area Network (VLAN) [2] towards scalable traffic management in cloud and the Scalable Workload Driver Partitioning (SWDP) [2] in cloud environment. Simulation is conducted to measure and evaluate performance of F-MRSQN method with factors such as average success rate, resource scheduling efficiency and response time for varying user requests in cloud environment.

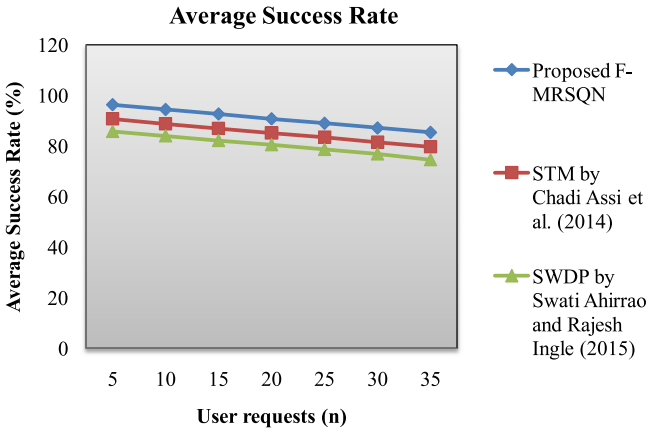


Fig. 3. Graph for User requests (n) Vs Average Success Rate (%).

Proposed Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) in cloud data centers is compared with existing Virtual Local Area Network (VLAN) towards Scalable Traffic Management (STM) developed by Assi et al. [2] and Scalable Workload Driven Partitioning (SWDP) developed by Ahirrao and Ingle [3]. Performance factors are analyzed with the help of graphs.

4. Results and discussion

To demonstrate the effectives of Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN), experiments are conducted and results analyzed with discussions provided on three metrics, average success rate, resource scheduling efficiency and response time.

4.1. Impact of Average Success Rate

Average Success Rate (ASR) is the ratio of number of users' requests addressed by virtual machine grid through the resource manager at a particular time in cloud environment. Average success rate is mathematically formulated as follows.

$$ASR = \frac{No.\ of\ user\ requests\ addressed\ by\ virtual\ machine\ grid}{Total\ no.\ of\ user\ requests} * 100 \quad (13)$$

From Eq. (13), average success rate '*ASR*' is measured with respect to number of cloud user requests. It is measured in terms of percentage (%). If average success rate is high, then the method ensures optimal resource scheduling.

Fig. 3 shows average success rate in terms of number of user requests in cloud infrastructure for proposed F-MRSQN method and existing methods such as STM by Assi et al. [2] and SWDP by Ahirrao and Ingle [3]. The average success rate is measured while sharing files on cloud infrastructure. Number of user requests is varied in the range from 5 to 35 for simulation purpose. It is clear that average success rate is reduced for all methods with the increase in number of user requests. But proposed F-MRSQN method achieves better performance on average success rate when compared to existing methods.

In order to improve the average success rate, F-MRSQN method uses Fuzzy-based Multidimensional Resource Scheduling that allocates resources according to trapezoidal fuzzification function. The Trapezoidal fuzzification function performs efficient multidimensional resource allocation. Additionally F-MRSQN method derives fuzzy square-based inferences that are related to human

Input: Resource Threshold Factor 'RTF', Cloud User '$CU = CU_1, CU_2, \ldots, CU_n$', Cloud Server '$CS = CS_1, CS_2, \ldots, CS_n$', Resource '$R$', Time '$t$',
Output: Optimized Resources
Begin 1: Initialize or Start 2: For 'CU' with balanced load and resources to be assigned 3: Measure $Util_R(t)$ using Equation (9) 4: If $Max\ ((Dem_{Rm} * \lambda_{Rm}(t)) \leq RTF)$ 5: Measure average processing time of a cloud server using Equation (11) 6: Measure average processing time of all cloud servers using Equation (12) 7: Else 8: Multiple users access the resources 9: End if 10: End for 11: End

Algorithm 2 Multidimensional Queuing Load Optimization algorithm

thinking by using linguistic terms. As a result, optimum outputs are obtained close to the target output (i.e. average success ratio). Hence F-MRSQN method increases average success rate by 6% when compared to existing STM by Assi et al. [2] and 12% when compared to SWDP by Ahirrao and Ingle [3] respectively.

