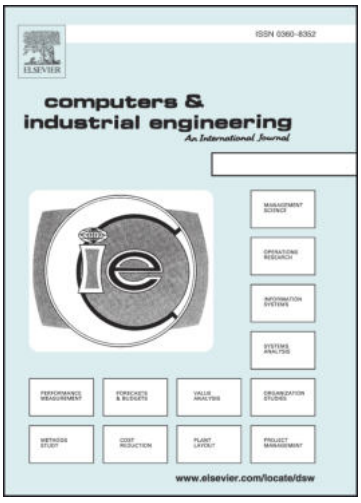
Journal Pre-proofs



Improved Many-objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing

Sahar Saeedi, Reihaneh Khorsand, Somaye Ghandi Bidgoli, Mohammadreza Ramezanpour

PII: S0360-8352(20)30383-1
DOI: https://doi.org/10.1016/j.cie.2020.106649
Reference: CAIE 106649

To appear in: *Computers & Industrial Engineering*

Received Date: 10 January 2020
Revised Date: 23 June 2020
Accepted Date: 4 July 2020

Please cite this article as: Saeedi, S., Khorsand, R., Ghandi Bidgoli, S., Ramezanpour, M., Improved Many-objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing, *Computers & Industrial Engineering* (2020), doi: https://doi.org/10.1016/j.cie.2020.106649

Improved Many-objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing

Sahar Saeedi[1]. Reihaneh Khorsand[2]. Somaye Ghandi Bidgoli [3]. Mohammadreza Ramezanpour [4]

Corresponding Author: Reihaneh Khorsand
[1,2]Department of Computer Engineering, Dolatabad Branch, Islamic Azad University, Isfahan, Iran
[1]E-mail:R.khorsand@iauda.ac.ir, Reihaneh_khm@yahoo.com
[2]E-mail:sahar.saeedi1817@gmail.com

[3] Department of Industrial Engineering, Faculty of Engineering, University of Kashan, Kashan, Iran
[3]E-mail:S.ghandi@kashanu.ac.ir
[4]Department of computer engineering, Mobarakeh branch, Islamic Azad University, Mobarakeh, Isfahan, Iran
[4]E-mail: ramezanpour@mau.ac.ir

Sahar Saeedi received the M.S. degrees in Software Engineering from Dolatabad Branch, Islamic Azad University, Isfahan, Iran, in 2019. Her research interests include Distributed Computing, Cloud Computing, Autonomic Computing, Edge/Fog Computing, and Soft Computing.
Reihaneh Khorsand received her Ph.D. (Software engineering) in 2017 from Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. She is assistant professor of Computer Engineering Department, Dolatabad Branch, Islamic Azad University, Isfahan, Iran. Her research interests are software engineering, distributed computing, scheduling and business workflow management.
Somaye Ghandi Bidgoli her Ph.D. (Industrial Engineering) in 2015 from Tarbiat Modares University (TMU), Tehran, Iran. She is assistant professor of Industrial Engineering Department, Faculty of Engineering, University of Kashan, Kashan, Iran. Her research interests are Industrial engineering, Multi-objective optimization, Path Planning Problems
Mohammadreza Ramezanpour received his B.Sc. degree in Computer Engineering from Islamic Azad University, Kashan, Iran. His M.Sc. degree from Islamic Azad University, Arak, Iran, and his Ph.D. degree in computer engineering from Science and Research Branch, Islamic Azad University, Tehran, Iran. His research interests include computer vision and image processing, pattern recognition and video coding standards

Improved Many-objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing

Sahar Saeedi[1]. Reihaneh Khorsand[2]. Somaye Ghandi Bidgoli [3]. Mohammadreza Ramezanpour [4]

Abstract

Optimized scientific workflow scheduling can greatly improve the overall performance of cloud computing. As workflow scheduling belongs to NP-complete problem, so, meta-heuristic approaches are more preferred option. Most studies on workflow scheduling in cloud mostly consider at most two or three objectives and there is a lack of effective studies and approaches on problems with more than three objectives remains; because the efficiency of multi-objective evolutionary algorithms (MOEAs) will seriously degrade when the number of objectives is more than three, which are often known as many-objective optimization problems (MaOPs). In this paper, an approach to solve workflow scheduling problem using Improved Many Objective Particle Swarm Optimization algorithm named I_MaOPSO is proposed considering four conflicting objectives namely maximization of reliability and minimization of cost, makespan and energy consumption. Specifically, we use four improvements to enhance the ability of MaOPSO to converge to the non-dominated solutions that apply a proper equilibrium between exploration and exploitation in scheduling process. The experimental results show that the proposed approach can improve up to 71%, 182%, 262% the HyperVolume (HV) criterion compared with the LEAF, MaOPSO, and EMS-C algorithms respectively. I_MaOPSO opens the way to develop a scheduler to deliver results with improved convergence and uniform spacing among the answers in compared with other counterparts and presents results that are more effective closer to non-dominated solutions.[1]

Keywords: Cloud computing, Many-objective PSO, Workflow scheduling.

1. Introduction

Cloud computing is a popular computational paradigm for affordable and easy-to-use computations. Virtualization is the main empowering technology in cloud computing, that is used to divide a physical machine into several Virtual Machines (VMs) in a cost-effective manner [1-3]. Virtualization is based on a market-oriented paradigm where customers are charged based on their consumption and can consume these services based on Service Level Agreements (SLAs) [4]. A workflow is a set of dependent or independent tasks that is illustrated as a Directed Acyclic Graph (DAG) in which the nodes indicate the tasks and a directed edge indicates the dependency among the corresponding tasks. To implement such workflow, required user resources are provided as VMs by the Infrastructure as a service (IaaS) in cloud [5-7]. The main complexity in this area is deciding the order in which the tasks will be executed and the optimum task-to-VM mapping, to achieve specified performance conditions that is known as a workflow-scheduling problem.

1.1. Motivation

The motivations of this paper are as follows:

- At present, the main challenge is that the existing researches addresses workflow scheduling in cloud as a single-objective [1], bi-objective[8], or three-objective optimization problems[9] by either making unreal suppositions that cannot be happened in many application scenarios, or neglecting some significant objectives of the service provider or the users [5,10]. Some of the MOEAs such as Non-Dominated Sort Genetic Algorithm II (NSGA- II) [11], Strength Pareto Evolutionary Algorithm 2 (SPEA2) [12] and Speed-Constrained Multi-Objective Particle Swarm Optimization (SMPSO) [13], use the Pareto-dominant relationship to determine the quality of the solutions. Hence, they are known as Pareto-based algorithms. However, the capabilities of these algorithms are greatly reduced when the number of objectives exceeds

Corresponding Author: Reihaneh Khorsand
[1,2]Department of Computer Engineering, Dolatabad Branch, Islamic Azad University, Isfahan, Iran
[1]E-mail:R.khorsand@iauda.ac.ir, Reihaneh_khm@yahoo.com
[2]E-mail:sahar.saeedi1817@gmail.com
[3] Department of Industrial Engineering, Faculty of Engineering, University of Kashan, Kashan, Iran
[3]E-mail:S.ghandi@kashanu.ac.ir
[4]Department of computer engineering, Mobarakeh branch, Islamic Azad University, Mobarakeh, Isfahan, Iran
[4]E-mail: ramezanpour@mau.ac.ir

three objectives, MOEAs may end up with a set of well-distributed non-dominated solutions, which are unfortunately far from non-dominated solutions. Therefore, one of the motivations of this paper is to study the problem of scheduling of scientific workflows as a many-objective optimization problem.

- In cloud workflow scheduling, quality parameters form an important role. Among these parameters makespan, cost, energy efficiency and reliability need more attention. Considering makespan ensure that workflow is completed within the desired time or the specified deadline [14]. The cost ensures the budget determined by the user is not overshot. Energy efficiency is a significant objective for computing infrastructure providers due to its environmental and economic implications. The failure rate of a VM during the execution of the workflow is the reliability of the VM. Here might be a state of conflict within these QoS requirements/objectives, where there exist no single solution. For example, in order to increase the reliability of a VM, you should use a higher-power VM that will naturally cost more. We also need to use faster and more expensive resources to reduce the makespan. Therefore, one of the motivations of this paper is to study the problem of scheduling of scientific workflows on multiple VMs with considering makespan, cost, energy efficiency and reliability in a cloud-computing environment.
- Considering an appropriate balance between exploration and exploitation. Therefore, how to search in more wide space including individual's selection and dimension's selection is one of the motivations of this paper.

1.2. Contributions

In this paper, to enhance the ability of MOEAs to converge to the non-dominated solutions, an improved Many-Objective Particle Swarm Optimization (I_MaOPSO) algorithm is proposed for dynamic workflow scheduling in cloud environments. The goal of the proposed approach is to meet the many-objective QoS requirements, in both cloud users' and providers' context by minimizing makespan, cost and maximization reliability for users, and minimizing energy consumption for providers. Addressing more than three objectives simultaneously in workflow scheduling is one aspect of innovation in this work, which has not been addressed so far. MaOPSO uses a set of reference points dynamically determined according to the search process, allowing the algorithm to converge to the non-dominated solutions. The main contributions of the paper to further improve the performance of MaOPSO and its use in cloud based workflow scheduling are summarized as follows.

- Proposing four greedy heuristic methods to generate uniformly distributed particles to improve the quality of the initial population.
- Applying an efficient approach to compute the velocity of particles in order to achieve an appropriate balance between exploration and exploitation.
- Applying a roulette wheel selection process instead of the tournament selection to enhance the selecting the social leader.
- Applying an efficient approach to choose the new cognitive leader in order to achieve a balance between the diversification and intensification capabilities of the I_MaOPSO algorithm
- Comparing the performance of the proposed approach with other related approaches, where it is found that the proposed approach outperforms its counterparts.

The remaining paper is organized as follow: Section 2 presents the necessary background for this study. The related work done on multi-objective and many-objective optimization problems for workflow scheduling is discussed in Section 3. Section 4 explains the proposed approach and the details of the I_MaOPSO algorithm are given, Section 5 presents the simulation strategy and results analysis. Finally, conclusions and future work are provided in Section 6.

2. Background

2.1. Many-objective optimization problems

Multi-objective optimization problems (MOPs) are characterized by multiple objectives which conflict with each other. Due to the conflicting nature of the objectives, usually no single optimal solution exists; instead, a set of trade-off solutions, known as non-dominated solutions can be found for MOPs. Over the past two decades, evolutionary

algorithms (EAs) and other population based meta-heuristics have been demonstrated to be a powerful framework for solving MOPs, since they can find a set of non-dominated solutions in a single run.

However, the efficiency of Pareto-based multi-objective evolutionary algorithms (MOEAs) will seriously degrade when the number of objectives is more than three, which are often known as many-objective optimization problems (MaOPs). The main reason for this performance deterioration is that the selection criterion based on the standard dominance relationship fails to distinguish solutions in a population already in the early stage of the search, since most of the solutions in the population are non dominated, although some of them may have a better ability to help the population to converge to the non-dominated solutions. Once the dominance based selection criterion is not able to distinguish solutions, MOEAs may end up with a set of well-distributed non-dominated solutions, which are unfortunately far from non-dominated solutions.

Figueiredo et al. [12] to enhance the ability of MOEAs to converge to the non-dominated solutions proposed Many-Objective Particle Swarm Optimization (MaOPSO) that uses a set of reference points dynamically determined according to the search process, allowing the algorithm to converge to the non-dominated solutions, but maintaining the diversity of the non-dominated solutions. MaOPSO started by generating N particles randomly in the decision space using a uniform distribution to form the initial swarm S0. Then, the particles are evaluated using a fitness assignment method. MaOPSO also has an external archive that is empty at first. Given that social leader selection process is used to differentiate between solutions within the external archive a set of well-distributed reference points is generated to be used during this process. Then, The algorithm applies iteratively a series of steps that involves: (1) Select the cognitive and social leaders for the particles from the external archive (2) Apply polynomial mutation to 15% of the particle swarm in (3) Update the external archive using the new solutions visited by the particles. (4) Prune the external archive when its maximum size excesses. The above steps are repeated until the termination condition is reached.

Chen et al. [15] first defines the no dominated solutions exhibiting evident tendencies toward the Pareto-optimal front as prominent solutions, using the hyperplane formed by their neighboring solutions, to further distinguish among no dominated solutions. Then, a novel environmental selection strategy is proposed with two criteria in mind: 1) if the number of non-dominated solutions is larger than the population size, all the prominent solutions are first identified to strengthen the selection pressure. Subsequently, a part of the other non-dominated solutions is selected to balance convergence and diversity and 2) otherwise, all the non-dominated solutions are selected; then a part of the dominated solutions are selected according to the predefined reference vectors. Moreover, based on the definition of prominent solutions and the new selection strategy, they proposed a hyperplane assisted evolutionary algorithm, referred here as hpaEA, for solving MaOPs.

He et al. [16] proposed a radial space division based evolutionary algorithm for many-objective optimization, where the solutions in high-dimensional objective space are projected into the grid divided 2-dimensional radial space for diversity maintenance and convergence enhancement. Specifically, the diversity of the population is emphasized by selecting solutions from different grids, where an adaptive penalty based approach is proposed to select a better converged solution from the grid with multiple solutions for convergence enhancement.

Sharma et al. [17] proposed an efficient environmental selection and normalization scheme for NSGA-III algorithm. The environmental selection operator was developed to equally prioritize solutions associated with different lines drawn from the origin and the reference points. Actually, the line-prioritized environmental selection is proposed to select at least one solution representing each reference line to the next generation population. A normalization scheme was also suggested in which the extreme point is used which gets updated on the designed rules. For future studies, their approach can be extended for solving constraint multi-objective optimization problems. The differences between the proposed algorithms in the mentioned papers in this subsection and the proposed I_MaOPSO algorithm in this paper and the strength of our proposed algorithm are listed and explained in Table 1.

Table 1. The strength and the differences between the proposed algorithm and the many objective algorithms

Ref	Algorithm	Differences	The strength of I_MaOPSO algorithm over the mentioned algorithm/ the shortcomings of the mentioned algorithm
[12]	**Ma**ny **O**bjective **P**article **S**warm **O**ptimization (MaOPSO)	1-The initial population generation method.	1- The proposed method leads to generating uniformly distributed particles and improving the quality of the initial population.

		2-The velocity of particles computing approach.	2- An efficient approach is applied in order to achieve an appropriate balance between exploration and exploitation.
		3-The social leader selecting process.	3- The used roulette wheel selection gives a chance to all of solutions in the archive to be selected, contrary to the truncation selection used in MaOPSO.
		4-The cognitive leader choosing approach.	4- The main advantage of this approach is that during the search process and creating the population members, the search is not confined to only a reduced number of regions (e.g., around the ideal point).
[15]	**h**yper**p**lane **a**ssisted **E**volutionary **A**lgorithm (hpaEA)	hpaEA and I_MaOPSO can deal with the balance of convergence and diversity to a certain extent. However, the strategy of hpaEA to enhance convergence and diversity is separate, which may weaken the communication of the information. However, I_MaOPSO tries at eliminating the drawbacks of current approaches in the balancing between convergence and diversity.	1- hpaEA optimizer is overspecialized for addressing several types of benchmark functions and end up sacrificing performance on real world scenarios [18].
			2- hpaEA is specialized for handling large scale problems and its performance on small scale problems is unsure. Because for small-scale problems, this algorithm is trapped at local optima and do not converge properly to the true Pareto Front [19].
			3- hpaEAt give priority to convergence and then consider diversity alone. Convergence and diversity are treated separately with different strategies, which makes it difficult to make full use of current population information to achieve a balance of them [20].
[16]	**R**adial **S**pace division based **E**volutionary **A**lgorithm (RSEA)	Unlike I_MaOPSO, RSEA is a **S**urrogate-**A**ssisted **E**volutionary **A**lgorithm (SAEAs) and its performance highly depends on the number of objectives. SAEAs use efficient computational models, often known as surrogate models, for approximating the fitness function in evolutionary algorithms.	1- RSEA pays more attention to convergence maintenance than diversity maintenance [21].
			2- In RSEA it is important to decide which surrogate model should be used and when to use it [22].
			3- The surrogate model must be updated during the search process to achieve a better approximation of the objective functions. However, how to perform such an update is not straightforward [22].
[17]	**L**ine-Prioritized **E**nvironmental Selection **A**nd Normalization **F**ramework (LEAF)	Both of I_MaOPSO and LEAF algorithms use a similar framework of **N**on-dominated **S**ort **G**enetic **A**lgorithm **III** (NSGA-III). In LEAF, the non-dominated sorting and association are the same as the NSGA-III. However, the environmental selection and normalization are different. I_MaOPSO has an external archive in which the founded non-dominated solutions during the search process are stored whereas NSGA-III has only a population. In addition, an important difference between these algorithms is that I_MaOPSO employs Pareto dominance and information about density and proximity to push the particles towards the PF whereas NSGA-III does not use any explicit reproduction selection operation as the parents are randomly picked of the population.	1- I_MaOPSO use both extreme solutions and projected points generated by an interesting mechanism called ASF. This mechanism impose a pressure selection toward the entire Pareto front and promote the distribution of the solutions over this Pareto front and thanks to this mechanism, the I_MaOPSO algorithm obtained simultaneously a good convergence and a high diversity in a reasonable time.
			2- LEAF is emerged as one of the competitive algorithms and can be an alternative for many-objective optimization. However, in order to improve the performance of LEAF, it is better to use it with an algorithm in which the diversity is preserved first over the dominance (e.g., the **S**trength **P**areto **E**volutionary **A**lgorithm based on **R**eference direction (SPEA/R) algorithm) [17].

2.2. Workflow model

Workflow scheduling is the process of mapping task to the appropriate resource. Selecting resources and mapping tasks should be done in such a way that the quality requirements of the users and providers are met. A popular representation of a workflow model is the directed acyclic graph (DAG): G (T, E) as shown in Fig.1, where T is a set of *n* workflow tasks {WT_1, WT_2, . . .,WT_n}, and E is a set of directed edges. {e_{ij}| (t_i ,t_j) ∈ E} representing inter-task data dependencies such that the execution of $t_j \in T$ cannot be started before $t_i \in T$ finishes its execution. If there is data transmission from WT_i to WT_j, the WT_j can start only after all the data from WT_i has been received.

