International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Design of A Multi-Constraint PSO for Resource Allocation and Task Scheduling

[1]Rajkumar Kalimuthu, [2]Brindha Thomas

Submitted: 01/10/2023 **Revised**: 21/11/2023 **Accepted**: 02/12/2023

Abstract-Cloud computing (CC) is a modern technology where resource allocation and task scheduling are considered as an essential factor. Based on the literature, particle swarm optimization (PSO) is a stochastic optimization approach inspired by the foraging nature of bird flocks. PSO is extensively used in various fields like scheduling cloud resources, scheduling problems, etc. The efficiency of the model intends to address the issues encountered in existing approaches. Here, time-slot-based rule generation (TS-RG) is designed to handle workflow scheduling in the cloud. A particle scrambling process is provided to map the VM for every task and perform scheduling sequentially. An idle time slot-aware re-scrambling process is anticipated to re-scramble the particles to various scheduling solutions. Due to PSO randomness, the cloud encounters invalid task priorities; however, this issue is handled effectually by the repair method used for handling the invalid task priorities and makes them valid. The anticipated model is compared with various prevailing approaches, and the experimental outcomes demonstrate that the anticipated model outperforms other works in deadline fulfilment and execution cost computation.

Keywords- *cloud computing, task scheduling, resource allocation, rule generation, randomness*

1. Introduction

In distributed computing, there is an accelerating technology called cloud computing. Many applications cloud computing uses, like data analytics, data storage, and the Internet of Things [1]. An individual or enterprise deploys the services which are changed with cloud computing through conventional methods. Various types of services are provided by cloud computing to the users who are registered as web services. It makes the users does not invest in the infrastructure of CC. SaaS, PaaS, and IaaS are the services in CC [2]. The users needed to submit the request for every type of service through the internet to the service provider. The service provider does the resource management to satisfy the user's generated tasks.

Service providers adopt scheduling algorithms to deal with the incoming tasks or requests and accomplish the computing resources effectively. The maximization of revenue and resource utilization to certain limits is done with the help of providers in task scheduling and resource management. Practically, resource allocation and scheduling are the main difficulties concerning the performance of resources in cloud computing [3]. The authors give attention to the task scheduling research for this reason. The procedure to arrange the incoming tasks or requests in a particular way to utilize the available resources properly is called task scheduling [4]. Service users need to submit the request online due to the technology of cloud computing providing the services via the internet as the medium. The number of requests or tasks is created due to the number of users in every service. Due to the waiting period, few short-term requests can be terminated [5].

Moreover, systems have not used the scheduling that features the tasks' waiting periods. The scheduler considers the number of constraints at the time of scheduling has the task's nature, tasks size, the execution time of a task, resource availability, queue of the task, and the resources load. An important problem in CC is task scheduling. It is an effective resource utilization process. It has resulted in proper task scheduling. The proper utilization of resources is promoted as an important benefit of cloud computing [6]. Then, allocating resources and scheduling the task are the two sides of this cloud computing. Both affect each other.

The novel VM selection mechanism for each task contributes significantly to the efficiency of the scheduling process by tailoring the allocation of virtual machines (VMs) to tasks based on their specific resource requirements and constraints. This mechanism considers factors such as CPU and memory demands, as well as communication patterns between tasks. By allocating VMs more precisely, the scheduling process optimizes resource utilization, reduces contention for shared resources, and minimizes the potential for resource bottlenecks. Consequently, the overall system efficiency is enhanced as tasks are allocated to VMs that best suit

[1]Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, India.
[2]Department of Information Technology, Noorul Islam Centre for Higher Education, India.
Corresponding email : rajkumarengg2020@gmail.com

their requirements, leading to improved task execution times and better utilization of available resources.

The proposed MC-PSO model demonstrates its ability to outperform various existing approaches in terms of reducing overall execution time. Through the integration of multi-constraint optimization principles and PSO, the model efficiently navigates the solution space to identify near-optimal task scheduling and resource allocation solutions. This results in a reduced makespan and improved task completion times. Numerical outcomes showcase specific performance improvements, such as a notable percentage reduction in execution time compared to alternative methods. The MC-PSO's capacity to simultaneously consider multiple constraints and adapt to dynamic scenarios contributes to its superior performance in optimizing complex scheduling problems.

Moreover, Internet users access the gratified anytime and anywhere devoid of considering the host's infrastructure. These hosting infrastructures comprised different machines with different abilities, and the service provider can manage and handle these capabilities [7]. These capabilities are enhanced for this infrastructure to access the internet in cloud computing. Services provide to make cloud service providers earn profits for cloud service users. The end-user of cloud service can be utilized the complete computing services stack that is ranged from applications to hardware. The services use the pay as you go basis in cloud computing. The available resources are reduced or increased by the end-user of cloud service concerning the application demands [8]. It is considered an important benefit of cloud computing, yet the users are accountable to pay extra costs for the services for the benefits. The resources can be rented at any time by the cloud service user and released without challenges. The user of cloud services has the liberty to use any services depending on the need of an application [9]. The problem occurs with this users' freedom to choose the service that is the upcoming user request that is not predicted properly. Then, the allocation of resources and task scheduling are the important sections of research on cloud computing [10]. The uses of resource efficiency are based on the methodologies of load balancing and scheduling than the allocation of resources randomly. Solving complicated requests or tasks are used in cloud computing. The scheduling algorithms used are recommended in resolving the problems of complicated tasks. The resources are leveraged using these scheduling algorithms [11] – [15]. The following are the major research contributions:

1) This work focused on modelling and effectual hybrid optimization approaches to eliminate various issues faced by the standard optimization approaches.

2) The cloud-based workflow scheduling approach merges the idle time slot-based rules and multi-constraint PSO to reduce the execution cost during workflow application over various deadline constraints.

3) Here, a particle scrambling process is presented to specify the VM types essential for every task and schedule the task sequences. An idle time slot-aware re-scrambling process is anticipated to re-scramble the particles to various scheduling solutions.

4) The anticipated model is compared with various prevailing approaches, and the experimental outcomes demonstrate that the anticipated model outperforms various existing approaches in deadline fulfilment and execution cost computation.

The objective of the research titled "Design of a Multi-Constraint PSO for Resource Allocation and Task Scheduling" is to create an innovative optimization algorithm that combines Particle Swarm Optimization (PSO) to effectively address the complexities of resource allocation and task scheduling in intricate systems. This study aims to achieve several specific goals. Firstly, it aims to develop a PSO-based framework capable of handling multiple constraints simultaneously, encompassing factors such as resource availability, task dependencies, and deadlines. Secondly, the research intends to devise resource allocation strategies that intelligently distribute resources like processing units, memory, and network bandwidth among tasks, optimizing system efficiency while adhering to resource limitations. Additionally, the study aims to design novel task scheduling methods that consider task characteristics, such as execution times, precedence relationships, and priority levels, in order to determine optimal task sequences.