4.2. Impact of resource scheduling efficiency

Resource scheduling efficiency is defined as ratio of the number of resources is scheduled based on user request to the total number of resources in cloud. Resource scheduling efficiency is mathematically formulated as follows.

$$\textit{Resource Scheduling Efficiency} = \frac{\textit{No. of resources are scheduled}}{\textit{Total no. of resources}} * 100 \quad (14)$$

Resource scheduling efficiency measured in proposed F-MRSQN method ensures better performance for cloud users by using Max Min Fuzzy inference algorithm. The resource scheduling efficiency is measured in terms of percentage (%). If resource scheduling efficiency is high, then the method is said to be more efficient.

To explore the influence of resource scheduling efficiency on F-MRSQN method with the help of Fuzzy-based Multidimensional Resource Scheduling algorithm, the simulations were performed by varying the cloud users in Fig. 4. It also shows that with the application of Fuzzy-based MRS algorithm it extensively provides competitive results compared to the state-of-the-art methods, namely STM by Assi et al. [2] and SWDP by Ahirrao and Ingle [3]. The resource scheduling efficiency using F-MRSQN method increases with the increase in the number of cloud users and file densities comparatively performs better than the state-of-the-art methods. This is because the aggregated membership function using F-MRSQN method schedules all user resources based on the cloud user request. Also the data center with high resource scheduling efficiency is considered with the help of centroid

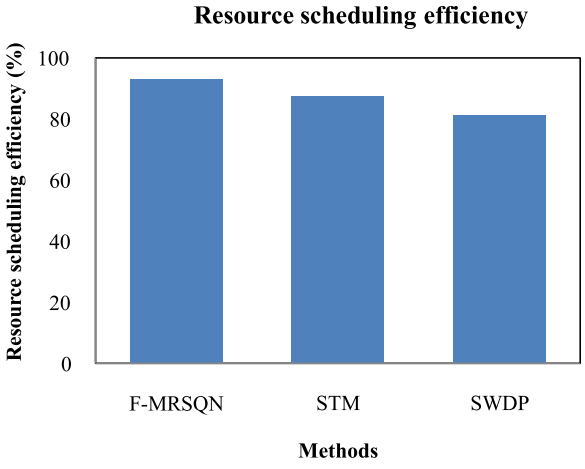


Fig. 4. Comparison of resource scheduling efficiency Vs F-MRSQN, STM and SWDP.

method. This in turn improves the resource scheduling efficiency by 6% compared to STM by Assi et al. [2] and 7% compared to SWDP by Ahirrao and Ingle [3] respectively.

4.3. Impact of response time

Response time is defined as time utilized to respond by a MQLO algorithm in a distributed manner while scheduling the resources in cloud. The mathematical formulation of response time is given below.

$$RT = n * time(respond\ for\ user\ request) \quad (15)$$

From Eq. (15), response time 'RT' is measured by time taken for responding the user request '$time(respond\ for\ user\ request)$' by the algorithm with respect to number of users' requests 'n' for availing

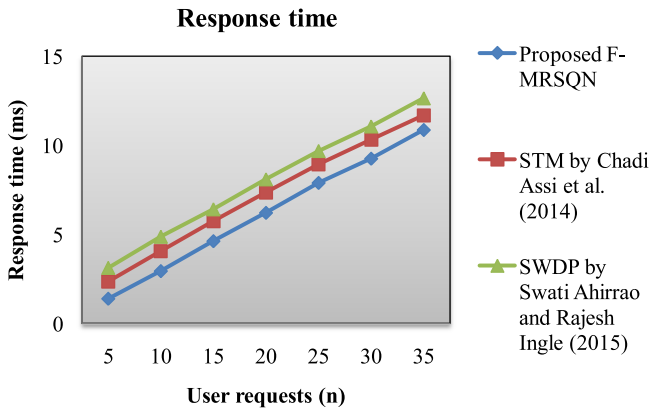


Fig. 5. Graph for User requests (n) Vs Response time (ms).

resources. If response time is low, then the method is said to be more efficient. It is measured in terms of milliseconds (ms).

Fig. 5 shows the performance of response time calculated using proposed F-MRSQN method and compared with existing methods namely STM by Assi et al. [2] and SWDP by Ahirrao and Ingle [3]. Number of user requests is varied in the range from 5 to 35 for simulation purpose. It is clear that response time gets increased for all methods with the increase in number of user requests. But proposed F-MRSQN method achieves better performance on response time when compared to existing methods.