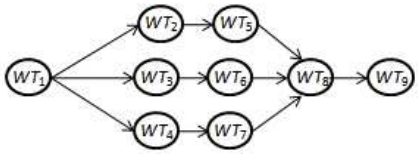


Fig. 1. An example of directed acyclic graph (DAG)

2.3. Cloud model

The cloud model is a cloud service provider that includes computational resources. VM_i is a virtual machine with different $P_{vm\,i}$ processing power and different costs C_{vmi}. It is supposed that each VM of the set *r* can execute all tasks of a workflow. The processing power of a resource, $P_{vmi} \in R$, is expressed in millions of instructions per second (MIPS). The pricing model used in this paper is based on pay-as-you-go model in which the users are payed based upon the number of time intervals that they have used the resources. It is assumed that there is no limitation on the number of VMs to lease from the provider. In this study, the VMs provided in the AmazonEC2 and its proposed memory usage characteristics are used, in which virtual machines have enough memory to perform the workflow tasks [23, 24]. It is also supposed that the VMs are all on a same network with similar bandwidth. The goal is to implement and present the proposed scheduling algorithm and to meet the quality requirements of users and cloud providers. The notations applied in our approach are tabulated in Table 2.

Table 2. Explanation of symbols used in a cloud workflow scheduling problem

$ET_{t_i}^{VM_j}$	Duration of execution of task ti on VM_j
$TT_{e_{ij}}$	The transmission time from task t_i to task t_j
$PT_{t_i}^{VM_j}$	Total processing time
Makespan	The overall schedule length of the workflow
TEC	Total execution cost
TET	Total execution time
Rel_i	Reliability of a VM
E	Energy consumption
ρ_i	Convergence criterion
λ	A variance factor of the workflow deadline

We target workflow applications such as those presented by Rodriguez et al. [24] and Juve et al. [25]. Based on the profiling results obtained in their work for memory consumption and the VM types offered by Amazon EC2, we assume that VMs have sufficient memory to execute the workflow tasks. We assume that for every VM type, the processing capacity in terms of Floating Point Operations per Second (FLOPS) is available either from the provider or can be estimated [26]. This information is used in our algorithm to calculate the execution time of a task on a given VM, similar to references [24, 25]. Therefore, the execution time $ET_{t_i}^{VM_j}$ of task t_i in a VM of type VMj is estimated through Eq. 1 [24].

$$ET_{t_i}^{VM_j} = \frac{I_{t_i}}{P_{VM_j} * (1 - deg_{VM_j})} \quad (1)$$

where I_{t_i} is the size of a task in the flop unit and deg_{vm_j} is the performance degradation percentage of a VM.

Additionally, the time taken for the data to be transferred from the parent workflow task t_i to the child workflow task t_j is indicated by $TT_{e_{ij}}$ which is calculated through Eq. 2 [24].

$$TT_{e_{ij}} = d_{t_i}^{out} / \beta \quad (2)$$

where $d_{t_i}^{out}$ is the output data size via t_j.

In this paper, it is assumed that the entire workflow is performed on a data center. So the bandwidth on VMs (β) is same, and the transfer time between two workflow tasks that runs on the same VM is zero. Total processing power is shown as $PT_{t_i}^{VM_j}$ [24].

$$PT_{t_i}^{VM_j} = ET_{t_i}^{VM_j} + (\sum_{1}^{k} TT_{e_{ij}} * s_k) \quad (3)$$

where k is the number of edges of a task. If both workflow tasks are on a VM, k is zero, otherwise it will be one.

Scheduler S is defined as: $S = (R, M, TEC, Makespan, Energy, Reliability)$, which includes sets of resources (VMs), mapping workflow task to resource, total execution costs, makespan, energy consumption and reliability. $R = \{r_1, r_2, ...r_n\}$ is the set of VMs that should be leased. Every resource r_i has one type of VM assigned along with information such as the start time of assigning the workflow task t_i to resource r_i (LST_{ri}) and the end time of assigning the workflow task t_i to resource r_i (LET_{ri}). M presents a collection of mappings, and a combination of multiple set $m_{t_i}^{r_j} = (t_i, r_j, ST_{t_i}, ET_{t_i}, Rel_{jk})$ that creates for each workflow task in the workflow. The mapping $m_{t_i}^{r_j}$ indicates that workflow task t_i is run on the resource r_j and it is expected that the start time of the workflow task is ST and its end time is ET and the error rate (reliability) in performing this workflow task by this resource is Rel_{jk}.

2.4. Optimization Objectives

a) Makespan

Makespan is equal to the maximum end time of the operation for the tasks of a workflow, includes the total time that the tasks are entered until their actual expiration and completion. In other words, makespan is equal to the time the entire DAG graph is executed. The optimum performance of a workflow requires a minimum makespan [24]:

$$Makespan = \max \{AFT(t_{exit})\} \quad (4)$$

where AFT denotes the finish time of task ti on resource rp. Furthermore, the makespan for executing a workflow determines whether the scheduling of the workflow can satisfy the deadline as defined through Eq.5.

$$MakeSpan + Arrival\ time\ of\ workflow <= Deadline\ of\ workflow \quad (5)$$

We apply two metrics related to deadline violation shown in Eqs. (6) [22] and (7), respectively.

$$Time\ violation = \frac{\sum_{i=1}^{Total\ number\ of\ workflows} \frac{Makespan_i + Arrival\ time\ of\ workflow_i - Deadline\ of\ workflow_i}{Deadline\ of\ workflow_i - Arrival\ time\ of\ workflow_i}}{Total\ number\ of\ workflows} \quad (6)$$

$$Count\ violation = \frac{Number\ of\ missed\ workflow\ deadlines\ during\ processing\ workflow}{Total\ number\ of\ workflows} \quad (7)$$

To assign a deadline to each workflow, we need to define the makespan for the shortest schedule *S*, which is gained by scheduling each workflow task on a distinct VM with the highest ranking while all data transfer time is regarded as zero. Therefore, the deadline of workflow can be calculated through Eq.8.

$$Deadline\ of\ workflow_i = \ Arrival\ time\ of\ workflow_i \ + \lambda * S \tag{8}$$

where λ is a variance factor of the workflow deadline.

b) Cost

Based on utilization of cloud resources in cloud, there are two types of cost, computation cost and communication cost. The total cost is the sum of them as defined through Eq.9 [28, 29].

$$C_{total} = C_{comp} + C_{comm} \tag{9}$$

where C_{comp} is the computation cost of the workflow from task t_i to task t_j that is defined by using server *l* through Eq. 10

$$C_{comp} = PP_l * (FT_{t_i} - ST_{t_j}) \tag{10}$$

where PP_l is the processing unit price of server *l*.

Moreover, the communication cost of the workflow from task t_i to task t_j is defined by using server *l* through Eq. 11

$$C_{comm} = \ CP_l * \mathrm{CT}(e_{ij}^l) \tag{11}$$

where CP_l is the communication unit price of server and CP_l= 0.1 $ per Hour. When two task $t_i$and t_j excecuted in a same VM (l), then CT (e_{ij}^l) =0;

Otherwise, it is equal to CT (e_{ij}^l) =$\frac{cv_{ij}}{B}$ in witch, B is the bandwidth between two VM (=20 MBps) and cv_{ij} is the data size (in MB) from task $t_i$to its successor taskt_j.

c) Reliability

The probability of running all the tasks on a workflow successfully is called reliability, which is a significant criterion for evaluating the efficiency of workflow scheduling problem. According to the critical pathway method, if the execution time of the tasks in the critical path is postponed, it will delay the completion of the whole workflow sample. Whereas the execution time of a task is delayed in an uncritical path, the execution time of the entire workflow will not be delayed. Hence, tasks in the critical path must be assigned to high-reliability VMs to avoid overdue missed tasks. Thus, loading time slack *j* from task $T_j$is defined as follows:

$$Slack_j = LS_j - ES_j = LF_j - EF_j \tag{12}$$

where ES_j ,LS_j ,EF_j ,LF_j are the fastest start time, the shortest start time, the fastest finish time, and the shortest completion time of T_j , respectively. In addition, slack j = 0 indicates that the task T_j lies in the critical path and slack> 0 means the task is not in the critical path. Reliability Rel_{jk} of the task T_j on the virtual machine VM_k is defined through Eq. 13:

$$\begin{cases} Rel_{jk} = \exp\left(-\lambda p_k * TM_{jk}\right) & if\ slack_j = 0; \\ Rel_{jk} = exp\left(-p_k * TM_{jk}\right) & otherwise \end{cases} \tag{13}$$

where TM_{jk} represents the runtime of the task T_j in the virtual machine VM_k ,$p_k$is the failure rate of VM_k, and the multiplication factor λ is a real number greater than 1 and its value depends on the problem (in this paper λ= 2). With respect to λ, the tasks in the critical path cannot be assigned to a low-reliability VM. Notice that assigning tasks in the critical path (slack j=0) to higher-reliability VMs (low $p_k$values) can reduce the failure rate of the workflow execution, so the probability of delay in completing the workflow execution is reduced. Eventually, the total reliability of the workflow is defined through Eq. 14 [30].

$$Rel_i = \prod_{j=1}^{m} Rel_{jk} \quad (14)$$

where m is the tasks' number of workflow.

d) Energy consumption

Energy consumption is one of the qualitative needs of cloud service providers. The energy model used in this paper is based on the introduced approach in [13] witch calculates the energy consumption of an activity t_i on a resource VM_j by increasing its duration of execution by the power consumption of the resource (v_{VM_j}) through Eq. 15.

$$E(t_i, VM_j) = ET_{t_i}^{VM_j} * v_{VM_j} \quad (15)$$

Also, the high-end servers consume energy between [250W, 500W] in fully loaded mode and 100W in idle mode. The workflow energy consumption is calculated through Eq. 16.

$$E\,(W) = \sum_{i=1}^{N} E(t_i, VM_j) \quad (16)$$

According to the definitions, the research problem can be described as follows:

"Finding scheduler *S* with the lowest values of *TEC*, *energy consumption*, *makespan* so that the highest reliability and minimum failure in scheduling occurs."

Minimize {TEC, Makespan, Energy consumption, Failure rate}.

3. Related works

Workflow scheduling consists of mapping workflow tasks to computational resources to be executed. Most workflow scheduling algorithms are provided to minimize the makespan [7] or the cost [31]. Other objectives are energy consumption [32,33], load balancing [34,35] and so on which have been less considered so far. In workflow scheduling, reliability, makespan and cost are related to the quality requirements of users, and energy consumption, resource utilization and load balancing are related to the provider performance [36]. Generally, workflow scheduling problems can be separated into three categories based on the number of objectives as follows:

1) Single-objective workflow scheduling: some studies have focused on the workflow scheduling problem with one objective such as minimizing makespan [37,38] or minimizing cost [31]. Kaur et al. [1] presented a workflow scheduling approach based on a Shuffled Frog Leaping Algorithm (ASFLA) with considering the execution cost optimization. However, in the ASFLA, the cost reducing makes an overhead of increased makespan and ASFLA only has better performance when minimizing the cost metric is the major goal. Ding et al. [39] proposed a Q-learning based task scheduling framework for energy-efficient cloud computing (QEEC) that has two phases. In the first phase a centralized task dispatcher is used to execute the M/M/S queueing model, by which the arriving user requests are assigned to each server in a cloud. In the second phase a Q-learning based scheduler uses a continuously-updating policy to assign tasks to virtual machines. Zhu et al. [35] introduced a genetic algorithm for load balancing in scheduling that considers CPU and memory balances for provider efficiency. Wang et al. [40] presented an energy aware scheduling model based on Map Reduce. It divided the tasks into "Map tasks" and "Reduce tasks" that are scheduled by a modified genetic algorithm to solve this model. The authors introduced a local search operator to improve convergence speed and searching ability of the algorithm. The experiments show that their proposed algorithm is efficient to reduce the energy consumption. Chen et al. [41] introduced a set-based discrete PSO to minimize cost and makespan and maximize reliability, but only one objective is considered each time.

2) Bi-objective or multi-objective workflow scheduling: some studies have focused on the workflow scheduling problem with bi or three objectives such as cost and makespan [11] , makespan and energy consumption [42,43] makespan, cost and energy [9], makespan, cost and reliability [44]. The straightest solution for a multi-objective optimization problem is transforming into a single objective optimization problem. However, define a suitable aggregation weight is difficult in this case. For example, Li et al. [45] presented a cost-conscious weight factor to select resources, and then proposed a bi-objective heuristic, derived from HEFT to minimize makespan and minimize cost. Jena [46] used a clonal selection algorithm (TSCSA) to reduce the energy consumption and makespan. Verma et al. [47] offered a Bi-Criteria Priority based PSO (BPSO) considering execution time and execution cost under the

deadline and budget constraints. In the BPSO, workflow tasks are firstly prioritized using the bottom level and then executed according to their priority, which is similar to what is defined in HEFT. Arabnejad et al. [48] suggested a workflow-scheduling algorithm named as Deadline-Budget Constrained Scheduling (DBCS) to find an appropriate answer with respect to budget and deadline constraints. The results indicate that the suggested algorithms reach lower cost and higher success rate compared to its counterparts. Mansouri et al. [49] proposed a hybrid task-scheduling algorithm named FMPSO that is based on Fuzzy system and modified particle swarm optimization technique to enhance load balancing and cloud throughput. Their proposed algorithm efficiently uses the resources, reduces makespan and increases efficiency and degree of imbalance. However, it does not consider the precedence of tasks and load balancing. Kaur et al. [44] introduced a workflow scheduling algorithm namely BAT based on the echolocation behavior of the virtual bats that minimizes cost and the execution time and maximizes the reliability metric. Some specifications of the VM such as processor speed, size of RAM, and failure rate to calculate the reliability of a VM are considered. Choudhary et al. [8] proposed a combination of a Gravitational Search Algorithm (GSA) and a Heterogeneous Earliest Finish Time (HEFT) for scheduling and tried minimize cost and makespan. Their proposed approach outperforms GSA and HEFT algorithms since the number of instructions is used to calculate the task execution time but this may not work precisely for the complex tasks. In addition, it has supposed that the bandwidth between the virtual machines is fixed and all the workflow tasks contain simple programming instructions. Zhou et al. [50] offered a Multi-Objective Evolutionary Algorithm (MOEA) for multi-objective scheduling in grid environment. Using MOEA approach, an optimal set of solutions close to the non-dominated solutions is generated. A multi-objective workflow-scheduling namely R-NSGA-II is presented by [51] to improve three conflicting objectives such as execution time, total cost, and reliability within a short period of time in cloud computing. Chen et al. [52] propose a heuristic strategy, which consists of proactive and reactive approaches, to schedule real-time multiple workflows with uncertain task execution time. In their strategies, nevertheless, the priority and decision parameters of each successor task which will be scheduled are throughout the original setting values during the scheduling. Liu et al. [27] proposed an onliNe multi-workflow Scheduling Framework, named NOSF that divides the scheduling process of workflows into three phases, and dynamically allocates VM resources for randomly arrived workflows to share the billing time of leased VMs as much as possible. The proposed deadline-aware heuristic algorithm can adaptively adjust the scheduling priorities and sub-deadlines of successor tasks ready for scheduling online to conquer the impacts of VM performance fluctuations. Gill et al. [53] proposed a particle swarm optimization based resource scheduling algorithm namely BULLET which is used to execute workloads effectively on available resources. At first, it analyzes workloads and does workload clustering. Next, it identifies the required resources. Then, based on the user's requirements, a component maps the user workloads to the proper resources. Finally, a scheduler schedules user workloads to the available running resources for guaranteeing near optimal satisfaction of the user's requirements. Their approach provides effective outcomes as compared to existing PSO based scheduling algorithms at different levels of cost, time and energy as shown in test cases. All of the mentioned studies have focused on the workflow-scheduling problem with bi or three objectives and those articles cannot handle workflow-scheduling problems with many objectives.

3) Many objective workflow scheduling: only a limited number of researches have addressed four or more objectives in workflow-scheduling problems [54]. Fard et al. [13] applied a list scheduling heuristic for workflow scheduling considering four-objective comprising makespan, reliability, energy consumption, and cost in distributed computing infrastructures. Ye et al. [54] proposed an improved knee point driven evolutionary algorithm in a four-objective workflow scheduling problem considering improve the makespan, the execution time, the reliability, and the execution cost of workflow, but it does not consider the energy consumption reduction. As mentioned above, several heuristics, and metaheuristic are proposed for single, bi, and multi-objective scheduling problems. Although heuristics algorithms are computationally effective, but they are not able to present global optimal solution for workflow scheduling [13, 27, 43]. The use of metaheuristic methods leads to improve performance of scheduling problems [1, 9]. In order to solve many objective optimization problems, algorithms such as NSGA-III [30], grid-based evolutionary algorithm (GrEA) [55], reference vector based evolutionary algorithm (RVEA) [54], and Many Objective Particle Swarm Optimization (MaOPSO) [12] have been presented so far. However, until now, the workflow-scheduling problem has not been implemented with these algorithms. In this paper, we apply improved MaOPSO algorithm to produce a set of non-dominated solutions for cloud workflow scheduling. The details of some related works in scheduling algorithms of cloud computing are tabulated in Table 3.

Table 3. Comparison of related works for workflow scheduling

Ref	Workload	Algorithm/ policy or strategy	Model	Type of	achievement	Weakness/limitations

				Algorithm		
[1]	Montage, Cyber Shake, LIGO workflows	Augmented Shuffled Frog Leaping Algorithm (ASFLA)	Single objective	Meta-heuristic	Optimizing the cost	1- Execution time increased due cost reduction. 2- ASFLA only improve performance.
[39]	Number of independent tasks	A Q-learning based task scheduling framework for energy-efficient cloud computing (QEEC)	Single objective	Heuristic	Optimizing the energy consumption	QEEC only improve energy consumption.
[9]	Montage, Cyber Shake, Epigenomics, LIGO, SIPHT workflows	Combination MOPSO algorithm and a list based heuristic (HPSO) in Cloud	Multi-objective	hybrid	Optimize cost, makespan and energy consumption	Reduce the efficiency of the algorithm with increasing the number of objectives
[48]	Montage, Cyber Shake, Epigenomics, LIGO and SIPHT workflows	Proportional deadline constrained (PDC) and deadline constrained Critical path (DCCP) algorithms	Bi-objective	Heuristic	more cost reduction and higher success rate	1-It is only effective in data intensive workflows. 2- They only work on a limited group of resources
[49]	Number of independent tasks	A hybrid task-scheduling algorithm named FMPSO that is based on Fuzzy system and modified particle swarm optimization technique	Multi-objective	Meta-Heuristics	Reduce makespan and increase the efficiency and degree of imbalance	It does not consider the precedence of tasks and load balancing
[8]	Montage, Cyber Shake, Epigenomics and SIPHT workflows	Combination of a Gravitational Search Algorithm (GSA) and a Heterogeneous Earliest Finish Time (HEFT) for scheduling in cloud	Bi-objective	hybrid	Optimize cost and makespan	1- This may not work precisely for the complex tasks. 2- It has supposed that the bandwidth between the virtual machines is fixed
[52]	Montage, CyberShak, SIPHT, LIGO	Uncertainty-aware online scheduling for real-time workflows in cloud service environment	Multi-objective	Heuristic	Optimize cost, deviation, resource utilization and fairness	It does not consider the reliability of VMs and energy efficiency
[27]	Montage, Cyber-Shak, Epigenomics, LIGO and SIPHT	Online multi workflow scheduling under uncertain task execution time in IaaS clouds	Multi-objective	Heuristic	Reducing the VM rental cost and the deadline violation probability, as well as improving the resource utilization efficiency	The proposed approach does not consider the joint optimization of the VM rental cost and energy efficiency

[53]	The workload is modeled as processing of images to convert from one format to another	A PSO based resource scheduling technique called BULLET for scheduling of workloads in cloud environment	Multi-objective	Meta-Heuristic	Optimize execution time, cost, energy, availability, resource utilization, latency and reliability	Their framework cannot identify relationship between workload (patterns) and the resource demands in the cloud.
[13]	Synthetic workflows	A generic multi-objective list scheduling(MOLS) in heterogeneous distributed computing infrastructures	Many objective	Heuristic	Optimize makespan, cost, average execution time and reliability	Results are optimized for energy minimization, which however, are not optimal for users.
Proposed approach	Montage, Cyber Shake, Epigenomic sworkflows	An improved many-objective particle swarm optimization algorithm for workflow scheduling in cloud	Many objective	Meta-Heuristic	Optimize four conflicting objectives, makespan, cost, energy consumption and reliability	The proposed approach does not consider data distribution in a multi cloud environment

4. Proposed approach

Scientific workflow scheduling is not only a many-objective optimization problem in nature but also is a NP-hard problem in cloud computing. Considering the problem's properties, we present the improved MaOPSO algorithm to produce a collection of the best possible scheduling solutions considering four conflicting objectives namely minimization of cost, makespan and energy consumption and maximization of reliability. In this section, firstly, we will state the basic background on general PSO, MOPSO and MaOPSO, and then we will describe the proposed algorithm of the workflow scheduling in cloud environment.