To evaluate the proposed algorithm's effectiveness, appropriate performance metrics will be established to assess solution quality, convergence speed, and robustness across various scenarios. The research also involves comparing the multi-constraint PSO algorithm with other leading optimization techniques for resource allocation and task scheduling, highlighting the algorithm's strengths and weaknesses. Through applications in cloud computing, manufacturing, and project management, the study seeks to demonstrate the algorithm's practicality. Sensitivity analysis will uncover the algorithm's performance under diverse conditions, while scalability evaluations will provide insights into its effectiveness for intricate systems. Exploring adaptive mechanisms, parameter tuning, and hybridization with

other techniques will further enhance the algorithm's efficiency. Ultimately, this research aims to contribute new knowledge to optimization, resource allocation, and task scheduling domains by presenting a unique approach and insights into its integration with multi-constraint challenges.

The work is drafted as follows: section 2 offers a comprehensive analysis of various prevailing approaches and elaborates on the merits and demerits of the anticipated model; section 3 provides a detailed analysis of the anticipated model. The numerical outcomes and the graphical representations are given in section 4, followed by a research summary.

2. Related works

Shen et al. [16] propose a heuristic-based Particle Swarm (PSO) method for scheduling cloud service workloads considering both computation and data transmission costs. By altering the cost of computing and connectivity, the study experimented with both the implementation of a procedure. When adopting PSO, the study compares cost reductions to the present "Best Resource Selection" (BRS) method. Taleb et al. [17] provide a genetic algorithm-based scheduling technique for VM asset load adjustment. This methodology recognizes the effect after trying to send the necessary VM assets and then chooses the least-full of sense of agreement. It achieves the highest-burden adapting and lowers or tries to avoid energetic relocation based on accurate information and the present incarnation of the framework and through Simulated annealing. In the aftermath of scheduling, this method addresses the issue of weight strangeness and high movement costs caused by traditional approaches. In cloud services, Wang et al. [18] investigate two-dimensional job scheduling instruments based on load adjustment and its process in the context of the CloudSim toolkit. This assignment scheduling software may fulfil the clients' requirements while also utilizing good assets.

Marotta et al. [19] investigate the possibility of distributing Virtual Machines (VMs) flexibly to maximize the use of physical assets. The automated sequencing technique for the methodology is based on an Improved Genetic Algorithm (IGA). The IGA uses the most restricted attributes and provides a Dividend Payout ratio in Economics to pick between an optimum and an ideal categorization for the VMs' wants. The recreation tests demonstrate that this dynamic routing solution outperforms the Eucalyptus, Open Nebula, Nimbus IaaS cloud, and similar clouds. The studies show that the IGA's speed under Grid conditions triples that of the traditional GA scheduling problem. The utilization rate of assets is consistently greater than that of expansive IaaS cloud infrastructures. Kawashima et al. [21] suggested a Simplified Discrete Particle Swarm Optimization to plan workloads among cloud advantages that consider both the cost of transmission of information and the expenditure of the method. A cloud values demonstration is used to lead the inquiry using many work process industries by varying their communication and information expenditures and algorithm changes. With RDPSO, the conventional PSO, and the BRS (Best Resource Selection) algorithms, an association is formed to make range and cost-cutting fractions and the cost fund managers [22] – [25]. The RDPSO algorithm may achieve much greater cost reserve funds and good execution.

Ghaznavi et al. [26] offer a capable cloud service system that recognizes, separates, and aggregates are remaining cloud jobs based on the loads it manages and their QoS needs. The other program was conducted using several scheduling methodologies and their accompanying algorithms. Using current scheduling algorithms, the CloudSim toolbox is used to monitor the implementation of these algorithms. Bhamare et al. [27] used an intermediate number premise to depict the sensitivity of the registration conditions and a scheduling design to mitigate the influence of sensitivity on scheduling effectiveness in a cloud server farm. In light of this architecture, the study proposes a unique scheduling algorithm for scheduling continuous, irregular, autonomy tasks that effectively employ active and passive programming tactics. The study provides three techniques to scale all over the platform's registration assets as suggested by the remaining task to increase asset use and reduce energy consumption for the cloud services to boost power utilization.

Bagua et al. [28] represent the work process timing dilemma as a Multi-objective Optimization Problem (MOP) for Cloud scenarios, which accelerates both completion time and cost. In the frameworks as a service (IaaS) stage, the investigation provides an Evolutionary Multi-Objective Optimization (EMO) technique to address this workflow management issue. This program proposes new strategies for issue graphic recording, crowd induction, wellness evaluation, and genetic modifications. Broad tests into both certifiable and arbitrarily produced work processes show that the calendar delivered by this transformational algorithm outperforms the occasion-based IaaS registration and evaluating algorithms on the preponderance of work processes. The results also show that, on the whole, this approach may achieve much better results than existing best-in-class QoS Optimization scheduling techniques. Zhang et al. [29] suggested the Genetic optimization technique riotous Ant Colony method to solve this

compelling number of co-optimization problems, in which four improvements and character estimation are linked. Using a Markov-based method, the creators conduct a reliability analysis of cloud benefits. They characterize the cloudy scheduling problem as a multi-objective optimization problem with trustworthiness, task scheduling, and flow time constraints [30].The modified best fir reduces SLA violation and energy consumption by managing resource allocation and provisioning. However, all these methods are single objective constraints. The proposed model works to deal with multi-objective constraints and intends to enhance the performance. Table 1 depicts the comparison of existing approaches.

Table 1 Comparison of Existing approaches

References	Goal	Allocation Environment	Workload	Resources
Sherbha et al., [31]	Task consolidation is adopted to diminish energy consumption and enhance resource utilization	Dynamic	Homogeneous	VM and IaaS
Xhang et al., [32]	Predicts users request to allocate heterogeneity based VMs	Dynamic	Heterogeneous	SaaS, IaaS and VM
Zhang et al., [32]	Threshold-based energy aware model for VM mapping	Dynamic	Heterogeneous	VMs and IaaS
Kholidy et al., [33]	Performs migration and VM mappings	Dynamic	Synthetic workload	VMs, CPU and IaaS
Srimoyee et al., [34]	Perform resource mapping to fulfill QoS	Dynamic	Heterogeneous	VMs
Rui et al., [15]	Workload allocated to VM	Dynamic	Heterogeneous	VMs and IaaS
Gu et al., [14]	Probabilistic resource allocation	Dynamic	Synthetic workload	IaaS

3. Methodology

The single objective constraint is handled by considering multi-objective constraints using three baseline factors: 1) cloud-based workflow scheduling approach merges the idle time slot-based rules and multi-constraint PSO to reduce the execution cost during workflow application various deadline constraints. Here, a particle scrambling process is presented to specify the VM types essential for every task and schedule the task sequences. An idle time slot-aware re-scrambling process is anticipated to re-scramble the particles to various scheduling solutions.