From Fig. 5, it is clear that response time is reduced gradually for proposed F-MRSQN method when compared to other existing methods. This efficient reduction on response time is achieved using Multidimensional Queuing Load Optimization algorithm. Both average processing time of a cloud server and average processing time for all cloud servers are calculated simultaneously to satisfy load balancing based on file density. In addition, resource utilization is also checked based on resource threshold factor and provides optimum result by preventing resource requirement greater than threshold factor. Hence response time is reduced in proposed F-MRSQN method by 26% when compared to existing STM by Assi et al. [2] and 45% when compared to SWDP by Ahirrao and Ingle [3] respectively.

5. Conclusion

In this paper, Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) method is proposed for efficient scheduling of resources and optimizing the load for each cloud user requests with the efficient evolution of data center. The main objective of this proposed method is the utilization of effective integrated scheduling and load balancing algorithm subject to maximum resource utilization and minimum processing time in cloud environment. For coarser construction on user requests, resource scheduling efficiency is improved by performing the Fuzzy-based Multidimensional Resource Scheduling in proposed F-MRSQN method. Then, with the efficient resource being scheduled between the cloud users, comparable services with minimum response time based on the resources being utilized. Followed by this with the application of multidimensional queuing network, efficient balancing of loads on scheduled resources is said to be achieved, therefore enhancing the average success rate for each cloud user requests. The effectiveness of F-MRSQN method is estimated by attaining simulation results for testing the average success rate and resource scheduling efficiency and response time. The simulation is carried out for different parameters such as average success rate, resource scheduling efficiency and response time. The results show that F-MRSQN method provide better performance with an improvement of average success ratio by 9% and reduce the response time by 20% compared to existing methods. With several information and computation intensive applications on cloud environment, intermediate sharing of data and information is becoming an important research area. Privacy preserving for intermediate sharing of data and information is one of salient yet demanding research issues, and needs in-depth consideration. With the contributions of this paper, we are planning to further investigate privacy-aware efficient resource scheduling with load balancing of intermediate data and information in cloud by taking privacy preserving as a metric together with other metrics such as average success ratio and response time.