4.1. Background on general PSO, MOPSO and MaOPSO Optimizations

Particle Swarm Optimization (PSO) is a population based evolutionary computation technique, where a population is called a swarm. A swarm has *N* particles (solutions) and the searching space has D dimensional. Each particle has knowledge of its previous personal best solution and knows the global best solution found by the entire swarm. Each particle updates its own velocity and calculates its own new position through Eqs. (17) and (18) to determine the speed and direction of its fly, respectively.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1(t)[y_{i,j} - x_{i,j}(t)] + c_2 r_2(t)[\hat{y}_{i,j} - x_{i,j}(t)] \quad (17)$$

$$X_i(t+1) = X_i(t) + v_i(t+1) \quad (18)$$

where w is known as the inertia weight and is critical for the algorithm's convergence. $v_{i,j}$ is the recent velocity of the particle and v(t+1) represents new particle velocity. X(t) is the current position of the particle *i* at time *t* and X(t+1) is new position of the particle. c_1 and c_2 are cognitive and social acceleration coefficient, respectively. $r_1$and $r_2$are two random numbers with values between 0 and 1. $y_{i,j}$ stands for the best position visited by a particle (called pbest) and $\hat{y}_{i,j}$ indicates the global best found by the neighboring particles (called gbest).

In general PSO; firstly, the system is randomly initialized with the solutions and each solution is moved around in the solution space. Then, for each particle, a fitness value is calculated and *Pbest* and *Gbest* are updated. Next, the velocity and position of each particle to specify the speed and its movement direction is updated according to its fitness values. Finally, each particle is guided by its own new position and velocity to find a better result in the solution space. The process is repeated until the most favorable solution is found or the stopping condition is satisfied.

PSO has limitation of optimization as single objective, so Multi-Objective PSO (MOPSO) algorithms were introduced that are used to improve several conflicting objectives [9]. Approaches based on Pareto are most

appropriate for MOPSO, due to their capability to produce several solutions in less computation period. In MOPSO, an external archive (A_t) is added to the general PSO algorithm to store non-dominated solutions found by the algorithm during the search process until the iteration t and construct the non-dominated solutions. The velocity and position updates equations are same as equation (14) and (15) in the general PSO but all objectives are used to find Gbest and Pbest for each particle. The best detected position of the particle i since the start of the search procedure is called particle's cognitive leader. The social leader of the particle i (that is chosen from the external archive according to a predefined criterion such as diversity) is the best position detected by its neighborhood (i.e., the set of particles that it can interact with) since the start of the search procedure. After allocating a cognitive leader and a social leader for each particle, the velocity $v_{i,j}$ of each particle i in the dimension $j \in \{ 1 , 2 , \ldots , n \}$ at iteration $t + 1$ is updated. After updating the velocity, the position vector $\mathbf{x}_i$ of each particle is updated. The algorithm store previously generated non-dominated solutions by recording the Pbest found by a particle during the search process. The external archive is normally pruned if the number of non-dominated solutions surpasses a pre-defined threshold.

Due to the lack of effective studies and approaches on problems with more than three objectives, Figueiredo et al. [12] proposed a reference-based MOPSO for solving many objective problems called MaOPSO. It utilizes the idea of reference points in its fitness assignment function to impose the selection pressure required for the algorithm to converge into the non-dominated solutions. MaOPSO started by generating N particles randomly in the decision space by an even distribution to form the initial swarm S_0 . Then, the particles are evaluated by a fitness assignment function. Similar to other MOPSOs, MaOPSO has an external archive which is empty at first. Then in order to use during the selection process of the social leader aiming to differentiate the solutions in the external archive, a set of evenly distributed reference points is generated. Afterwards, the algorithm applies repeatedly a sequence of steps which contains mainly in: (1) Choose the cognitive and social leaders for the particles from the external archive and move the particles in the decision space; (2) Apply polynomial mutation to 15% of the particles in the swarm to improve the diversity and avoid the too early convergence; (3) Update the external archive using the new solutions visited by the particles so that the new archive just contains non-dominated solutions; (4) Prune the external archive when its maximum size excesses. The above steps are repeated until the termination condition is reached.

A detailed illustration of the general PSO, MOPSO and MaPSO algorithms is given in Fig. 2, which follows three general steps. First, initialize step to generate a set of particles on the swarm; second, evaluate step to find, update and select the social leader and the cognitive leader for each particle; third, update step to update the velocity and position in each iteration. This paper aims to improve the MaOPSO algorithm to more efficiently solve the many-objective workflow scheduling problem in cloud environment such as applies an appropriate balance between exploration and exploitation.

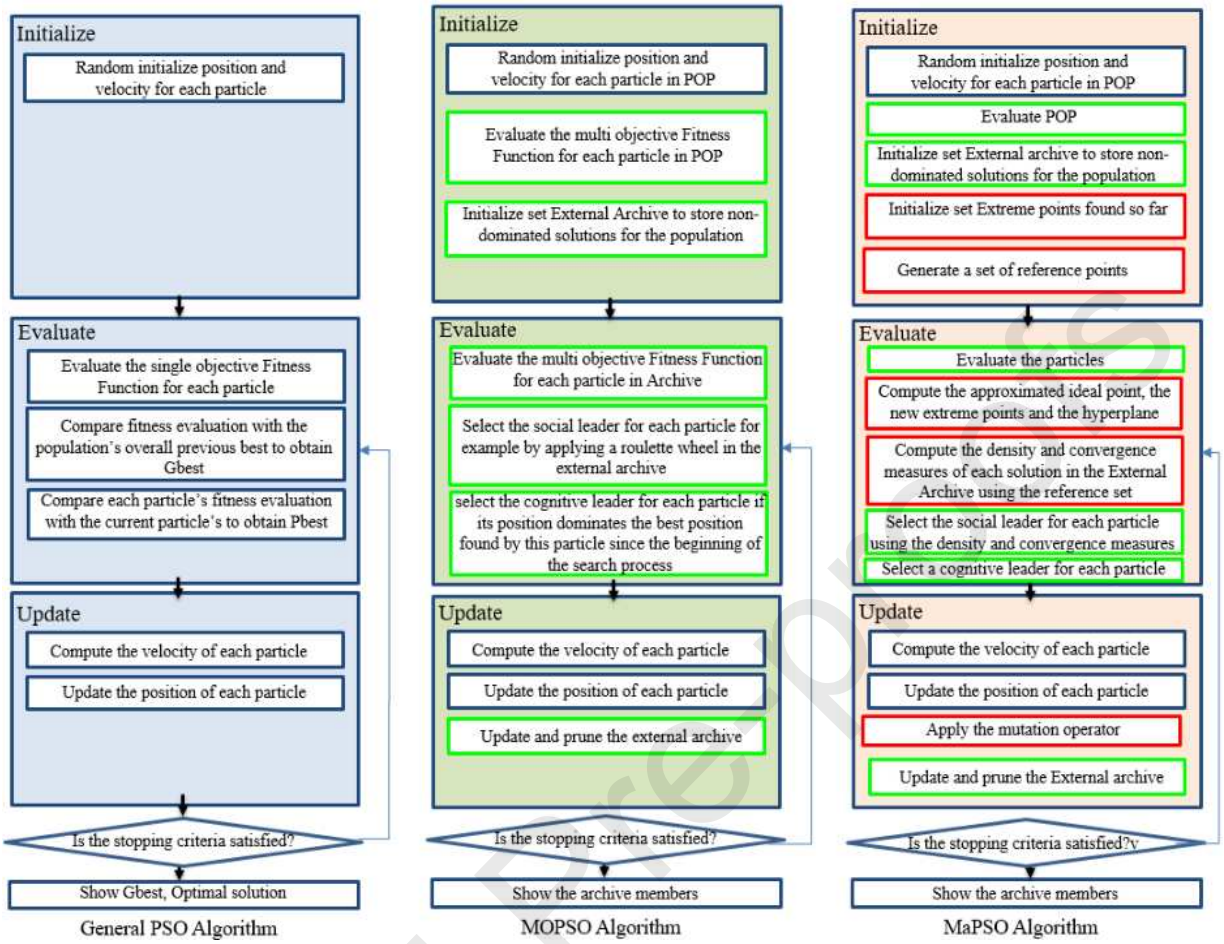


Fig. 2. A detailed illustration of the general PSO, MOPSO and MaOPSO algorithms

4.2. Description of the improved MaOPSO

In this section, we describe our approach detailing the pseudocode of proposed I_MaOPSO, and the differences with respect to MaOPSO (the algorithm which I_MaOPSO is based on).

4.2.1. Pseudocode and flowchart of the proposed algorithm

Algorithm 1 shows the pseudocode of I_MaOPSO. It begins by initializing the N mapping (particle) between workflow tasks and available resources (Line 1), which includes the position and velocity of the particles. Subsequently, each mapping is evaluated using Eqs. (4), (5), (10), and (12) (Line 2). Similar to other MOPSOs, I_MaOPSO also has an external archive (A_t) which is used to store non-dominated solutions that are generated by the algorithm during the search process and is initially empty (Line 4). In the following, a set of extreme points and a set of distributed reference points are produced that are used during the selection process of the leader to distinguish between solutions in the external archive (Lines 5 and 6). Nex, the primary loop of the algorithm runs until the termination condition is met. The external archive, the approximated ideal point, the new extreme points and the hyperplane are computed and updated (Lines 8-11) and the social and cognitive leaders are chooses from the external archive to move the particles in the decision-making space (Lines 12 and 13). Then a mutation operator is applied with a given probability (Line 16) to improve the diversity of solutions and prevent premature convergence and the resulting particles are evaluated (Line 17). The algorithm returns the external archive as the approximation set found (Line 20). If the external archive gets full then we use the crowding distance algorithm to discriminate the solutions and decide which one should be eliminated and which should remain in external archive.

Algorithm 1: Pseudocode of the proposed workflow scheduling algorithm I_MaOPSO
Input: Workflow tasks, Cloud resources
Output :Non dominated workflow schedule solutions

1. Initialize population of Size N swarm ();
2. Evaluate the swarm using the Fitness Metrics (cf. Eq. (4), (5), (10) and (12));
3. t = 0;
4. Initialize the external archive A_0;
5. Initialize the set of extreme points found so far Z_0;
6. Generate the set of reference points using NBI technique;
7. **while** t < $t_{\max}$ **do**
8. Update A_t from $A_{t-1} \cup S_{t-1}$ and Prune it if needed;
9. Calculate the approximated ideal point $z^{\min}$ from A_t;
10. Calculate the new extreme points Z_t from $Z_{t-1} \cup A_t$ *(using ASF function);*
11. Calculate the hyperplane from extreme points Z_t;
12. Calculate the density measure μ_a (using the Density operator) and the convergence measure ρ_a (using the Convergence operator) of each solution $a \in A_t$ using the reference set;
13. Select a social and a cognitive leader $\in A_t$ for each particle $i \in S_t$ using the density and convergence measures;
14. Compute Velocity ();
15. Update Position ();
16. Mutation;
17. Evaluation;
18. t ++
19. **end while**
20. Return $A_{t_{\max}}$ containing best non-dominated schedules.

Based on the above descriptions, the algorithm's flowchart is drown and showed in Fig. 3.

4.2.2. Differences with respect to MaOPSO

Given that MaOPSO is the basis of our I_MaOPSO, and it is necessary to distinguish between them as follows:

1) In the approach presented in [12], the initial swarm S0 is randomly generated using a uniform distribution in decision space. However, we propose four greedy heuristic methods for generating four of N initial solution, which consider the objective functions in order to create the initial solutions. In this way, the possibility of generating a 'more explored' initial swarm increases considerably. It should be noted that other $N - 4$ members of the initial swarm are randomly generated. These methods are as follows:

- **Heuristic 1:** given the similarity of the workflow-scheduling problem considering the makespan as the objective function to the parallel machine sequencing and scheduling problem (Q_m/prec/c_{max}). Since the LPT (the Longest Processing Time first) method for this problem leads to good and near optimal solutions, this heuristic is applied to create one of the solutions of the initial swarm. In this approach the tasks are sequenced based on the descending order of their processing time on the processors and each task is assigned to the machine with the minimum total execution time.
- **Heuristic 2:** in order to minimize the total cost objective, at any given moment, the task and machine with the minimum total computation and communication costs are selected for allocation.
- **Heuristic 3:** in this method, the tasks are assigned to the most reliable resource. This assignment leads to a solution with the maximum reliability and the minimum failure rate respectively.
- **Heuristic 4:** energy consumption can be optimally solved by assigning all tasks onto the resource with the least energy consumption with respect to its computational speed. Therefore, in this approach, at any given moment the task is assigned to the machine with the minimum energy consumption.

2) In the MaOPSO, according to the analysis presented by Clerc and Kennedy [56], aiming to control the particle's velocity, a constriction coefficient is introduced as $\chi = \frac{2}{\emptyset - 2 + \sqrt{\emptyset^2 - 4\emptyset}}$ in which $\emptyset = \emptyset_1 + \emptyset_2 > 4$ and$\emptyset_1, \emptyset_2 > 0$.

Then the acceleration coefficients and *w* coefficient are calculated as $w = \emptyset, c_1 = \chi \times \emptyset_1, c_2 = \chi \times \emptyset_2$. Among all possible values for $\emptyset_1$ and $\emptyset_2$ the optimal values are $\emptyset_1 = \emptyset_2 = 2.05$. Based on this value, the constriction coefficient is equal to $\chi =$ 0.7298, the acceleration coefficients c_1 and c_2 are set to 1.49618 and the *w* coefficient is set to 0.7298. However, in order to strike an appropriate balance between exploration and exploitation, in this article in addition to this approach another efficient approach is used to set and change the
of these coefficients. In this paper, the coefficients c_1 and c_2 are set to the fix value 2 and the coefficient *w* is set to 1 in the first iteration. In order to reduce the exploration rate over time, *w* is multiplied by a constant coefficient (=0.9 in this research) in each iteration. For example, in the first iteration $w = 1$, in the second iteration $w = 0.9$, in the *n*-th iteration $w = (0.9)^{n-1}$, and so on. In order to compute the velocity of a specific particle, one of these two approaches are selected with the same probability of selection (= 0.5) with fitness proportionate selection method. This means that before calculating the velocity of a particle, a random value is created in the interval [0, 1]. If this random value is less than 0.5, the first approach and otherwise the second one is used to calculate the velocity of the mentioned particle.

To choose the social leader for the sub-swarm $\overline{S}_2$, the roulette wheel selection process is used instead of the tournament selection. The proposed roulette wheel selection method can be described as follows.

Let us consider there are $|A_t|$ solutions in the external archive, each characterized by its density measure ($\mu \geq 0$) and its convergence measure ($\rho \geq 0$). The selection probability of the *i*-th solution in A_t is specified by Eq. (19):

$$prob_i = \frac{1}{(\mu_i + \rho_i)} \Big/ \sum_{j=1}^{|A_t|} \frac{1}{(\mu_j + \rho_j)} \tag{19}$$

where μ_i and ρ_i are the density and convergence measures of solution *i* and $|A_t|$ is the size of the external archive. It is necessary to mention that the sum of the probability of selecting all solutions in the archive equals to 1 ($\sum_{i=1}^{|A_t|} prob_i = 1$).

Suppose a roulette wheel with the size of sectors, which is proportional with $\frac{1}{(\mu_i + \rho_i)}$ $(i = 1, 2, ..., |A_t|)$. Choosing a point on the wheel randomly and locating the corresponding sector are equal to an individual selection. Roulette wheel selection makes a line segment of length $\sum_{j=1}^{|A_t|} \frac{1}{(\mu_j + \rho_j)}$ out of consecutive sectors of length $\frac{1}{(\mu_i + \rho_i)}$ $(i = 1, 2, ..., |A_t|)$, creates a random number R such that $0 < R < \sum_{j=1}^{|A_t|} \frac{1}{(\mu_j + \rho_j)}$, and then specify the corresponding sector, and choosing the respective solution. Fig. 4 shows an instance for choosing a solution as a social leader based on the density and convergence measures with roulette wheel.