3.1. Workflow scheduling

The proposed model includes particles population, and every particle gives the probable solution to resolve these issues. Here, N specifies the population size with the particles over the population. The fitness function provides the finest solution search by evaluating the particle quality. Every particle is composed of velocity and position. The i^{th} particle velocity at t^{th} iteration is specified by $v_i^t = (v_{i1}^t, v_{i2}^t, \dots, v_{iD}^t)$ and position $x_i^t = (x_{i1}^t, x_{i2}^t, \dots, x_{iD}^t)$ where D specifies the search space dimension. The previous best position $p_i^t = (p_{i1}^t, \dots, p_{iD}^t)$ specifies the best prior position with superior fitness value for $the\ i^{th}$ particle at t^{th} iteration. The global best position $g^t = (g_1^t, \dots, g_D^t)$ gives the global best position attained with all particles. At every PSO iteration, the position and velocity of the d^{th} particle dimension is updated with the following Eq. (1) & (2);

$$v_{id}^{t+1} = \omega v_{id}^t + c_1 r_1 (p_{id}^t - x_{id}^t) + c_2 r_2 (g_d^t - x_{id}^t) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (2)$$

Here, ω inertia weight gives the balance among the local and global search; r_1 and r_2 give the uniform random number [0,1]; $c_1\ and\ c_2$ specify the acceleration factors to manage the social and cognitive components, and its preliminary value is set as 2. The algorithm of the baseline PSO for scheduling process is shown below. It iteratively evolves the particle population and solution to attain global best particle during final scheduling solution. There are diverse issues to be considered to

understand when the PSO is intended to resolve the scheduling process. The model includes some particle scrambling representation and de-scrambling processes to attain a scheduling solution. Next, the fitness functions specify the optimization objective, followed by population initialization and deadline constraints of scheduling issues. Some infeasible solutions are acquired during the hunt for PSO to eliminate the deadline constraints. The multi-constraint handling (MC-PSO) approach is proposed to deal with these infeasible solutions, an enhanced version of PSO. There are some cases to get some feasible solution: better fitness value with an optimal solution. The solution to the least execution time is superior when both solutions are infeasible. The target of considering workflow scheduling is to perform pipeline processing (scheduling) into a single task that reduces the runtime overhead and preserves the data transfer time among the tasks. The pipelining process specifies the special task t_i and t_j with a constant relationship. Here, one task is for the parent, and the other is for the child.

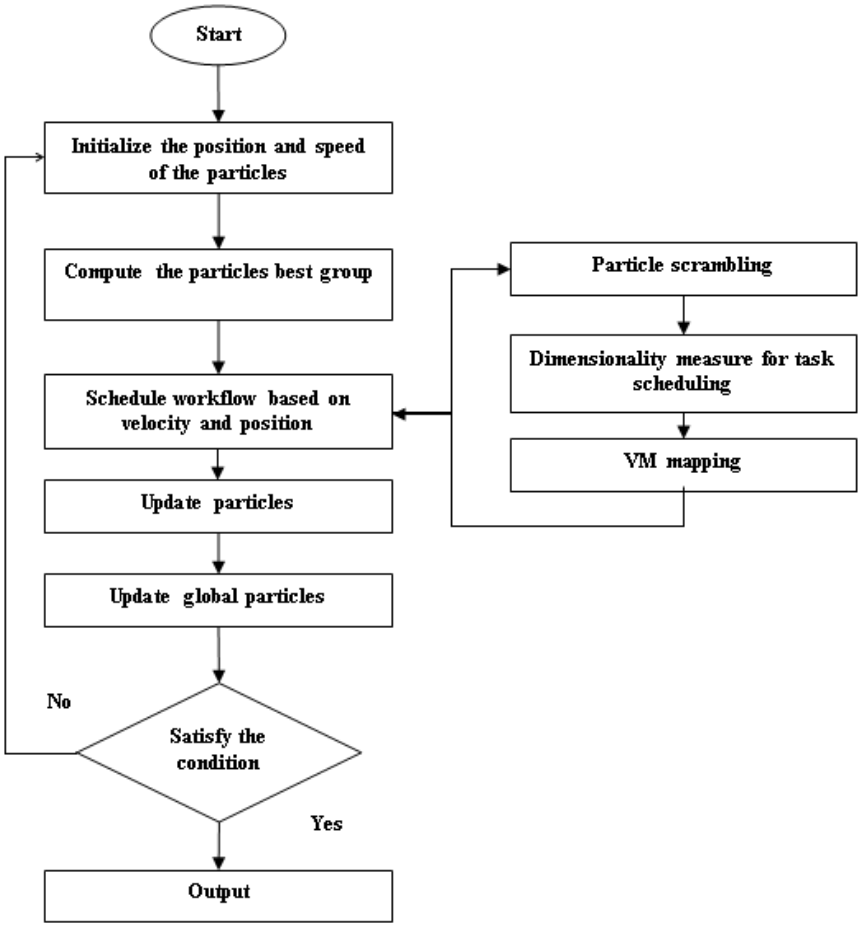

Fig.1 DFD of proposed model

Algorithm 1: MC-PSO

Input: Directed acyclic graph $G = < T, E >$

Output: Scheduling solution $S = < \theta, M, R >$

1. Initializing workflow and population;
2. Initialize parameters like inertial weight and N population size;
3. Set the number of iterations $t = 1$;
4. **While**$t \leq$ total amount of iterations do
5. **for** every particle $q_i, i \epsilon \{1,2,..,N\}$ do
6. Update velocity v_i^t;

7. Update position x_i^t;
8. Re-scramble the process as $q_i \rightarrow S_i$;
9. Compute Ω and Ψ; //based on S_i;
10. Update the prior best solution p_i of q_i;
11. **End for**
12. update global best particle g;
13. $t = t + 1$;
14. **end while**
15. output scheduling solution;