References

[1] Amandeep Kaur, Mr. Pawan Luthra, A review on load balancing in cloud environment, Int. J. Comput. Technol. 17 (1) (2018) 7120–7125.
[2] Chadi Assi, Sara Ayoubi, Samir Sebbah, Khaled Shaban, Towards scalable traffic management in cloud data centers, IEEE Trans. Commun. 62 (3) (2014) 1033–1045.
[3] Swati Ahirrao, Rajesh Ingle, Scalable transactions in cloud data stores, J. Cloud Comput. 4 (1) (2015).
[4] L.D. Dhinesh Babu, P. Venkata Krishna, Honey bee behavior inspired load balancing of tasks in cloud computing environments, Appl. Soft Comput. 13 (2013) 2292–2303.
[5] Junwei Cao, Keqin Li, Ivan Stojmenovic, Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers, IEEE Trans. Comput. 63 (1) (2014).
[6] Bruno Guazzelli Batista, Julio Cezar Estrella, Carlos Henrique Gomes Ferreira, Dionisio Machado Leite Filho, Luis Hideo Vasconcelos Nakamura, Stephan Reiff-Marganiec, Marcos José Santana, Regina Helena Carlucci Santana, Performance evaluation of resource management in cloud computing environments, PLOS ONE 10 (2015) http://dx.doi.org/10.1371/journal.pone.0141914.
[7] Mohammad Nour Hindia, Ahmed Wasif Reza, Kamarul Ariffin Noordin, Muhammad Hasibur Rashid Chayon, A novel LTE scheduling algorithm for green technology in smart grid, PLOS ONE (2015) http://dx.doi.org/10.1371/journal.pone.0121901.
[8] Flavio Lombardi, Roberto Di Pietro, Secure virtualization for cloud computing, J. Netw. Comput. Appl. 34 (4) (2010).
[9] Msc. Nasrin Hesabian, Hamid Haj Seyyed Javadi, Optimal Scheduling In Cloud Computing Environment Using the Bee Algorithm, Int. J. Comput. Netw. Commun. Secur. 3 (6) (2015) 253–258.
[10] Abdulhussein Abdulmohson, Sudha Pelluri, Ramachandram Sirandas, Energy efficient load balancing of virtual machines in cloud environments, Int. J. Cloud-Comput. Super-Comput. 2 (1) (2015) 21–34.
[11] Sarpreet Singh, R.K. Bawa, Optimized assignment of independent task for improving resources performance in computational grid, Int. J. Grid Comput. Appl. 6 (1/2) (2015).
[12] Krzysztof Rzadca, Anwitaman Datta, Gunnar Kreitz, Sonja Buchegger, Game-Theoretic mechanisms to increase data availability in decentralized storage systems, ACM Trans. Auton. Adapt. Syst. 10 (3) (2015) 1–32.
[13] Georgios L. Stavrinides, Helen D. Karatza, Scheduling multiple task graphs with end-to-end deadlines in distributed real-time systems utilizing imprecise computations, J. Syst. Softw. 83 (6) (2010) 1004–1014.
[14] Aarti Singh, Dimple Junejab, Manisha Malhotraa, Autonomous agent based load balancing algorithm in cloud computing, in: International Conference on Advanced Computing Technologies and Applications, ICACTA-2015, 45, Elsevier, 2015, pp. 832–841.
[15] Yang Wang, Wei Shi, Budget-Driven scheduling algorithms for batches of mapreduce jobs in heterogeneous clouds, IEEE Trans. Cloud Comput. 2 (3) (2014) 306–319.
[16] Marco Polverini, Antonio Cianfrani, Shaolei Ren, Athanasios V. Vasilakos, Thermal-Aware scheduling of batch jobs in geographically distributed data centers, IEEE Trans. Cloud Comput. 2 (1) (2014) 71–84.
[17] Qi Zhang, Mohamed Faten Zhani, Raouf Boutaba, Joseph L. Hellerstein, Dynamic heterogeneity-aware resource provisioning in the cloud, IEEE Trans. Cloud Comput. 2 (1) (2014) 14–28.
[18] Chun-Wei Tsai, Wei-Cheng Huang, Meng-Hsiu Chiang, Ming-Chao Chiang, Chu-Sing Yang, A hyper-heuristic scheduling algorithm for cloud, IEEE Trans. Cloud Comput. 2 (2) (2014) 236–250.
[19] Chun-Cheng Lin, Hui-Hsin Chin, Der-Jiunn Deng, Dynamic multiservice load balancing in cloud-based multimedia system, IEEE Syst. J. 8 (1) (2014) 225–234.
[20] Joao Ferreira, Gustavo Callou, Paulo Maciel, A power load distribution algorithm to optimize data center electrical flow, Energies (Basel) 6 (2013) 3422–3443.

[21] E. Pinto Neto, Gustavo Callou, Fernando Aires, An algorithm to optimise the load distribution of fog environments, in: IEEE Int. Conference on Systems, Man, and Cybernetics, SMC, 2017, pp. 1292–1297.
[22] Smitha Sundareswaran, Anna Squicciarini, Dan Lin, Ensuring distributed accountability for data sharing in the cloud, IEEE Trans. Dependable Secure Comput. 9 (4) (2012) 556–568.
[23] Olivier Beaumont, Lionel Eyraud-Dubois, Hejer Rejeb, Heterogeneous resource allocation under degree constraints, IEEE Trans. Parallel Distrib. Syst. 24 (5) (2013) 926–937.
[24] D. Chitra Devi, V. Rhymend Uthariaraj, Load balancing in cloud computing environment using improved weighted round robin algorithm for nonpreemptive dependent tasks, Sci. World J. 2016 (2016) 1–14.
[25] Zhongma Zhu, Rui Jiang, A secure anti-collusion data sharing scheme for dynamic groups in the cloud, IEEE Trans. Parallel Distrib. Syst. 27 (1) (2016) 40–50.
[26] Zhenhua Wang, Haopeng Chen, Ying Fu, Delin Liu, Yunmeng Ban, Workload balancing and adaptive resource management for the swift storage system on cloud, 51, 2015, pp. 120–131,
[27] Yan Wang, Jinkuan Wang, Cuirong Wang, Xin Song, Research on resource scheduling of cloud based on improved particle swarm optimization algorithm, in: Advances in Brain Inspired Cognitive Systems, vol. 7888, Springer, 2013, pp. 118–125.