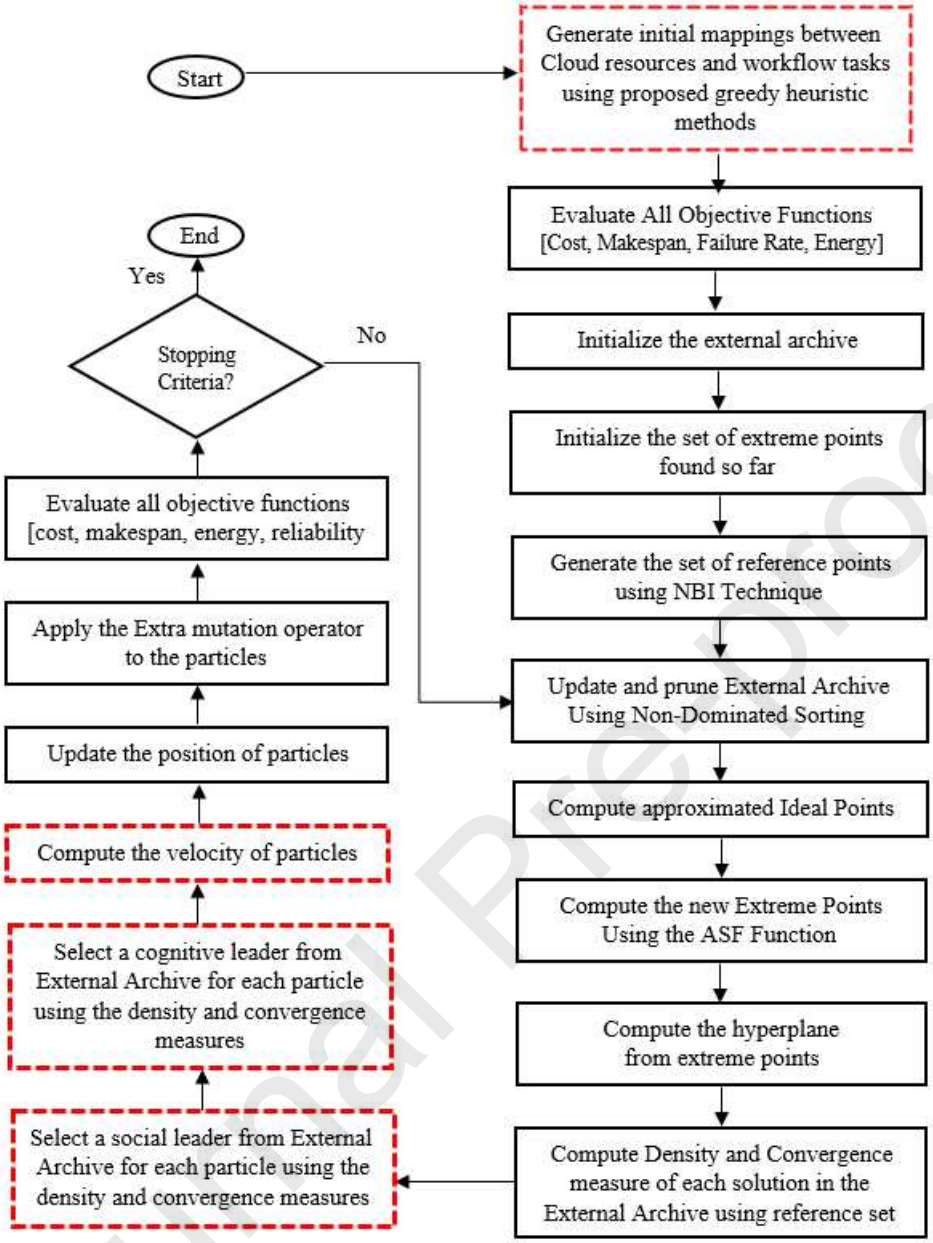


Fig. 3. Flowchart of the proposed algorithm (I_MaOPSO), which shows the innovations added to MaOPSO algorithm under the dashed rectangles.

3) Applying a roulette wheel selection procedure provides some benefits as:

a) This approach guarantees that the probability of selecting a weaker solution (i.e., a solutions with the bigger sum of the density and convergence measures) as social leader be smaller than the probability of selecting a better solution.
b) While candidate solutions with a smaller sum of the density and convergence measures will be more likely to be selected, because their selection probability is less than 1 (or 100%), there is still a chance that they may be not selected.
c) Contrary to the less developed selection algorithm (i.e., the truncation selection used in MaOPSO) which will choose a fixed percentage of the best candidates, roulette wheel selection always gives a chance to all of solutions in the archive to be selected.

Therefore, even if the selection probability of weak solutions is low, it is not zero that means it is still probable they will select. This is a remarkable point, since in some cases even weak solutions may have characteristics that can be useful as a leader; and there is still a chance that they will be selected.

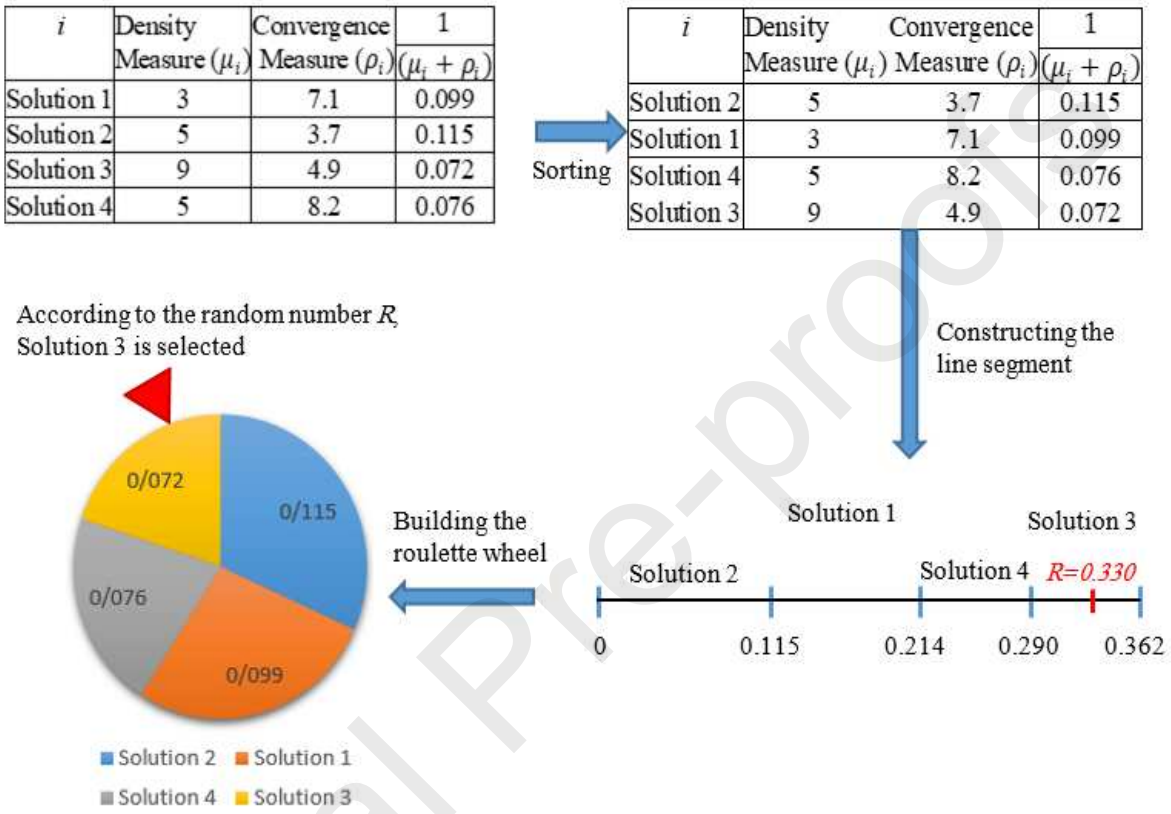


Fig. 4. Social leader selection by a roulette wheel based on the density and convergence measures of the solutions in the External Archive.

4) In this paper, the same as the MaOPSO algorithm, the Pareto-optimal front is adaptively discretizes by creating and adopting an entire normalized hyperplane. This hyperplane is created based on a widely distributed set of points (e.g., reference and extreme points) which ensure preserving the diversity of Pareto-optimal front during the search process. In fact, the keeping diversity between particle swarm is done by providing and updating a number of well-distributed reference and extreme points. In the MaOPSO algorithm during selection a cognitive leader for a solution, if it is not possible to compare solutions with each other, the Euclidean distances of the current solution and the cognitive solution to the ideal point are calculated, and the solution with the lowest distance is chosen as the new cognitive leader. It should be noted that the ideal point is a point in which the amount of each objective equals to the desired target value of that objective. The disadvantage of choosing the closest solution to the ideal point as the cognitive leader is leading the algorithm to premature convergence. In order to resolve this deficiency, in this work in the condition of incomparable solutions, the solution with the lowest Euclidean distance to the hyperplane is chosen as the new cognitive leader. The main advantage of this choice is that during the search process and creating the population members, a balance between the diversification and intensification capabilities of the I_MaOPSO algorithm is established. In intensification the regions around Pareto-optimal front are explored more thoroughly in the hope to find better solutions, while in diversification non-explored regions must be visited to ensure that all regions of the search space are evenly explored and that the search is not confined to only a reduced number of regions (e.g., around the ideal point).

4.3. Applying the proposed I_MaOPSO for the workflow scheduling

In order to implement the proposed I-MaOPSO algorithm for the workflow scheduling two main inputs should be determined: list of user requests (workflows), and 2) list of VMs as VM = {VM_1, VM_2, …, VM_m. Each users request consists of 1,…, n workflow instances , where for example $User\ request_1$ may have n workflow instances { WI_1 ,WI_2,..., WI_n}, then each workflow instance may have different workflow tasks WI_1={ $WT_{1,1}$, $WT_{1,2....}$, $WT_{1,m}$}, WI_2={ $WT_{2,1}$, $WT_{2,2....}$, $WT_{2,m}$}, and WI_n={ $WT_{n,1}$, $WT_{n,2....}$, $WT_{n,m}$}.
The initial population of the proposed approach is generated based on the mapping between tasks and VMs, where represents *particles* or the potential solutions of the problem domain. The set of particle consists of P= P_1, P_2, ..., P_n. Each *particle* is encoded with three variables *SEQ*, *MAP,* and *VEL* (Fig.5) as follows:

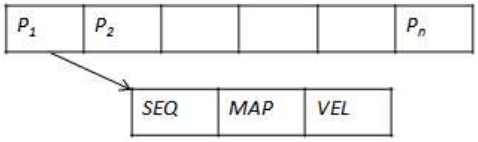


Fig.5. Structure of a particle

SEQ denote the scheduling sequence of all workflow tasks. It contains workflow task numbers and identifies the sequence of execution of workflow tasks in the VM. An example of *SEQ* matrix for workflow tasks given in the Fig.1 is *SEQ*= [WT_{11}, WT_{12}, WT_{15}, WT_{13}, WT_{14}, WT_{16}, WT_{17}, WT_{18}, WT_{19}].
MAP determine the mapping of workflow tasks to VMs. In addition, the *MAP* can be expressed as follows:

$$Map=\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ p_{n1} & p_{n1} & \cdots & p_{nm} \end{bmatrix}$$

where the possibility of assignment of workflow task WT_i to the VM_j is denoted as p_{ij} uniformly distributed in the interval [0,1] which is defined as

$$\sum_{j=1}^{m} p_{ij} = 1 \quad , p_{ij} \in [0,1] \qquad i\epsilon\{1,2,\ldots,n\},\ j\epsilon\{1,2,\ldots,m\} \qquad (20)$$

where Eq.1 presents a sum of all probability values of allocation $user\ request_i$ that the value should not be greater than 1. If probable value of workflow task allocation is the highest in the row, we will map the workflow task to that VM. An example for *map* matrix with four VMs and workflow tasks given in the Fig.1 is as follows:

$$Map=\begin{array}{c} \\ WT_{11} \\ WT_{12} \\ WT_{13} \\ WT_{14} \\ WT_{15} \\ WT_{16} \\ WT_{17} \\ WT_{18} \\ WT_{19} \end{array}\begin{array}{c} \begin{bmatrix} VM1 & VM2 & VM3 & VM4 \end{bmatrix} \\ \begin{bmatrix} 0.13 & 0.25 & \mathbf{0.45} & 0.17 \\ 0.23 & 0.19 & \mathbf{0.33} & 0.25 \\ 0.31 & 0.05 & 0.30 & \mathbf{0.34} \\ 0.27 & \mathbf{0.28} & 0.24 & 0.21 \\ \mathbf{0.45} & 0.11 & 0.42 & 0.02 \\ \mathbf{0.39} & 0.30 & 0.04 & 0.27 \\ 0.26 & \mathbf{0.34} & 0.24 & 0.17 \\ \mathbf{0.64} & 0.14 & 0.11 & 0.11 \\ 0.30 & 0.21 & 0.10 & \mathbf{0.38} \end{bmatrix} \end{array}$$

VEL is the velocity of each particle uniformly distributed in the interval [-1,1].

In this paper, first, based on the *seq* and *map* value, the Position matrix for a particle in the proposed approach for scientific workflow scheduling in Clouds is as follows:

$$\begin{bmatrix} VM1 & VM2 & VM3 & VM4 \end{bmatrix}$$

$$Position = \begin{matrix} WT_{11} \\ WT_{12} \\ WT_{15} \\ WT_{13} \\ WT_{14} \\ WT_{16} \\ WT_{17} \\ WT_{18} \\ WT_{19} \end{matrix} \begin{bmatrix} 0.13 & 0.25 & \mathbf{0.45} & 0.17 \\ 0.23 & 0.19 & \mathbf{0.33} & 0.25 \\ \mathbf{0.45} & 0.11 & 0.42 & 0.02 \\ 0.31 & 0.05 & 0.30 & \mathbf{0.34} \\ 0.27 & \mathbf{0.28} & 0.24 & 0.21 \\ \mathbf{0.39} & 0.30 & 0.04 & 0.27 \\ 0.26 & \mathbf{0.34} & 0.24 & 0.17 \\ \mathbf{0.64} & 0.14 & 0.11 & 0.11 \\ 0.30 & 0.21 & 0.10 & \mathbf{0.38} \end{bmatrix}$$

This position matrix is decoded as shown in Table. 4.

Table 4. The decoded Position matrix as an example for a particle in the proposed approach for scientific workflow scheduling

Workflow Task	WT_{11}	WT_{12}	WT_{15}	WT_{13}	WT_{14}	WT_{16}	WT_{17}	WT_{18}	WT_{19}
Virtual machine	$VM3$	$VM3$	$VM1$	$VM4$	$VM2$	$VM1$	$VM2$	$VM1$	$VM4$

Next, the velocity of each particle is initialized by randomly generating in the range of [-1, 1]. It is necessary to explain that after updating the position of a particle, if the amount of p_{ij} for a particular i and j becomes less than zero, it replaced by $-p_{ij}/4$ and if it becomes more than 1, it replaced by $p_{ij}/4$.

The aim of this study is to optimize many objective scheduling of scientific workflows in a cloud computing environment based on the proposed I_MaOPSO algorithm to obtain a uniformly distributed solution set with better convergence toward the non-dominated solutions (Pareto front) in terms of reduced makespan, minimized cost, efficient energy consumption of the VMs and maximized reliability. Finally, we calculate the squared Euclidean distance (SED) [57] for each output solution i as follows that estimates how far our current solution is from the summation of true Pareto points of a problem. Then we choose the output solution that has the least amount of the SED that is convergent toward the true Pareto front.

Therefore for each output solution $i=1, 2, \ldots, n$:

$$SED_i = \sum_{j=1}^{Total\ number\ of\ Pareto\ points} [(TEC_i - TEC_j^{Pareto})^2 + (Energy\ consumption_i - Energy\ consumption_j^{Pareto}) \quad (21)$$

Final output solution k is selected with the lowest value of SED_i.

5. Performance evaluation

The proposed approach is compared with three other leading algorithms called MaOPSO and LEAF, in the context of solving many-objective optimization problems and an Evolutionary Multi-objective Scheduling for Cloud (EMS-C) algorithm to solve the workflow scheduling problem on an Infrastructure as a Service (IaaS) platform.

a) MaOPSO

Many-Objective particle swarm Optimization (MaOPSO) is a many-objective optimization algorithm based on particle swarm and Pareto dominance which utilizes the idea of reference points uniformly distributed in its fitness assignment method for imposing the selection pressure required for the algorithm to converge toward the non-dominated solutions [12]. MaOPSO started by generating N particles randomly in the decision space using a uniform distribution to form the initial swarm S_0. Then, the particles are evaluated using a fitness assignment method. MaOPSO also has an external archive that is empty at first. Given that social leader selection process is used to differentiate between solutions within the external archive a set of well-distributed reference points is generated to be used during this process. Then, The algorithm applies iteratively a series of steps that involves: (1) Select the cognitive and social leaders for the particles from the external archive (2) Apply polynomial mutation to 15% of the particle swarm in (3)

Update the external archive using the new solutions visited by the particles. (4) Prune the external archive when its maximum size excesses. The above steps are repeated until the termination condition is reached.

b) LEAF

Another algorithm is [17] and it

b) EMS-C

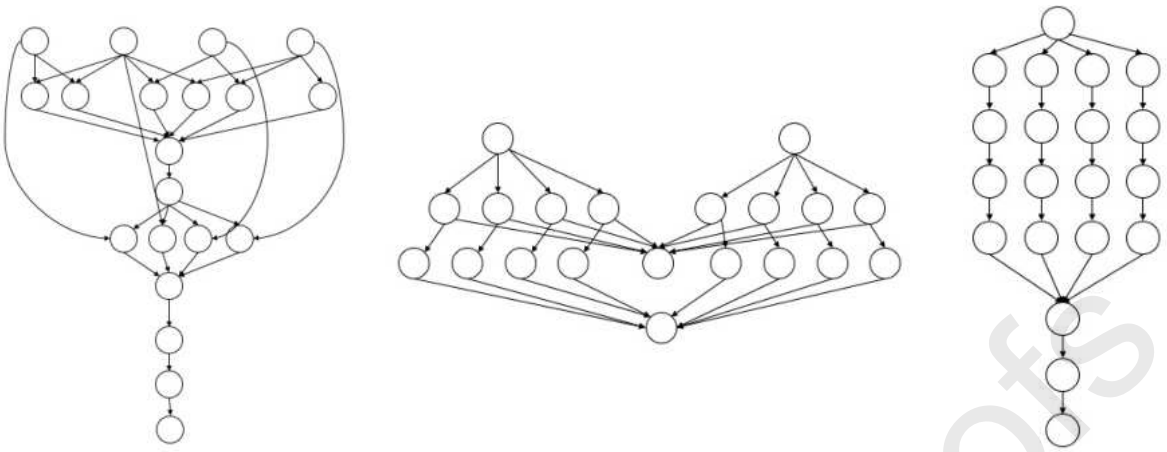
EMS-C algorithm [58] is under the multi objective NSGA-II framework and introduces a set of new genetic operators, the evaluation function and the population initialization scheme to solve the multi-objective Cloud scheduling problem, which minimizes both makespan and cost simultaneously, but it does not consider minimizing the energy consumption and maximizing reliability. The outcome of the algorithm is the final population with the results in both decision and objective spaces.

5.1. The benchmark workflows

To evaluate the proposed workflow scheduling algorithm, three scientific workflows with names Montage, Cybershake, and Epigenomics have been used. The Montage workflow is used in astronomy, the Cybershake workflow is used to describe the earthquake hazards in a region, and Epigenomics is used in biology [59] The Montage is known as an intensive data-processing workflow, which takes a lot of time to transmit data, but does not require complex calculations, and it only increases data transmission costs. Cybershake and Epigenomics need more computing resources. By choosing these three types of workflows, the efficiency of the proposed approach can be measured both in terms of processing data transfer. For each type of workflow, three kinds of DAG with dissimilar number of tasks and edges are presented. For example, the type of Cybershake workflow respectively has three kinds with 25, 50, 100 and 1000 tasks. In this article, the test case is workflow with maximum number of tasks. The DAG specifications of these workflows are given in Table 5, which shows the experimental results under the dashed rectangles. The sample structures of different scientific workflows are given in Fig. 6.