3.2. Particle scrambling

The primary concept with MC-PSO is to handle the particle scrambling process. However, VM instances are infinite where the service providers offer a finite amount of VM. However, task prioritization demonstrates task scheduling by scrambling the instances. The task scheduling sequence is acquired by soring the task priorities in lower to higher order (ascending). Thus, particles consist of two parts: 1) dimensionalities from $1 \rightarrow n$ exemplify the task mapping to VM types and 2) dimensionalities from $n+1 \rightarrow 2n$ specify task prioritization. The particle dimensionality is equal to several tasks (workflow). The dimensionality value ranges from the entire range of $[1, m]$. The value of the next part gives a real number (positive). The initial particles show dimensions from $1 \rightarrow n$ related to tasks $t_1 \rightarrow t_n$. The nearer rounded integer of every dimension specifies the VM index related to the task-related with the instance of VM type. In the successive particle, the nearer rounded dimensionality integer $i \in [n+1, 2n]$ specifies the task priority $t_{i-n.}$ The example for this section is provided in Fig 2, in which the number of VM kinds is considered as 4. Dimensionality 1 is related to task t_1, and the value is 2.1. The VM needed by t_1 is π_2. The dimensionality value of $2 \rightarrow 5$ follows the same logic. The dimensionality value of $6 \rightarrow 10$ is rounded as 2, 9, 11, 3, and 14, specifying the priorities of $t_1 \rightarrow t_5$. The sorted value (ascending order) and the sequence of task scheduling are provided as $t_1 \rightarrow t_4 \rightarrow t_2 \rightarrow t_3 \rightarrow t_5$.

	t_1	t_2	t_3	t_4	t_5	t_1	t_2	t_3	t_4	t_5
Dim	1	2	3	4	5	6	7	8	9	10
Value	2.2	2.8	1.7	1.2	1.9	2.4	8.9	11.3	3.5	13.9

$t_1 \rightarrow \pi_2$	$t_1 \rightarrow \pi_2$	$t_2 \rightarrow \pi_2$	$t_4 \rightarrow \pi_1$	$t_5 \rightarrow \pi_2$

The initial part of the particle

$t_1 \rightarrow t_4 \rightarrow t_2 \rightarrow t_3 \rightarrow t_5$

The second part of the particle

Fig.2 Parameter initialization of MC-PSO

After the scrambling process, the particles specify various task mapping towards VM types and various task scheduling sequences. The time-slot-based heuristic scrambling process predicts the VM instances for leasing, and task scheduling is done for the leased instances.

3.3. Handling invalids task prioritization

A feasible task scheduling process needs to fulfil the data dependencies between various tasks. Moreover, those tasks are not fulfilled during the iterative process due to the stochastic PSO characteristics. This work needs to prioritize every task to acquire task scheduling sequences (ascending order). It specifies that task prioritization

needs to be higher than the parent task and cannot be the same as one another. Else, the task prioritization is invalid. The invalid task handling process is provided for repairing the tasks. The concept is to change the invalid task prioritization to provide the maximal prioritized value of the parent task to ensure that the changed prioritizes are not equal to any of the prioritized tasks. The prioritized task is higher than all the parent tasks with diverse priorities. When the task shows a valid/invalid prioritized task, it is changed to a valid one, added to V, and eliminated from T. Algorithm 2 depicts the invalid task prioritization process.

Algorithm 2: Invalid task prioritization

Input: Particles q and tasks $T = \{t_1, \dots, t_n\}$

Output: Repairing process (particle)

1. $\eta(1{:}n) = round\ (q(n+1{:}2n)$;
2. $V = \emptyset$;
3. Predict the incoming task t_{entry} with minimal prioritization;
4. $V = V \cup \{t_{entry}\}$ and $T = \frac{T}{\{t_{entry}\}}$;
5. **while**$Task = 0$
6. Predict the task with parents V specified by Q;
7. $v^{max} = maximal\ (task\ prioritization)$;
8. **for** (all tasks $t_j \to Q$)
9. $p^{max} = maximal\ \left(t_j\ prioritization\right)$
10. **if**$\eta(j) \le p^{max}$
11. $\eta(j) = p^{max} + 1$
12. **end if**
13. $V = V \cup \{t_j\}$ and $T = \frac{T}{\{t_j\}}$;
14. **end for**
15. **end while**
16. $q(n+1{:}2n) = \eta(1{:}n)$;
17. Attain repaired particle; //output

3.4. Time-slot based re-scrambling process

The time-slot-based re-scrambling process is anticipated to re-scramble the particle towards the scheduling process. The process uses the available (idle) time slot for leasing the VM to enhance resource utilization and reduce the workflow execution cost. To preserve the data transfer time, some successive instances are provided to t_j, which is chosen first from X and recorded to H. If there are no probable instances $t_j \to X$, then the set is provided as empty and continuous the parallel processing which is applicable for $t_j \to Y$. There are no probable instances that are adopted for scheduling $t_j \to Y$. Finally, λ_n with π_j is launched over t_j. There are successive instances that show smaller costs with differences in execution costs before and after the scheduling process. When the number of instances with a smaller cost difference is H, the samples are chosen randomly from t_j.

Algorithm 3: Time slot based re-scrambling process

Input: Consider particle $q(1: 2n)$

Output: Provide scheduling solution $S = (R, M, \Theta)$

1. Set $R = \emptyset, M = \emptyset, H = \emptyset$;
2. Perform repair over invalid task prioritization;
3. Attain probable scheduling sequence relies on the available particles;
4. **For** all tasks $t_j, j \in \{1,2, \dots, n\}$ in Θ do
5. $\pi_j = q(j)$;
6. choose parallel and serial samples with VM and specified as X and Y;
7. Search for suitable samples and record the samples over H;
8. **If**H is empty, then
9. Perform searching process with suitable samples from Y; //record them over H;
10. **end if**
11. **If**$H = not\ empty$, then
12. allocate $t_j \rightarrow \lambda_a \in H$ with the least cost differences;
13. **else**
14. Promote newer samples λ_n with $\pi_j \rightarrow t$;
15. **end if**
16. update $M\ and\ R$;
17. **end for**
18. $S = (R, M, \Theta)$; //output

3.5. Fitness function execution

Algorithm 4 depicts the evaluation of the fitness function. Here, total execution cost during scheduling the solution is acquired from the particles adopted for fitness value computation. The total execution time S_i is adopted to check whether the particle is suitable or not. When the total execution time is superior to deadline constraint, then particles $q_1\ and\ q_2$ is infeasible; otherwise, it is viable. For instance, when the total execution cost (particles) q1 is lesser than other particles q2, q1 is superior to q2.

Algorithm 4: Fitness function computation

Input: Extract solution $S_i = (R, M, \Theta)$ of particle

Output: Workflow time and execution cost, $\Psi\ and\ \Omega$;

1. Ψ is evaluated using R and S_i;
2. Ω is evaluated based on M of S_i;
3. extract the value of $\Omega\ and\ \Psi$;

3.6. Population initialization

To extract the valid particles N with probable task scheduling where the task ranks (upward and downward), tasks are utilized for the rank-based scheduling process. It is provided to evaluate the task prioritization process with the initial population. The task-based upward ranking is measured with the crucial path from t_i to the task existing process. It also includes the average execution time. The down-ranking t_i shows

the distance among the entry tasks by excluding the execution time. During the initialization process, the populations O_1 and O_2 are produced, and every population possesses N particles. For all particles O_1, the dimensionality value $i \in [n+1, 2n]$ is given with a downward ranking process with task $t_{i-n.}$ However, the model task scheduling sequences are provided by sorting the prioritized task in ascending order where the priority (all tasks) is O_2 and set as $\sigma - r_u\ (t_i)$, where σ specifies the maximal values for tasks by upward ranking and $r_u(t_i)$ represents the link with $t_i\ upward$. At last, the particles are chosen as initial population from O_1 and O_2 in ascending order, providing superior fitness value.

4. Numerical results and discussion

The simulation is done in MATLAB 2020a environment, and various workflows scheduling process is considered for evaluating the proposed MC-PSO model. Also, various deadlines are considered for computing the ability of MC-PSO to fulfil one of the multi-objective constraints (deadline). The least execution time ζ specifies the workflow time (task) scheduled with the same VM instances and VM type. The execution time δ is evaluated based on the separate task scheduling using VM instances with VM types. The transmission time is avoided while evaluating the values ζ and δ.However, the least execution time specifies the order of magnitude superior to faster one, and difference between them is based on the workflow schedule. The comparison is made with various VM types known as computing units (See Fig 3 and Fig 4). The bandwidths among all these VM instances are equal to 20 Mbps. Table 1 depicts the sample VM types.

Table 1 VM types

Types	Computing unit	Capacity
m3 (medium)	3	13,250
m3 (large)	6.55	28,000
m3 (xlarge)	13.1	57,000
m3 (2*large)	26.1	114,000
c3 (medium)	7.1	30,000
c3 (large)	14	60,000
c3 (xlarge)	28	123,000
c3 (4*large)	55	243,000
r3.large	13	57,000
r3. 4*large	52	228,000

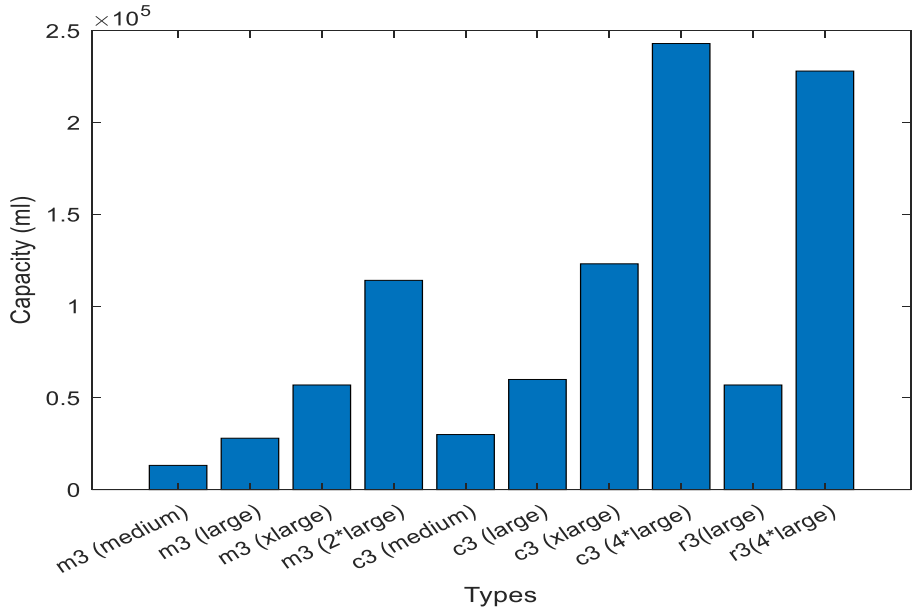

Fig 3 Comparison of VM types

To validate the performance of MC-PSO, six diverse approaches for comparison of workflow scheduling process known as standard PSO, GA, ACO, ICP, JIT, and SPSO are considered. GA concept is a multi-

objective optimization approach that simultaneously reduces the workflow cost and execution time. Similarly, SPSO adopts the concept of PSO for resolving the cost minimization and deadline constraint problem with the consideration of preliminary features of computing resources, dynamic provisioning, and VM performance variation. ICP uses a partial critical path and various tasks and schedules the lease instance that fulfils the task completion time. If no instances prevail over the existing path, some new instances with the least VM can complete the task before the final finish time is leased. JIT exploits the benefits of CC by considering the VM performance, i.e. delay acquisition and variability, to predict the scheduling time or deadline constraints. The reason for selecting these approaches relies on the process of task mapping to VM instances, VM instance mapping to types, and order of task scheduling. These existing processes rely on execution cost minimization and deadline constraints to handle the infeasible solutions.

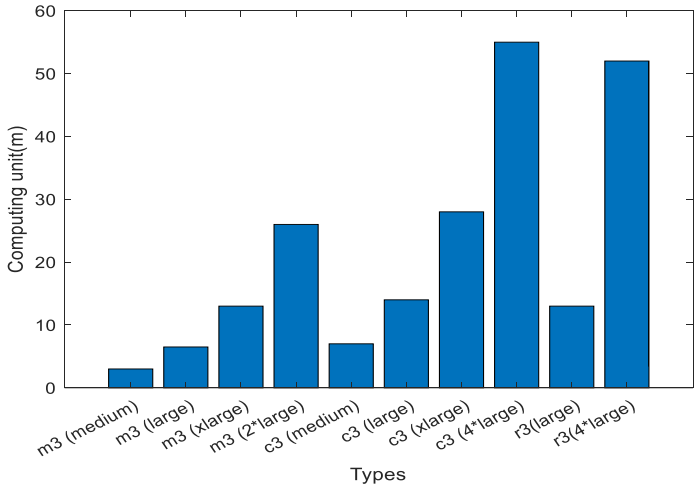

Fig 4 Computing capacity evaluation

The proposed MC-PSO does not possess any newer parameters for task mapping; however, it includes the PSO parameters (number of fitness functions, population size N, inertia weight ω and acceleration factor c_1&c_2). Table 2 depicts the execution cost of various factors. Fig 5 to Fig 7 and Table 3 depict the comparison of several evaluations.