Table 5. Characteristics of the benchmark workflows [60]

Workflow	Number of Nodes	Number of Edges	Average data size (MB)
Montage_25	25	95	3.43
Montage_50	50	206	3.36
Montage_100	100	433	3.23
Montage_1000	1000	4485	3.21
CyberShake_30	30	112	747.48
CyberShake_50	50	188	864.74
CyberShake_100	100	380	849.60
CyberShake_1000	1000	3988	102.29
Epigenomics_24	24	75	116.20
Epigenomics_46	46	148	104.81
Epigenomics_100	100	322	395.10
Epigenomics_997	997	3228	388.59



a) Montage Workflow b) Cybershake Workflow c) Epigenomics Workflow

Fig. 6. Structure and complexity of scientific workflows [7]

5.2. Simulation Setup

The speed of the processor and other VM specifications, such as cost model, memory, failure rate, bandwidth, etc. are implemented based on the Amazon's proposed model and in accordance with Table 6. The average bandwidth of all sources is the same as 20 Mbps, and the VM's processing power is set to between 1000 and 10,000 millions of instruction per second (MIPS) inspired by the reference [9]. Also, the cost and failure rates of VMs are proportional to their processing power. To simulate, the cloud environment is assumed to include a service provider that delivers seven different computing resources with different processing speeds and different prices. In order to simulate the cloud environment, the WorkflowSim software, an upgraded version of CloudSim, has been used. One of the most important changes in this simulator is that it can read DAX files generated from workflow structures and provide required parameters such as runtime, file size, output file size, and tasks dependency [47].

Table 6: Recommended Amazon EC2 virtual machines (From https://aws.amazon.com/ec2/instance-types/)

Energy consumption v_{VM_j} (Watt per Hour)	RHEL Cost(pp_l)	Failure rate (p_k)	Bandwidth(Mbps)	Memory (GiB)	core	CPU(mips)	Machine name
250	$0.16 per Hour	0.1	20	8	2	1000	m4.large
300	$0.26 per Hour	0.07	20	16	4	2000	m4.xlarge
350	$0.53 per Hour	0.03	20	32	8	4000	m4.2xlarge
400	$0.93 per Hour	0.02	20	64	16	6000	m4.4xlarge
450	$2.13 per Hour	0.008	20	160	40	8000	m4.10xlarge
500	$3.33 per Hour	0.006	20	256	64	10000	m4.16xlarge

Given that the algorithms used are all implemented in re-simulation, the initial parameters of each of them must be properly set. The estimated values for each algorithm are considered with regard to their best execution. Table 7shows the initial parameters of the algorithms.

Table 7: Algorithm parameters

Algorithm	algorithm Parameter
Common parameters	Population size = 100 , iteration = 250 , Repeat number of simulations = 30 Total number of VMs=50 in expermint1 Total number of VMs from 16 to 128 in expermint2
LEAF	Mutation rate =1 , Crossover rate =0.9
MaOPSO	c1=1.5→2.5,c2=1.5→2.5 r1=0 →1,r2 =0→1,w = 0.1
EMS-C	Mutation rate = 0.5
Proposed approach	c1=1.5 → 2 c2=1.5→2 ,w=0.5 → 0.5 polynomial mutation rate = 0.3

The replication condition of the algorithms is 250 repetitions and each simulation implementation provides 30 optimal solutions hence in order to get more precise results, the average values as the final results are presented in the form of tables and plots.

5.3. Evaluation Indicators

In order to evaluate multi-objective optimization problems, there are various indicators such as GD, IGD, HyperVolume and etc. Each of them is used to measure some of the performance criteria of the algorithm such as density. In this study, the HV criterion was used to evaluate the simulation results [61].

HyperVolume (HV): The HV criterion computes the volume of target space surrounded by the set of solutions generated by the algorithm and a reference point and is defined as follows:

$$HV(P) = \mu\left(\cup_{z_a \in p} \{ z \mid z_a < z < z_{ref} \} \right) \quad (22)$$

Where μ is the Lebesgue criterion, P is the set of approximate solutions and z_{ref} is the reference point. Using the HV criterion, both the density and the spread of the algorithm's solutions can be measured, maximum HV value is the optimal and expected. To determine the HV, the reference point is necessary. In this research, the reference point for calculating HV equals the worst value of each target among the values obtained from it. Fig. 7 shows how to calculate the HV.

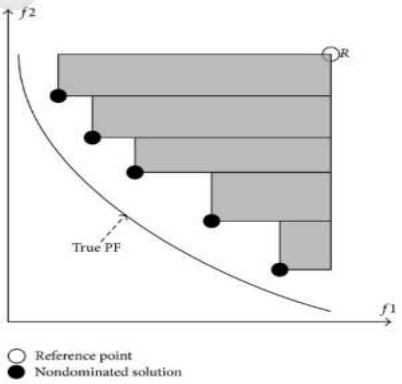


Fig. 7. HV criterion for 2D [62]

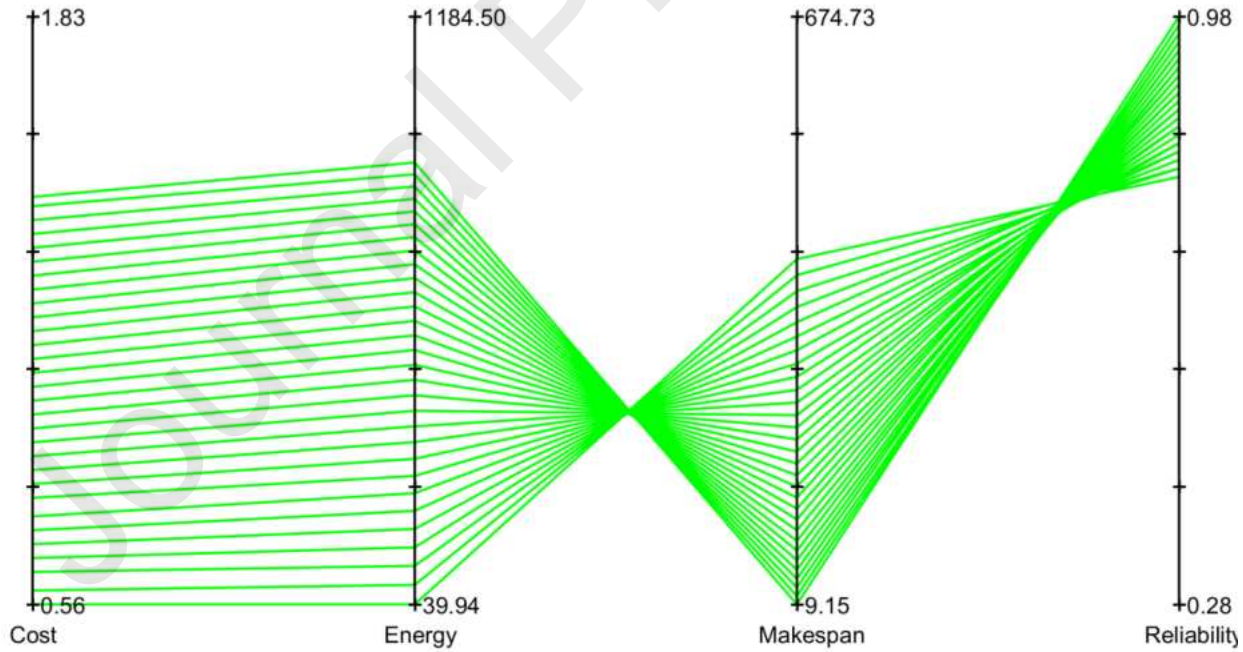
5.4. Evaluation resultsExperiment 1: Evaluation based on makespan, cost, reliability and energy consumption with different workflow instances and constant VM count

Implementation and evaluation of scheduling algorithms, energy, cost, reliability, and makespan are considered as targets. In order to facilitate displaying and analyzing the results of scheduling, parallel coordinates plots are used,

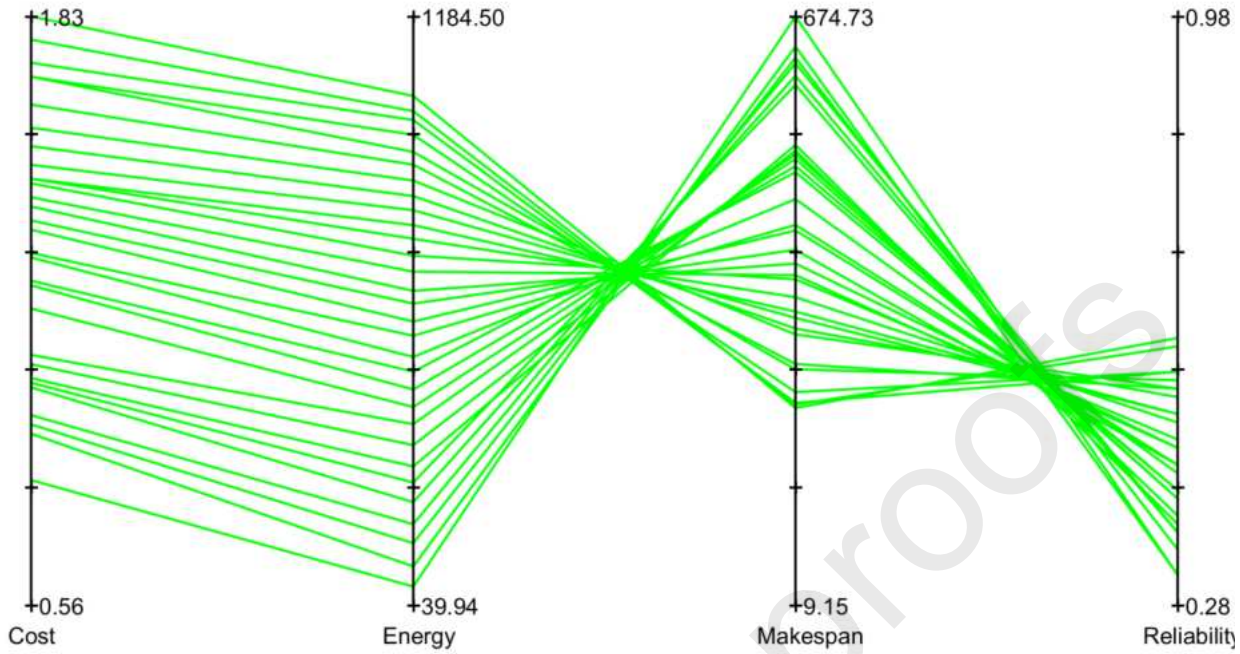
and four targets are presented at the same time. In the Figs. 8, 10 and 12, the horizontal axis shows different objectives and the vertical axis shows the value of each objective function. In order to calculate Pareto values, algorithms were repeated 250 times for each simulation and finally, each simulation implementation provides 30 optimal solutions (equals to archive size) and the maximum and minimum values of the targets were extracted. We use a collective frequency method to provide non-dominated solutions in any figure. A cumulative frequency distribution is the sum of the class and all classes below it in a frequency distribution. That entire means is you are adding up a value and all of the values that came before it.

a) Montage workflow

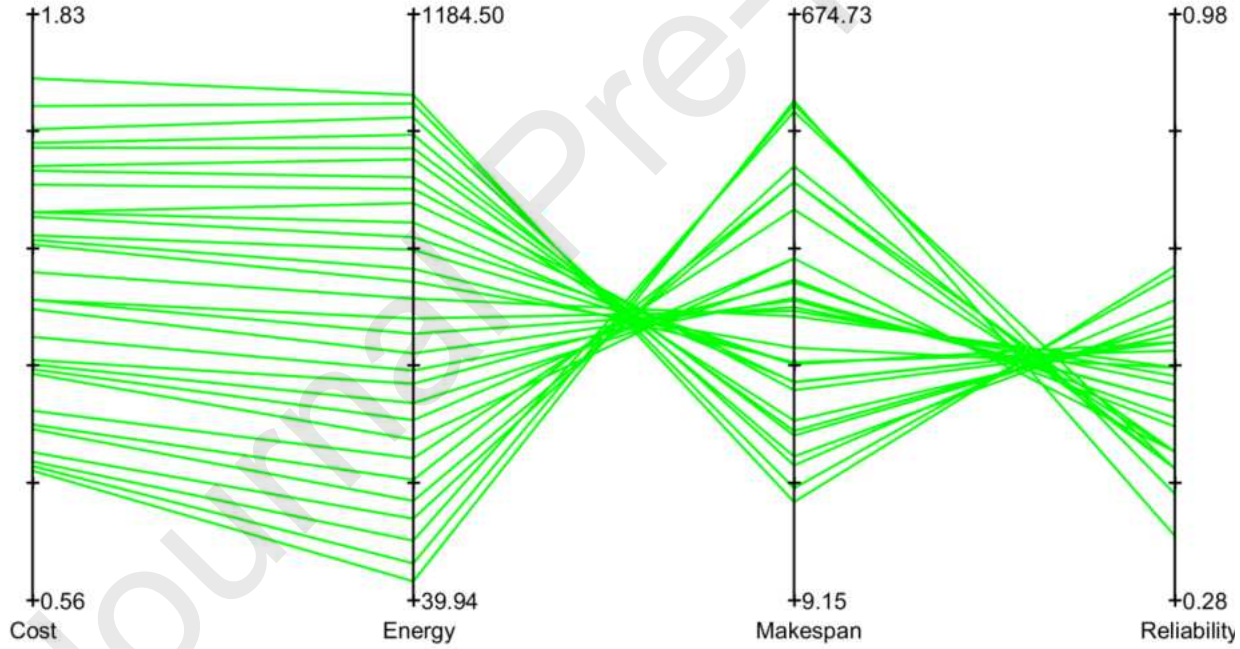
Figure 8 shows the comparison between the results for running 30 times of the LEAF, MaOPSO, EMS-C and proposed approach for energy, cost, makespan and reliability. The results of this figure indicate that the solutions of the proposed approach are closer to the non-dominated solutions and more uniform distribution in the target space in Montage workflow compared to other workflows. In the MaOPSO algorithm, the initial swarm S0 is randomly generated using a uniform distribution in decision space. However, the proposed approach applies four greedy heuristic methods for generating four of *N* initial solution, which consider the objective functions in order to create the initial solutions. In the proposed approach, the possibility of generating a 'more explored' initial swarm increases considerably. Compared to the LEAF method, the proposed approach applies an efficient approach to compute the velocity of particles in order to achieve an appropriate balance between exploration and exploitation.
According to Fig. 9, the HV criterion of the proposed scheduling algorithm has improved up to 71%, 182%, and 262% respectively, compared to LEAF, MaOPSO, and EMS-C algorithms. In the MaOPSO, to control the particle's velocity, a constriction coefficient is introduced. In the proposed approach in addition to this approach, another efficient approach is used to set and change the amounts of these coefficients in order to strike an appropriate balance between exploration and exploitation.



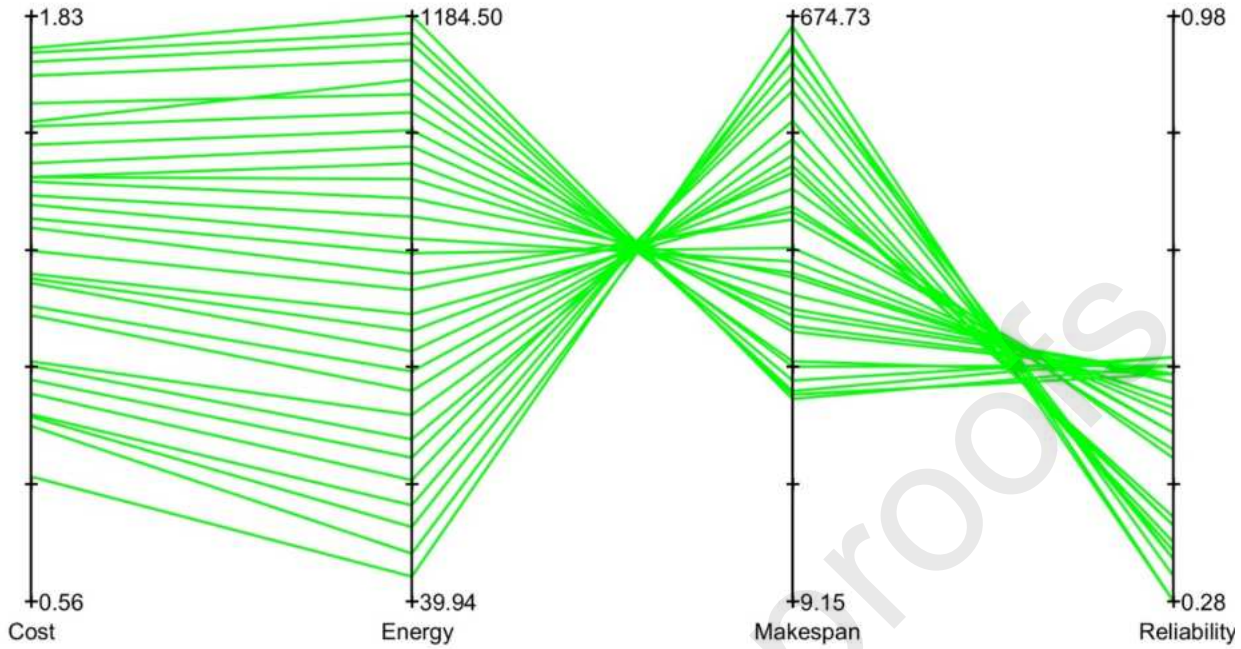
(a) Non-dominated solutions



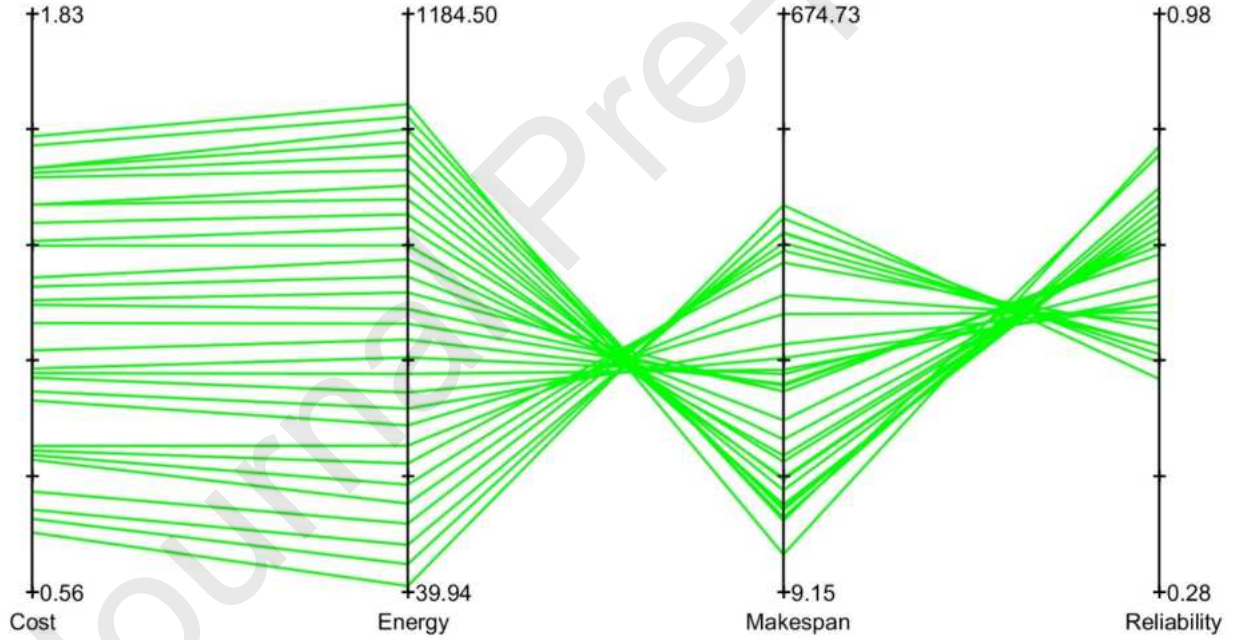
(b) MaOPSO



(c) LEAF



(d) EMS-C



(e) Proposed approach

Fig.8. Parallel coordinates plot of non-dominated solutions obtained from different approaches of Montage workflow scheduling for 4-objective