Table.2 Execution cost comparison

ω	(c_1, c_2)		
	(2,2)	**(2-0, 0-2)**	**(1-0, 0-1)**
0.5	2.5	2.6	2.45
0.9 – 0.4	2.5	2.5	2.6
0.1 – 0.01	2.7	**2.4**	2.41

Table.3 Parameter evaluation

Size (N)	**Algorithms**	**No. of fitness function (K)**					
		1000	**2000**	**4000**	**6000**	**8000**	**10,000**
20	**MC-PSO**	2.35	2.27	2.18	2.04	1.81	2.05
	GA	2.41	2.38	2.29	2.15	1.92	2.15
	ACO	2.81	2.76	2.67	2.72	2.63	2.66
	ICP	16.9	11.75	9.60	8.25	6.93	6.06
	JIT	21.70	18.26	17.50	13.08	14.93	13.04

50	**MC-PSO**	1.95	2.05	1.84	1.83	1.84	1.85
	GA	2.04	2.16	1.95	1.94	1.94	1.95
	ACO	2.74	2.70	2.70	2.65	2.52	2.59
	ICP	11.29	9.20	8.35	6.70	5.80	5.05
	JIT	23.03	20	16.46	12.70	14.40	15.70
100	**MC-PSO**	1.98	2.02	1.92	1.96	1.85	1.80
	GA	2.05	2.04	1.94	1.96	1.90	1.80
	ACO	2.70	2.66	2.57	2.59	2.52	2.58
	ICP	11.13	8.31	6.43	5.87	5.03	4.69
	JIT	22.85	20.63	17.33	14.08	11.80	14.45

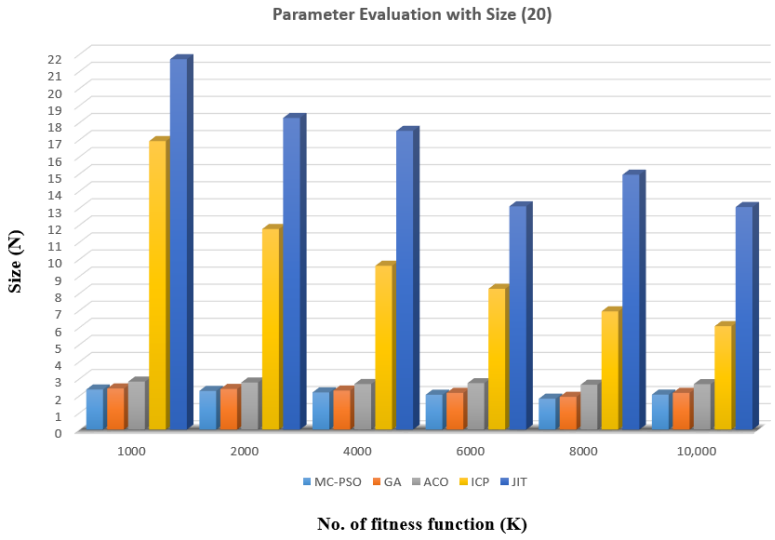

Fig 5 Parameter evaluation with size 20 (Iteration 1)

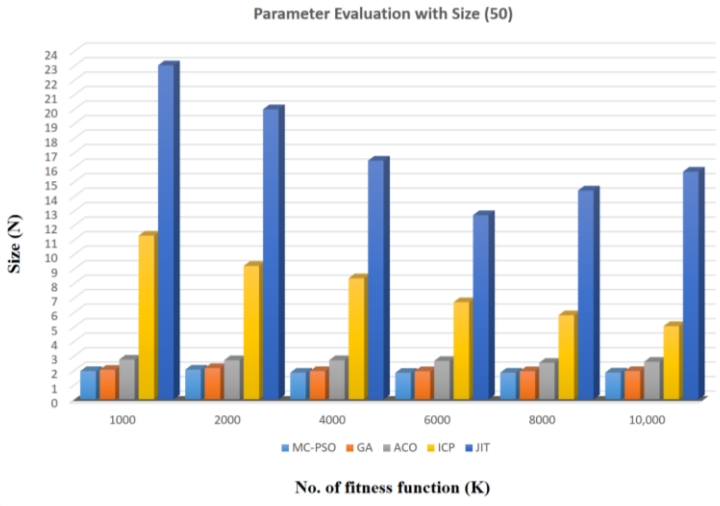

Fig 6 Parameter evaluation with size 50 (Iteration 2)

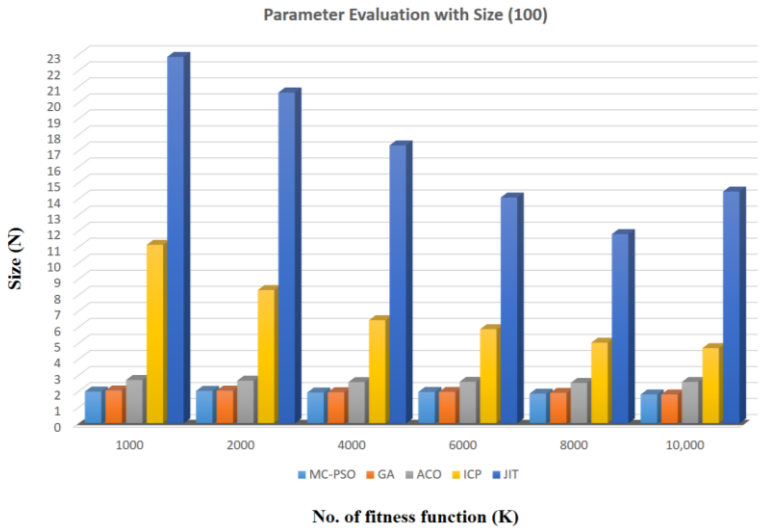

Fig 7 Parameter evaluation with size 100 (Iteration 3)