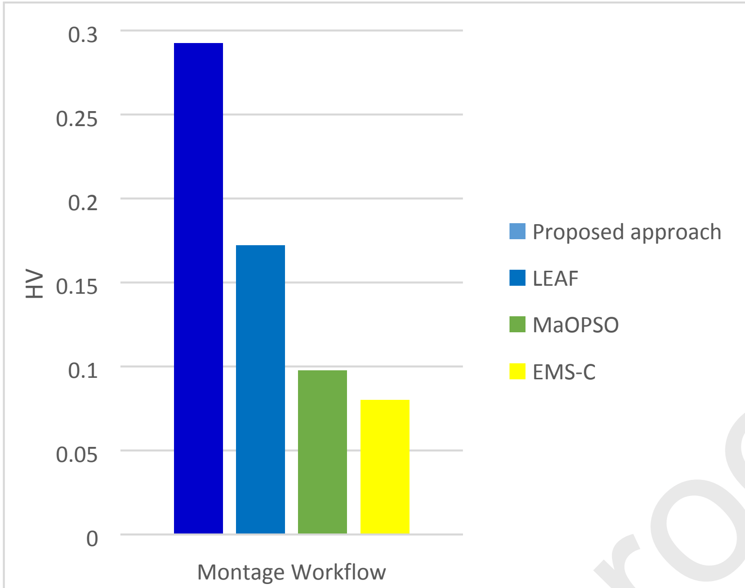
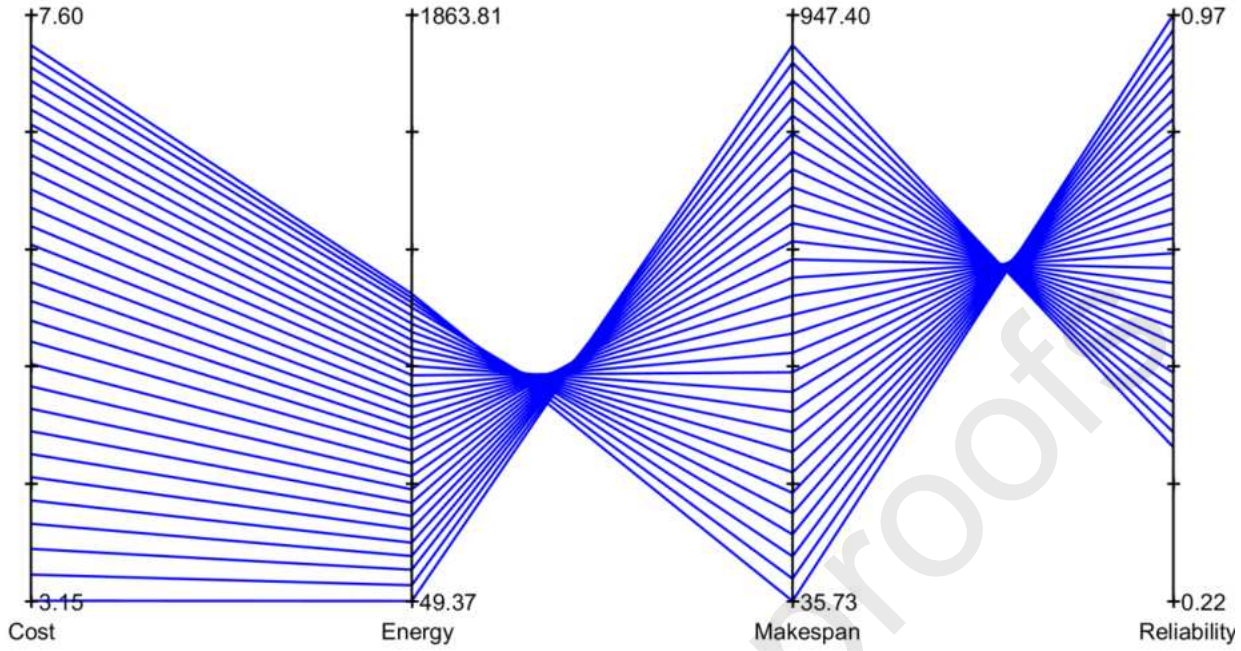


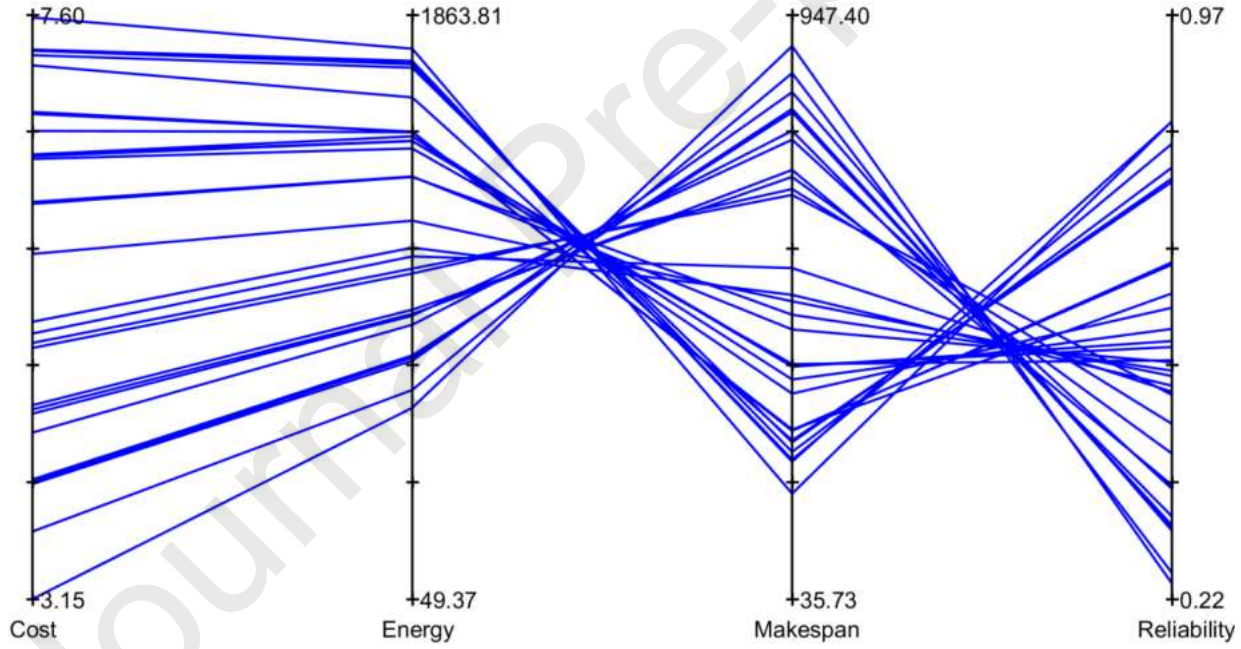
Fig. 9. Comparison of the HV criterion for the Montage workflow

b) Cybershake workflow

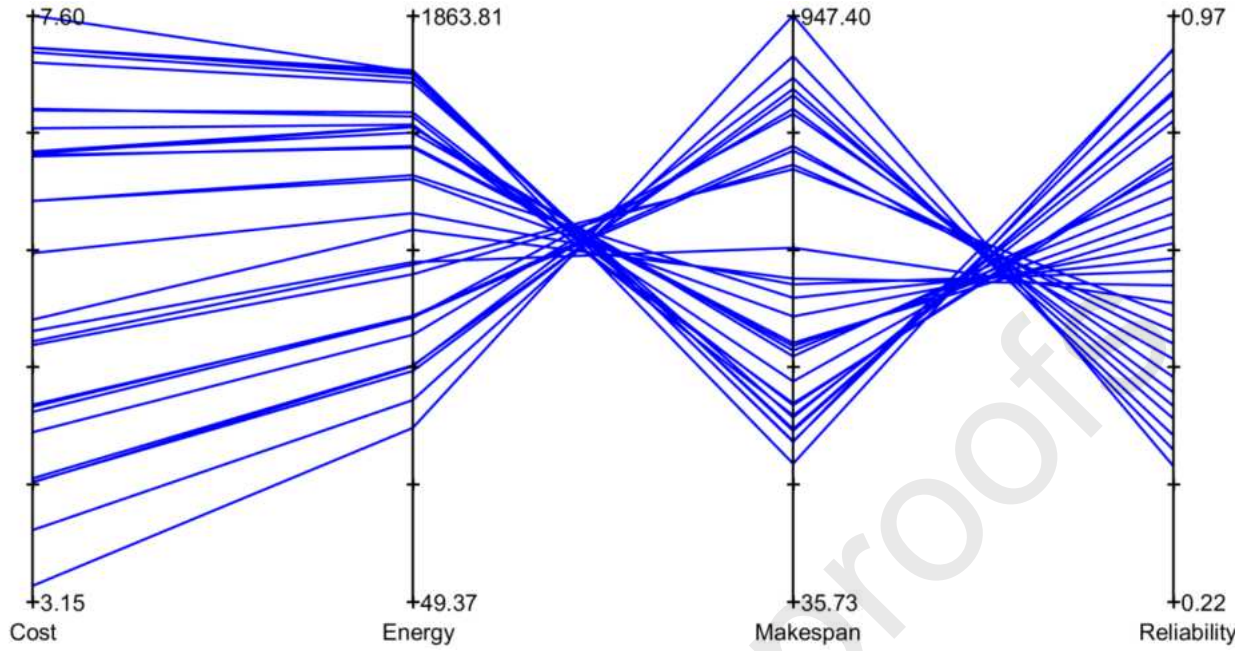
Figure 10 shows the average of cost, makespan, energy consumption, and reliability for each algorithm's non-dominated solutions. Although in Cybershake workflow the results of all algorithms are close to each other, with a closer look, it can be concluded that the first to forth rankings are assigned to the proposed approach, LEAF , MaOPSO, and EMS-C algorithms, respectively. Obviously, achieving a lower makespan requires resources that are more expensive. This Figure shows that reaching the lowest makespan requires the use of resources with more calculating power so there is a significant difference between energy consumption and cost at the beginning and end of the plot. The significant difference between the proposed approach and the other two algorithms is given in the uniformity of solutions. In the proposed approach, in the condition of incomparable solutions, the solution with the least Euclidean distance to the hyperplane is selected as the new cognitive leader. The main advantage of this choice is that during the search process and creating the population members, a balance between the diversification and intensification capabilities of the proposed algorithm is established. In intensification the areas around Pareto-optimal front are explored more carefully in order to find better solutions, while in diversification non-explored areas must be visited to make ensure that all areas of the search space are uniformly explored and that the search is not confined to a reduced number of areas (e.g., around the ideal point). In general, all three approaches have been able to obtain an acceptable score for reliability. According to Fig. 11, the HV criterion of the proposed scheduling algorithm has improved up to 7.59%, 9.72%, and 10% respectively, compared to LEAF , MaOPSO, and EMS-C algorithms. In the LEAF algorithm, author applies a □□□ □□□□□□□□□□□□□□□MaOPSO, we apply a roulette wheel selection process instead of the tournament selection to enhance the selecting the social leader.



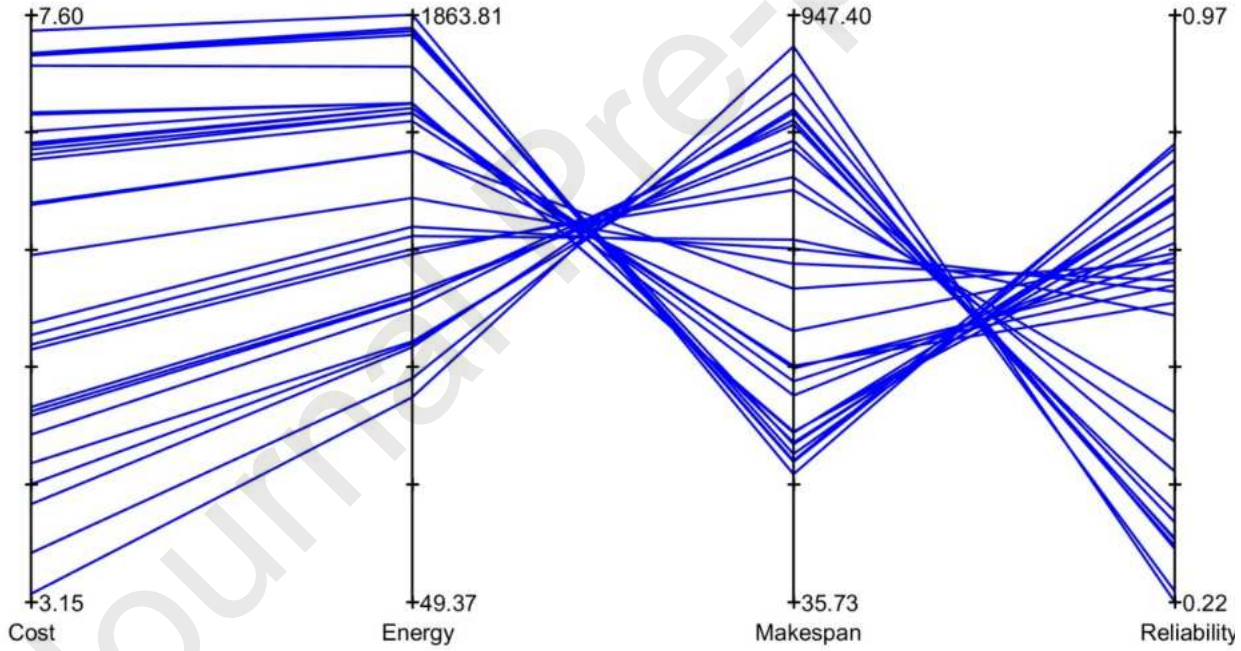
(a) Non-dominated solutions



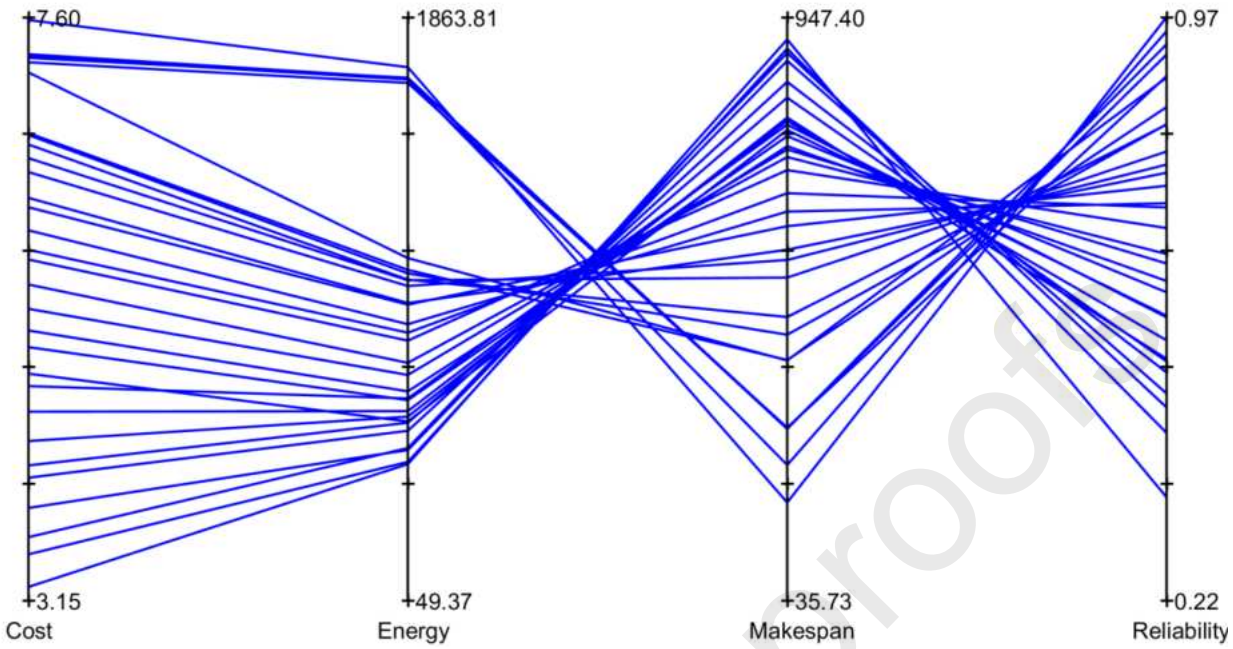
(b) MaOPSO



(c) LEAF



(d) EMS-C



(e) Proposed approach

Fig.10. Parallel coordinates plot of non-dominated solutions obtained from different approaches of Cybershak workflow scheduling for 4-objective