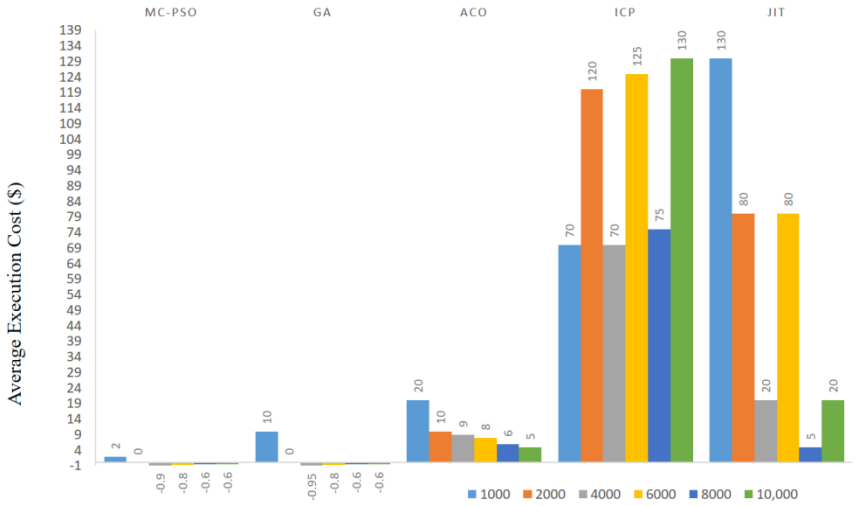

Fig 8 Average execution cost computation

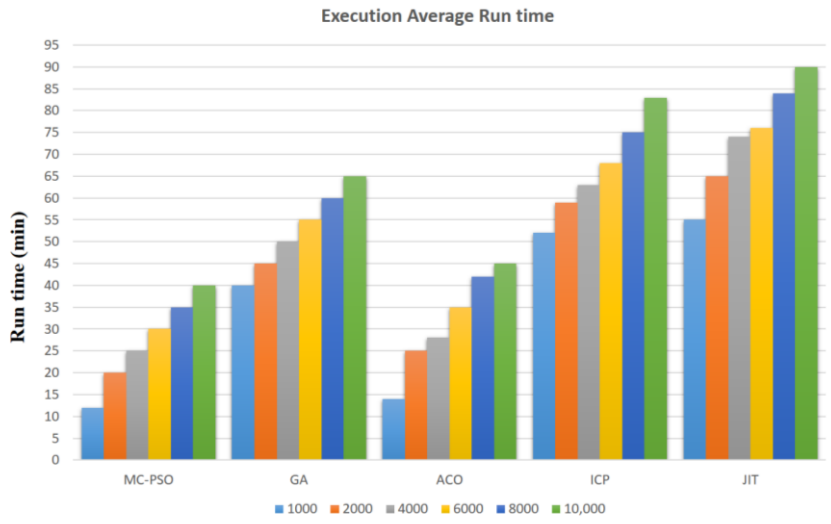

Fig 9 Average runtime execution

The methods used for comparison show 10 independent runs for every workflow application. With the increased N and K, the execution cost of ICP and JIT are higher, while the execution cost of GA and ACO does not show much variation. When N and K's value is extremely large, the existing approaches consume huge time for larger workflows. Therefore, the N values are set as 20. For the workflow deadline d_i, if the average workflow execution time is lesser than or equal to d_i, the proposed model fulfils the deadline constraints. It adopts the success rate of s^r, known as the deadline percentage fulfilled by the algorithm among various deadlines, to compute the ability to fulfil the workflow deadlines (See Fig 7). It is expressed as in Eq. (3):

$$s^r = \frac{k}{\varphi} * 100\% \quad (3)$$

sHere, k specifies the number of deadlines, φ specifies the total number of deadlines. The algorithm performance in reducing the execution cost is computed with an average execution cost of the scheduled workflow. Generally, the proposed MC-PSO attains lower execution costs compared to other approaches. Some existing rule-based approaches predict some possible outcomes for every workflow; however, the solution is lesser than the existing approaches. Some models show worse as the execution time attained in diverse runs are different signs which outcomes in higher average cost. The anticipated MC-PSO shows superior performance as it integrates both the standard PSO and time-based rule concept with superior benefits to the other approaches. The proposed MC-PSO is improved to examine the consequences of various task scheduling sequences during workflow execution cost. The model uses only a single task scheduling strategy acquired by varying all the tasks topologically. Thus, the particles over MC-PSO with task scheduling sequence require task mapping towards VM types, and it does not include task prioritization. Generally, the meta-heuristic optimization approach-based scheduling process requires longer computational time compared to various heuristic rules. The run time is computed by evaluating the run time of the algorithm for all workflow samples and dividing the overall workflow instances (See Fig 8).

5. Conclusion

This research concentrates on providing a novel variant of PSO for scheduling process by considering three factors like selecting suitable VM for every task, provisioning task scheduling sequentially, and idle time slot utilization. The proposed MC-PSO intends to resolve the issue as the particles specify the VM mapping essential for sequential task scheduling. Idle time-slot-based scheduling rules are anticipated to re-scramble the particles for scheduling. While in re-scrambling, the rules allocated for every task establishes schedule sequences to lease VM and consumes idle time slots. The anticipated model adopts repairing mechanism to handle the invalid task prioritization. The numerical outcomes rely on the workflow scheduling model attaining a 99% success rate by fulfilling the research objectives and outperforming various existing approaches. However, compared to other approaches, the proposed MC-PSO model gives superior performance in reducing the overall execution time. This research has laid a solid foundation by designing a Multi-Constraint PSO algorithm that effectively addresses resource allocation and task scheduling challenges. The findings presented here, along with the identified future research directions, contribute to advancing the field of optimization and offer valuable insights for researchers and practitioners working in complex systems optimization and management. However, in the future, this work will be executed over any real-time prototype to test the feasibility of the model.

Declaration:

Ethics Approval and Consent to Participate:

No participation of humans takes place in this implementation process

Human and Animal Rights:

No violation of Human and Animal Rights is involved.

Funding: No funding is involved in this work.

Conflict of Interest: Conflict of Interest is not applicable in this work.

Data Availability:

Data available on request from the authors

Acknowledgment:

No acknowledge

References

[1] Xu, K., Zhang, Y., Shi, X., Wang, H., Wang, Y., Shen, M.: Online combinatorial double auction for mobile cloud computing markets. In: 2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC), pp.1–8. IEEE (2014)

[2] Mitrović-Minić, S., Punnen, A.P.: Local search intensified: very large-scale variable neighbourhood search for the multi-resource generalized assignment problem. Discreet. Optim. 6(4), 370–377 (2009)

[3] Gavranović, H., Buljubašić, M.: An efficient local search with a noising strategy for Google machine

reassignment problem. Ann. Oper. Res. 242, 1–13 (2014)
[4] Mann, Z.D., Szabó, M.: Which is the best algorithm for virtual machine placement optimization? Concurr. Comput. Pract. Exp. 29(7), e4083 (2017)
[5] Sharma, P., Chaufournier, L., Shenoy, P., Tay, Y.C.: Containers and virtual machines at scale: a comparative study. In: International Middleware Conference, p. 1 (2016)
[6] Meng, X., Pappas, V., Zhang, L.: Improving the scalability of data centre networks with traffic-aware virtual machine placement. In: INFOCOM, 2010 Proceedings IEEE, pp. 1–9 (2010)
[7] Popa, L., Kumar, G., Chowdhury, M., Krishnamurthy, A., Ratnasamy, S., Stoica, I.: Faircloud: sharing the network in cloud computing. ACM SIGCOMM Comput. Commun. Rev. 42(4), 187–198 (2012)
[8] Li, X., Wu, J., Tang, S., Lu, S.: Let's stay together: towards traffic-aware virtual machine placement in data centres. In: INFOCOM, 2014 Proceedings IEEE, pp. 1842–1850 (2014)
[9] Lakers, K., Deng, S., Goyal, A., Balakrishnan, H.: Choreo: network-aware task placement for cloud applications. In: Conference on Internet Measurement Conference, pp. 191–204 (2013)
[10] Zhao, Y., Huang, Y., Chen, K., Yu, M., Wang, S., Li, DS: Joint VM placement and topology optimization for traffic scalability in dynamic datacenter networks. Comput. Netw. 80, 109–123 (2015)
[11] Ma, T., Wu, J., Hu, Y., Huang, W.: Optimal VM placement for traffic scalability using Markov chain in cloud data centre networks. Electron. Lett. 53(9), 602–604 (2017)
[12] Wang, L., Zhang, F., Aroca, J.A., Vasilakos, A.V., Zhang, K., Hou, C., Li, D., Liu, Z.: Greendcn: a general framework for achieving energy efficiency in data centre networks. IEEE J. Sel. Areas Commun. 32(1), 4–15 (2013)
[13] Rai, A., Bhagwan, R., Guha, S.: Generalized resource allocation for the cloud. In: ACM Symposium on Cloud Computing, pp. 1–12 (2012)
[14] Gu, L., Zeng, D., Guo, S., Xiang, Y., Hu, J.: A general communication cost optimization framework for big data stream processing in geo-distributed data centres. IEEE Trans. Comput. 65(1), 19–29 (2015)
[15] Rui, L., Zheng, Q., Li, X., Jie, W.: A novel multi-objective optimization scheme for rebalancing virtual machine placement. In: IEEE International Conference on Cloud Computing, pp. 710–717 (2017)
[16] Shen, M., Xu, K., Li, F., Yang, K., Zhu, L., Guan, L.: Elastic and efficient virtual network provisioning for cloud-based multi-tier applications. In: 2015 44th International Conference on Parallel Processing (ICPP), 929–938. IEEE (2015)
[17] Taleb, T., Bagua, M., Ksentini, A.: User mobility-aware virtual network function placement for virtual 5G network infrastructure. In: IEEE International Conference on Communications, pp. 3879–3884 (2016)
[18] Wang, T., Xu, H., Liu, F.: Multi-resource load balancing for virtual network functions. In: IEEE International Conference on Distributed Computing Systems (2017)
[19] Marotta, A., Kassler, A.: A power-efficient and robust virtual network function placement problem. In: Teletraffic Congress, pp. 331–339 (2017)
[20] Kawashima, K., Otoshi, T., Ohta, Y., Murata, M.: Dynamic placement of virtual network functions based on model predictive control. In: NOMS 2016 – 2016 IEEE/IFIP Network Operations and Management Symposium, pp. 1037–1042 (2016)
[21] Mehraghdam, S., Keller, M., Karl, H.: Specifying and placing chains of virtual network functions. In: 2014 IEEE 3rd International Conference on Cloud Networking (CloudNet), pp. 7–13. IEEE (2014)
[22] Addis, B., Belabed, D., Bouet, M., Secci, S.: Virtual network functions placement and routing optimization. In: IEEE International Conference on Cloud Networking, pp. 171–177 (2015)
[23] Wang, F., Ling, R., Zhu, J., Li, D.: Bandwidth guaranteed virtual network function placement and scaling in datacenter networks. In: IEEE International Performance Computing and Communications Conference, pp. 1–8 (2015)
[24] Bhamare, D., Samara, M., Abad, A., Jain, R., Gupta, L., Chan, H.A.: Optimal virtual network function placement in multi-cloud service function chaining architecture. Comput. Commun. 102(C), 1–16 (2017)
[25] Ghaznavi, M., Khan, A., Shahriar, N., Alsubhi, K., Ahmed, R., Boutaba, R.: Elastic virtual network function placement. In: IEEE International Conference on Cloud Networking (2015)
[26] Marotta, A., D'andreagiovanni, F., Kassler, A., & Zola, E. (2017). On the energy cost of robustness for green virtual network function placement in 5G virtualized infrastructures. Computer Networks, 125, 64-75.
[27] Taleb, T., Bagua, M., Ksentini, A.: User mobility-aware virtual network function placement for Mei, C., Liu, J., Li, J., Zhang, L., & Shao, M. (2020). 5G network slices embedding with sharable virtual

network functions. Journal of Communications and Networks, 22(5), 415-427.

[28] Zhang, Q., Xiao, Y., Liu, F., Lui, J.C.S., Guo, J., Wang, T.: Joint optimization of chain placement and request scheduling for network function virtualization. In: IEEE International Conference on Distributed Computing Systems, pp. 731–741 (2017)

[29] Larissa, A., Taleb, T., Bagua, M., Flinck, H.: Towards edge slicing: VNF placement algorithms for a dynamic & realistic edge cloud environment. In: 2017 IEEE Global Communications Conference, pp. 1–6. IEEE (2017)

[30] Laghrissi, A., & Taleb, T. (2018). A survey on the placement of virtual resources and virtual network functions. IEEE Communications Surveys & Tutorials, 21(2), 1409-1434.

[31] Sherubha, P., Sasirekha, S.P., Manikandan, V.: Graph based event measurement for analyzing distributed anomalies in sensor networks. Sadhana 45, 212 (2020).

[32] Zhang, Xinqian.: Energy-aware virtual machine allocation for cloud with resource reservation. Journal of System Software 147 (2019): 147-161.

[33] Kholidy, Hisham A.: An Intelligent Swarm Based Prediction Approach for Predicting Cloud Computing User Resource Needs. Computer Communications, vol. 151, (2020), pp. 133–44.

[34] Bhattacherjee, S., Das, R., Khatua, S. et al. Energy-efficient migration techniques for cloud environment: a step toward green computing. J Supercomput 76, 5192–5220 (2020).

[35] Agrawal, S.A., Umbarkar, A.M., Sherie, N.P., Dharme, A.M., Dhabliya, D. Statistical study of mechanical properties for corn fiber with reinforced of polypropylene fiber matrix composite (2021) Materials Today: Proceedings,

[36] Sherje, N.P., Agrawal, S.A., Umbarkar, A.M., Dharme, A.M., Dhabliya, D. Experimental evaluation of Mechatronics based cushioning performance in hydraulic cylinder (2021) Materials Today: Proceedings,