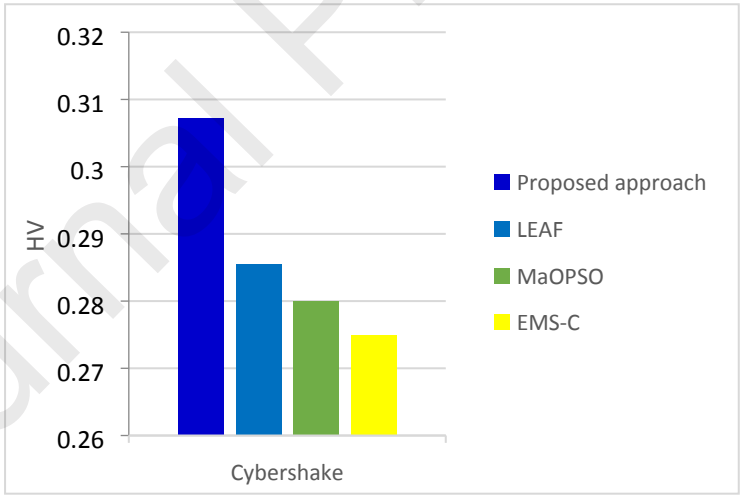
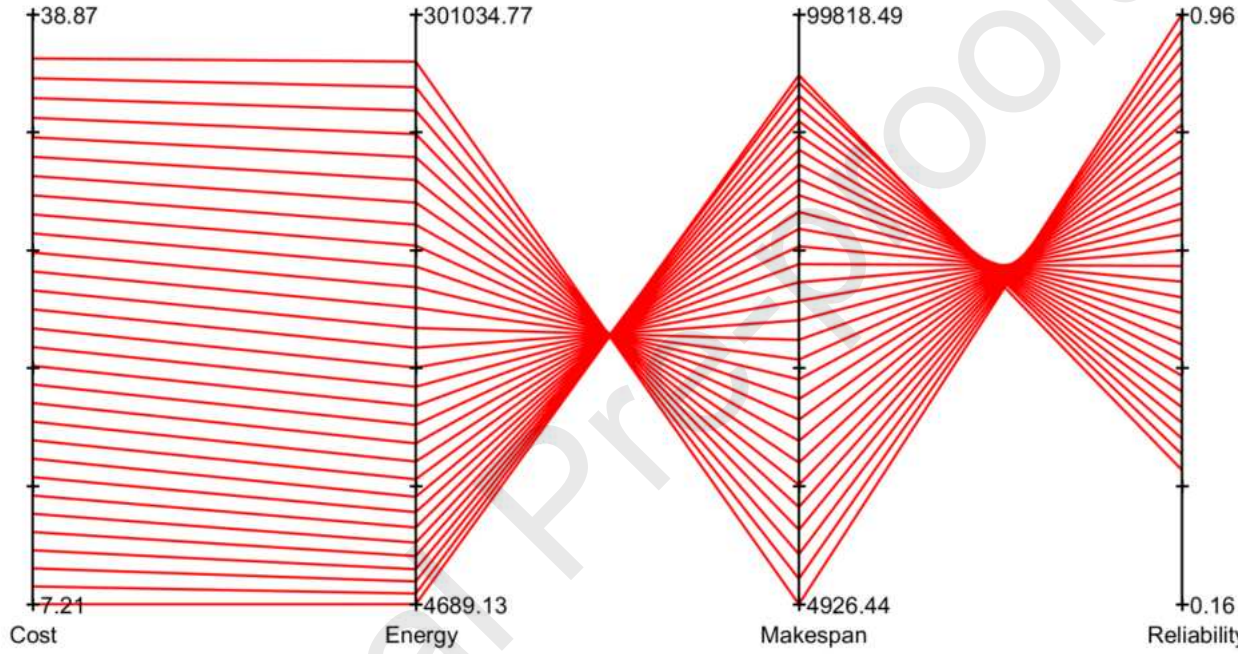


Fig. 11. Comparison of the HV criterion for the Cybershak workflow

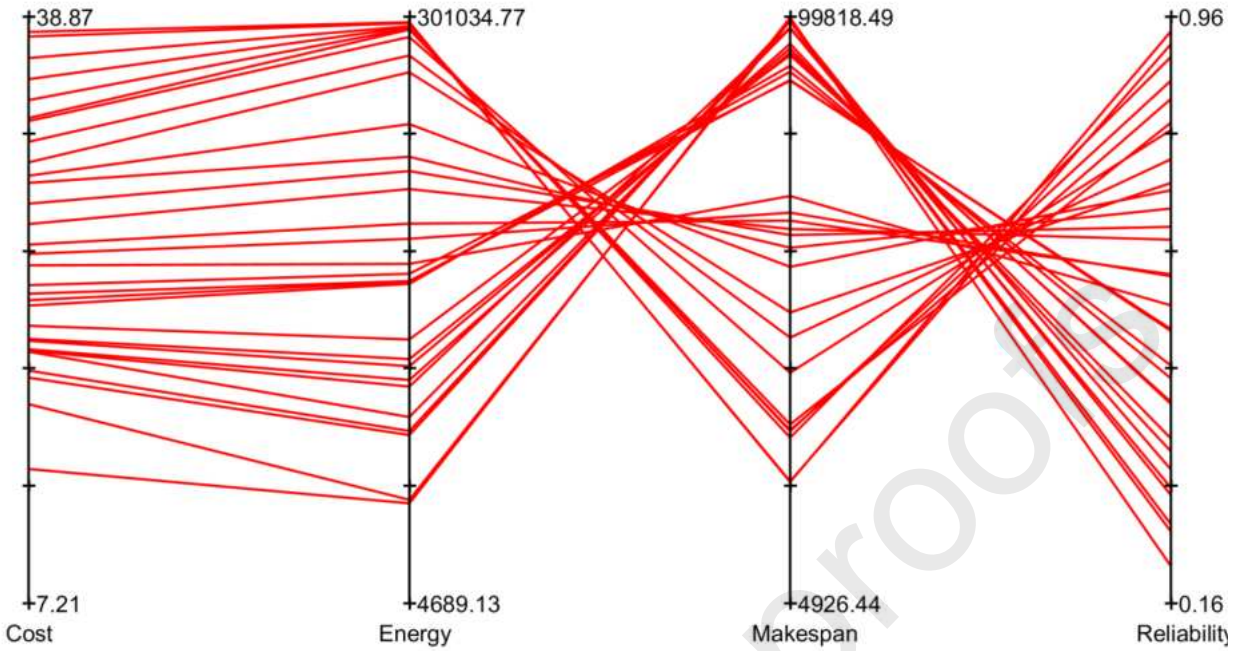
c) Epigenomics workflow

Epigenomics is one of the most difficult workflows for scheduling. The high volume of processing and data transfer between tasks has made it difficult to provide optimal scheduling. Therefore, maintaining the balance between convergence and uniformity of distribution of solutions in this workflow is an important challenge. Taking into account fig. 12 and examining the differences in the values of the makespan, cost and energy consumption of the Epigenomics with two other workflows, the complexity of this workflow is clear. It should be noted that each time the algorithms are executed, 30 optimal solutions (schedulers) are produced. As shown in Fig. 13, according to the

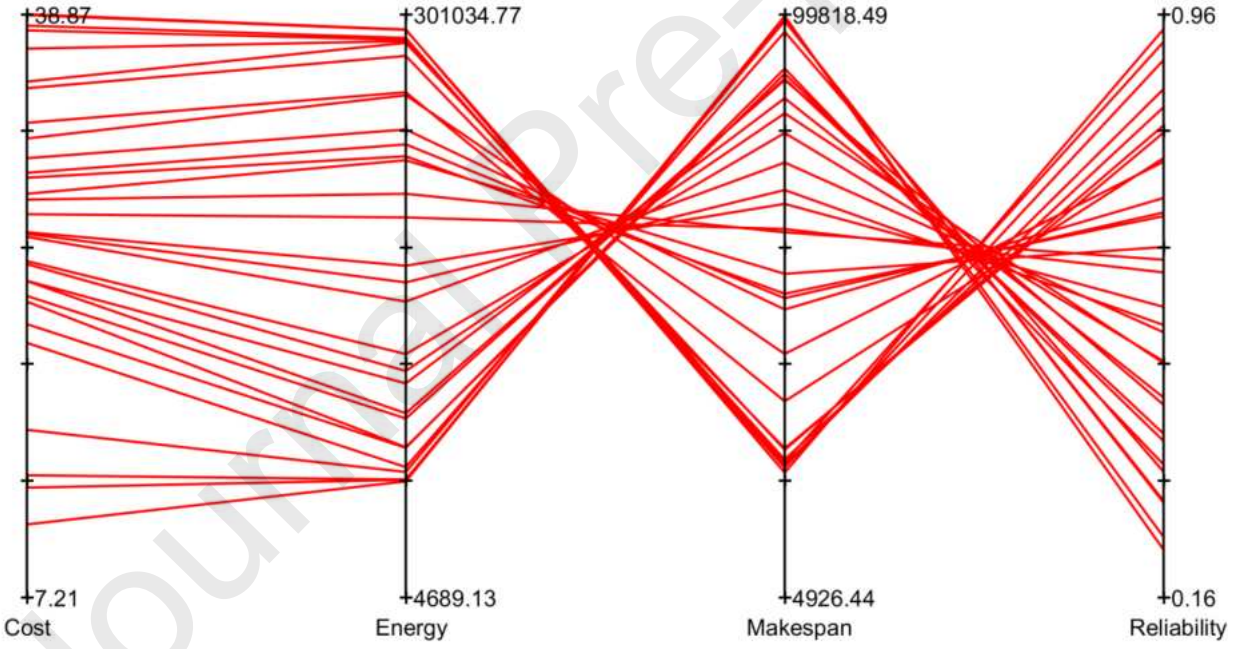
HV criterion, the proximity of the solutions provided by the proposed approach to non-dominated solutions relative to the MaOPSO, LEAF, and EMS-C algorithms has been improved up to 8.19% , 21.78%, and 25% which demonstrates the remarkable advantage of the proposed approach to the EMS-C algorithm and relative advantage over the MaOPSO and LEAF algorithms. In the proposed approach, in order to select the social leader for a sub-swarm, the roulette wheel selection procedure is used instead of the tournament selection. The roulette wheel selection guarantees that the chance of selecting a weaker solution as social leader be smaller than the probability of selecting a better solution. Contrast with the less sophisticated selection algorithm (i.e., the truncation selection used in MaOPSO) which will select a fixed percentage of the best candidates, roulette wheel selection always gives a chance to all of solutions in the archive to be selected. In addition, we apply an efficient approach to choose the new cognitive leader in order to achieve a balance between the diversification and intensification capabilities of the proposed approach.



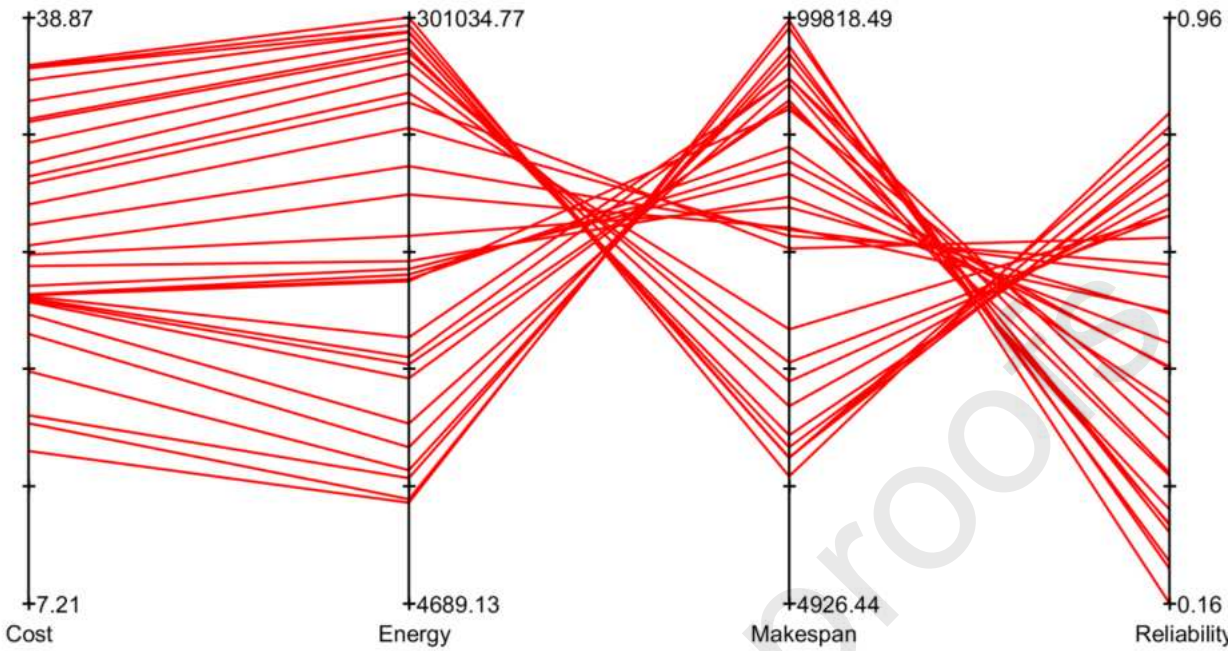
(a) Non-dominated solutions



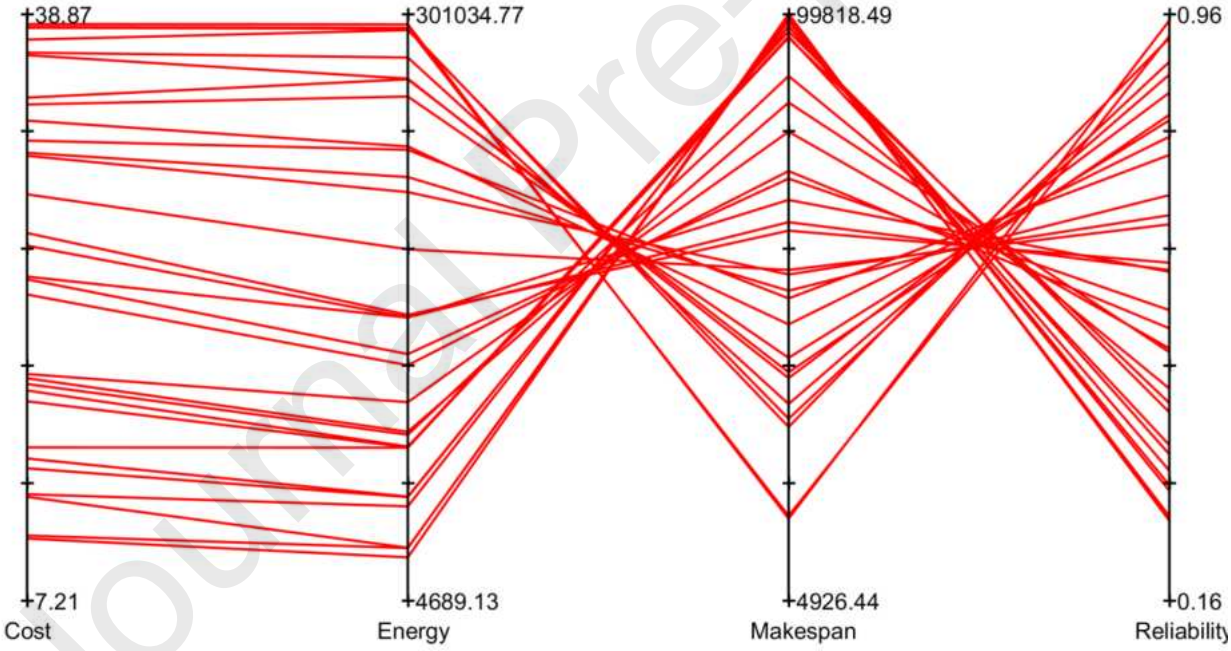
(b) MaOPSO



(c) LEAF



(d) EMS-C



(e) Proposed approach

Fig.12. Parallel coordinates plot of non-dominated solutions obtained from different approaches of Epigenomics workflow scheduling for 4-objective

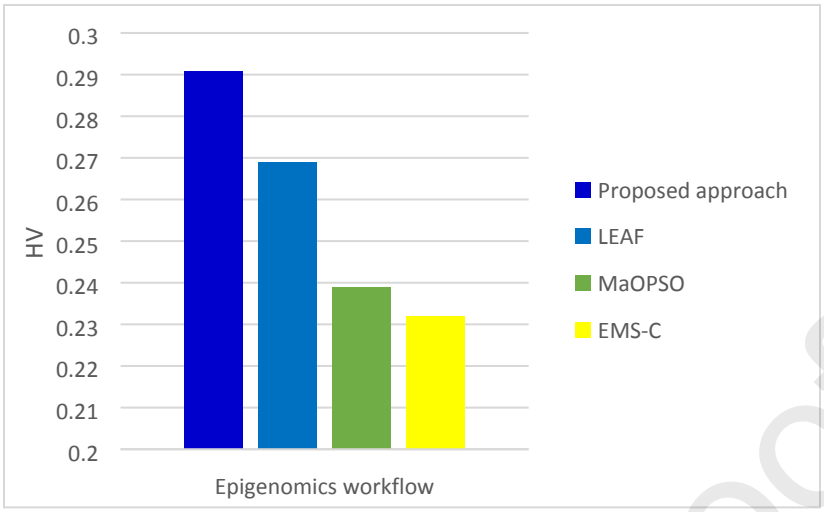


Fig. 13. Comparison of the HV criterion for the Epigenomics workflow

Experiment 2: Evaluation based on makespan parameter with variable VM count

Here, the VM count is raised from 16 to 128 to determine the efficiency of the algorithms in makespan. The average makespan gained for Montage, Cybershake, and Epigenomics are shown in Fig. 14(a-c). When the VM number increased from 16 to 96, the makespan for three approaches decreased. When it increased from 96 to 128, neither approach changed. It can be concluded that: (1) the workflow makespan will decrease when the VMs increase and will reach a point where further addition of VMs fail to decrease the makespan; (2) an increase in the number of VMs up to 96 improved makespan up to 25.8% , 44.35% , 45% for the proposed approach over the LEAF, MaOPSO, EMS-C respectively. In addition, for Cybershake, the difference between the makespan obtained by the proposed algorithm and the LEAF is low, where; the proposed algorithm has a better overall makespan over others. This is because in the proposed approach, we apply an efficient approach to compute the velocity of particles in order to achieve an appropriate balance between exploration and exploitation.

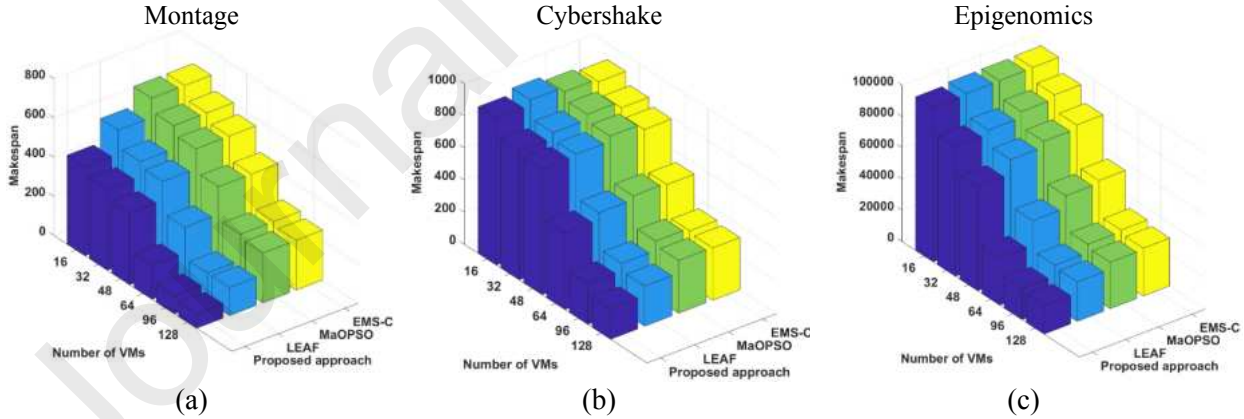
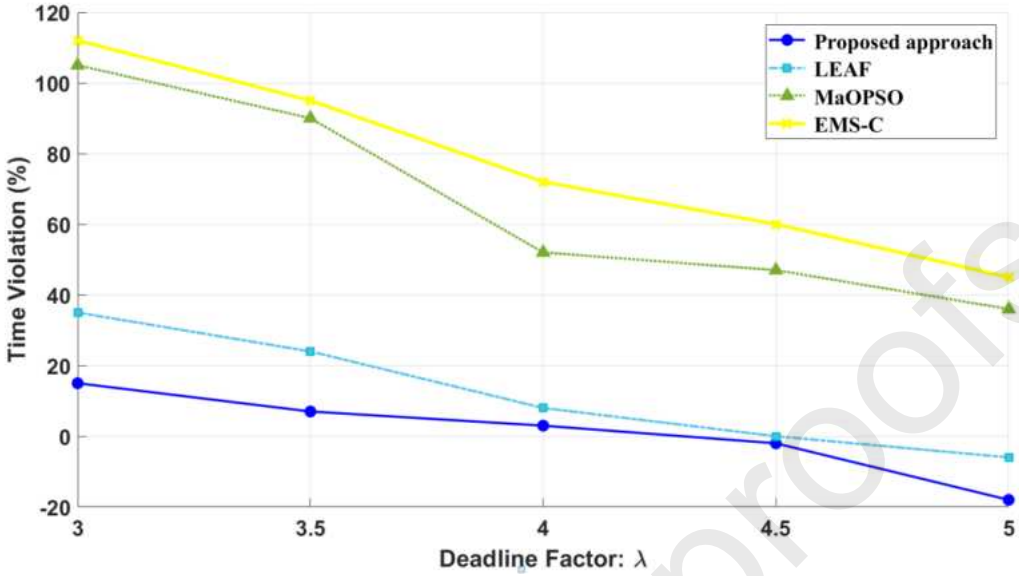


Fig.14. Impact of VM count. (a) On the makespan in Montage workflow. (b) On the makespan in Cybershak workflow. (c) On the makespan in Epigenomics workflow.

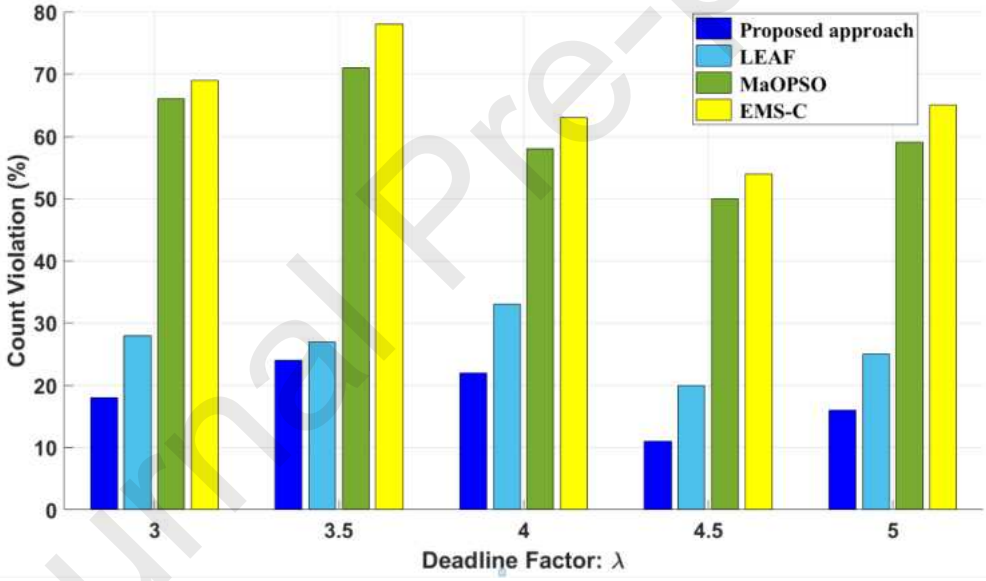
Experiment 3: Evaluation based on deadline violation probability with variable λ of the workflow deadline

In order to evaluate the deadline violation probability of the four algorithms, in this subsection, we compare the average time violation and count violation of the algorithms under various λ. The deadline factor λ is tested with the values from 3 to 5 with an increment of 0.5, and fix the workflow count. Fig. 15 shows that all four algorithms have a few workflows that do not complete before the deadlines. In addition, the deadline violation probabilities of the four algorithms decrease with the increment of λ i.e., extending the deadlines of workflows. The reason is when extending the deadlines of workflows; the workflows can be completed later without violating their deadlines. From Figs. 15(a)

and 15(b), we can observe that the deadline violation probability of EMS-C and MaOPSO are much higher than the results of proposed approach and LEAF because EMS-C and MaOPSO fail to deal with the variable execution time.



(a)



(b)

Fig.15. Deadline violation of workflow scheduling. (a) Time violation. (b) Count violation

6. Conclusions and Future Work

Maintaining a balance between various and conflicting requirements of both users and cloud providers is an important issue in workflow scheduling in cloud computing. In this research, the four-objective workflow scheduling problem was addressed using a many-objective I_MaOPSO optimization algorithm with the aim of producing optimal schedules close to the non-dominated solutions and it tries to minimize makespan, minimize energy consumption and cost, and maximize the reliability of the VMs. In order to increase the probability of creating a 'more explored' initial population, we propose four greedy heuristic method for generating four of N initial solution which consider the objective functions in order to create the initial solutions. In addition, in the I_MaOPSO an efficient approach is used to set and change the amounts of coefficients in order to strike a balance between exploration and exploitation. The simulation was performed with three different workflows called Montage, Cybershake, Epigenomics, each with different characteristics in terms of complexity and data communications. The experimental results show that the

proposed approach can improve up to 71%, 182%, 262% the HyperVolume (HV) criterion compared with the LEAF, MaOPSO, and EMS-C algorithms respectively. In the overall comparison between the four evaluated algorithms, proposed approach ranked first, and LEAF, MaOPSO, and EMS-C respectively ranked second, third, forth. In future studies, we intend 1) to address the issue of security and confidentiality of the user's data for user's workflow tasks and cluster the resources and tasks based on their security level, 2) to propose workload prediction and multi-objective auto-scaling using artificial intelligence (AI) for dynamic resource provisioning can be accomplished using Deep learning, 3) to find out the trade-off among different optimization objectives due to wide range of the Internet of Things (IoT) applications are running on cloud computing systems, 4) to assess the performances of the proposed approach on heterogeneous fog environments, and 5) to use novel AI motivated techniques for more efficient thermal aware scheduling of scientific workflow and resources. Finally, the interested readers can further explore using extensive survey of computing paradigms and technologies and the influence of triumvirate (block chain, IoT and AI) to the evolution of cloud computing [63].

References

[1] Kaur P, Mehta S. Resource provisioning and work flow scheduling in clouds using augmented Shuffled Frog Leaping Algorithm. Journal of Parallel and Distributed Computing. 2017 Mar 1;101:41-50.

[2] Khorsand R, Ramezanpour M. An energy-efficient task-scheduling algorithm based on a multi-criteria decision-making method in cloud computing. International Journal of Communication Systems. 2020 Jun;33(9):e4379.

[3] Khorsand R, Safi-Esfahani F, Nematbakhsh N, Mohsenzade M. Taxonomy of workflow partitioning problems and methods in distributed environments. Journal of Systems and Software. 2017 Oct 1;132:253-71.

[4] Fakhfakh F, Kacem HH, Kacem AH. Workflow scheduling in cloud computing: a survey. In2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations 2014 Sep 1 (pp. 372-378). IEEE.

[5] Dorronsoro B, Nesmachnow S, Taheri J, Zomaya AY, Talbi EG, Bouvry P. A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems. Sustainable Computing: Informatics and Systems. 2014 Dec 1;4(4):252-61.

[6] Khorsand R, Ghobaei-Arani M, Ramezanpour M. FAHP approach for autonomic resource provisioning of multitier applications in cloud computing environments. Software: Practice and Experience. 2018 Dec;48(12):2147-73.

[7] Khorsand R, Safi-Esfahani F, Nematbakhsh N, Mohsenzade M. ATSDS: adaptive two-stage deadline-constrained workflow scheduling considering run-time circumstances in cloud computing environments. The Journal of Supercomputing. 2017 Jun 1;73(6):2430-55.

[8] Choudhary A, Gupta I, Singh V, Jana PK. A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing. Future Generation Computer Systems. 2018 Jun 1;83:14-26.

[9] Verma A, Kaushal S. A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling. Parallel Computing. 2017 Feb 1;62:1-9.

[10] Lee YC, Zomaya AY, Yousif M. Reliable workflow execution in distributed systems for cost efficiency. In2010 11th IEEE/ACM International Conference on Grid Computing 2010 Oct 25 (pp. 89-96). IEEE.

[11] Szabo C, Kroeger T. Evolving multi-objective strategies for task allocation of scientific workflows on public clouds. In2012 IEEE Congress on Evolutionary Computation 2012 Jun 10 (pp. 1-8). IEEE.

[12] Figueiredo EM, Ludermir TB, Bastos-Filho CJ. Many objective particle swarm optimization. Information Sciences. 2016 Dec 20;374:115-34.

[13] Fard HM, Prodan R, Fahringer T. Multi-objective list scheduling of workflow applications in distributed computing infrastructures. Journal of Parallel and Distributed Computing. 2014 Mar 1;74(3):2152-65.

[14] Yazdanbakhsh M, Khorsand R. A Task Scheduling Strategy to Improve Qualitative Features in the Cloud Computing Environment. TABRIZ JOURNAL OF ELECTRICAL ENGINEERING. 2019 Dec 1;49(3):1427-37.

[15] Chen H, Tian Y, Pedrycz W, Wu G, Wang R, Wang L. Hyperplane assisted evolutionary algorithm for many-objective optimization problems. IEEE transactions on cybernetics. 2019 Mar 4.

[16] He C, Tian Y, Jin Y, Zhang X, Pan L. A radial space division based evolutionary algorithm for many-objective optimization. Applied Soft Computing. 2017 Dec 1;61:603-21.

[17] Sharma D, Shukla PK. Line-prioritized environmental selection and normalization scheme for many-objective optimization using reference-lines-based framework. Swarm and Evolutionary Computation. 2019 Dec 1;51:100592.

[18] Zhou J, Gao L, Yao X, Zhang C, Chan FT, Lin Y. Evolutionary algorithms for many-objective cloud service composition: Performance assessments and comparisons. Swarm and Evolutionary Computation. 2019 Dec 1;51:100605.
[19] Patil MV, Kulkarni AJ. Pareto dominance based Multiobjective Cohort Intelligence algorithm. Information Sciences. 2020 May 29.
[20] Fan R, Wei L, Sun H, Hu Z. An enhanced reference vectors-based multi-objective evolutionary algorithm with neighborhood-based adaptive adjustment. Neural Computing and Applications. 2019 Dec 14:1-23.
[21] Li T, Li J. Using modified determinantal point process sampling to update population. In2018 IEEE Congress on Evolutionary Computation (CEC) 2018 Jul 8 (pp. 1-7). IEEE.
[22] Wan K, He C, Camacho A, Shang K, Cheng R, Ishibuchi H. A Hybrid Surrogate-Assisted Evolutionary Algorithm for Computationally Expensive Many-Objective Optimization. In2019 IEEE Congress on Evolutionary Computation (CEC) 2019 Jun 10 (pp. 2018-2025). IEEE.
[23] S. O'Grady, AWS: Forget the revenue, did you see the margins?, http://redmonk.com/sogrady/2010/08/04/aws-margins/, accessed: 2019-02-09.
[24] Rodriguez MA, Buyya R. Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE transactions on cloud computing. 2014 Apr 2;2(2):222-35.
[25] Juve G, Chervenak A, Deelman E, Bharathi S, Mehta G, Vahi K. Characterizing and profiling scientific workflows. Future Generation Computer Systems. 2013 Mar 1;29(3):682-92.
[26] Ostermann S, Iosup A, Yigitbasi N, Prodan R, Fahringer T, Epema D. A performance analysis of EC2 cloud computing services for scientific computing. InInternational Conference on Cloud Computing 2009 Oct 19 (pp. 115-131). Springer, Berlin, Heidelberg.
[27] Liu J, Ren J, Dai W, Zhang D, Zhou P, Zhang Y, Min G, Najjari N. Online multi-workflow scheduling under uncertain task execution time in IaaS clouds. IEEE Transactions on Cloud Computing. 2019 Mar 19.
[28] Khorsand R, Ghobaei-Arani M, Ramezanpour M. A self-learning fuzzy approach for proactive resource provisioning in cloud environment. Software: Practice and Experience. 2019 Nov;49(11):1618-42.
[29] Ghobaei-Arani M, Khorsand R, Ramezanpour M. An autonomous resource provisioning framework for massively multiplayer online games in cloud environment. Journal of Network and Computer Applications. 2019 Jun 7.
[30] Jain H, Deb K. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: handling constraints and extending to an adaptive approach. IEEE Transactions on Evolutionary Computation. 2013 Sep 11;18(4):602-22.
[31] Nasonov D, Butakov N. Hybrid scheduling algorithm in early warning systems. Procedia Computer Science. 2014 Jan 1;29:1677-87.
[32] Safari M, Khorsand R. Energy-aware scheduling algorithm for time-constrained workflow tasks in DVFS-enabled cloud environment. Simulation Modelling Practice and Theory. 2018 Sep 1;87:311-26.
[33] Safari M, Khorsand R. PL-DVFS: combining Power-aware List-based scheduling algorithm with DVFS technique for real-time tasks in Cloud Computing. The Journal of Supercomputing. 2018 Oct 1;74(10):5578-600.
[34] Rafieyan E, Khorsand R, Ramezanpour M. An Adaptive Scheduling Approach based on Integrated Best-worst and VIKOR for Cloud computing. Computers & Industrial Engineering. 2020 Jan 8:106272.
[35] Zhu K, Song H, Liu L, Gao J, Cheng G. Hybrid genetic algorithm for cloud computing applications. In2011 IEEE Asia-Pacific Services Computing Conference 2011 Dec 12 (pp. 182-187). IEEE.
[36] Zhan ZH, Liu XF, Gong YJ, Zhang J, Chung HS, Li Y. Cloud computing resource scheduling and a survey of its evolutionary approaches. ACM Computing Surveys (CSUR). 2015 Jul 21;47(4):63.
[37] Huang KC, Tsai YL, Liu HC. Task ranking and allocation in list-based workflow scheduling on parallel computing platform. The Journal of Supercomputing. 2015 Jan 1;71(1):217-40.
[38] Nishant K, Sharma P, Krishna V, Gupta C, Singh KP, Rastogi R. Load balancing of nodes in cloud using ant colony optimization. In2012 UKSim 14th International Conference on Computer Modelling and Simulation 2012 Mar 28 (pp. 3-8). IEEE.
[39] Ding D, Fan X, Zhao Y, Kang K, Yin Q, Zeng J. Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Generation Computer Systems. 2020 Feb 28.
[40] Wang X, Wang Y, Zhu H. Energy-efficient multi-job scheduling model for cloud computing and its genetic algorithm. Mathematical Problems in Engineering. 2012;2012.

[41] Chen WN, Zhang J. A set-based discrete PSO for cloud workflow scheduling with user-defined QoS constraints. In2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2012 Oct 14 (pp. 773-778). IEEE.
[42] Guzek M, Pecero JE, Dorronsoro B, Bouvry P. Multi-objective evolutionary algorithms for energy-aware scheduling on distributed computing systems. Applied Soft Computing. 2014 Nov 1;24:432-46.
[43] Durillo JJ, Nae V, Prodan R. Multi-objective energy-efficient workflow scheduling using list-based heuristics. Future Generation Computer Systems. 2014 Jul 1;36:221-36.
[44] Kaur N, Singh S. A budget-constrained time and reliability optimization bat algorithm for scheduling workflow applications in clouds. Procedia Computer Science. 2016 Jan 1;98:199-204.
[45] Li J, Su S, Cheng X, Huang Q, Zhang Z. Cost-conscious scheduling for large graph processing in the cloud. In2011 IEEE international conference on high performance computing and communications 2011 Sep 2 (pp. 808-813). IEEE.
[46] Jena RK. Energy efficient task scheduling in cloud environment. Energy Procedia. 2017 Dec 1;141:222-7.
[47] Verma A, Kaushal S. Bi-criteria priority based particle swarm optimization workflow scheduling algorithm for cloud. In2014 Recent Advances in Engineering and Computational Sciences (RAECS) 2014 Mar 6 (pp. 1-6). IEEE.
[48] Arabnejad V, Bubendorfer K, Ng B. Scheduling deadline constrained scientific workflows on dynamically provisioned cloud resources. Future Generation Computer Systems. 2017 Oct 1;75:348-64.
[49] Mansouri N, Zade BM, Javidi MM. Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Computers & Industrial Engineering. 2019 Apr 1;130:597-633.
[50] Zhou A, Qu BY, Li H, Zhao SZ, Suganthan PN, Zhang Q. Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation. 2011 Mar 1;1(1):32-49.
[51] Garg R, Singh AK. Multi-objective optimization to workflow grid scheduling using reference point based evolutionary algorithm. International Journal of Computer Applications. 2011 May;22(6):44-9.
[52] Chen H, Zhu X, Liu G, Pedrycz W. Uncertainty-aware online scheduling for real-time workflows in cloud service environment. IEEE Transactions on Services Computing. 2018 Aug 21.
[53] Gill SS, Buyya R, Chana I, Singh M, Abraham A. BULLET: particle swarm optimization based scheduling technique for provisioned cloud resources. Journal of Network and Systems Management. 2018 Apr 1;26(2):361-400.
[54] Ye X, Liu S, Yin Y, Jin Y. User-oriented many-objective cloud workflow scheduling based on an improved knee point driven evolutionary algorithm. Knowledge-Based Systems. 2017 Nov 1;135:113-24.
[55] Yang S, Li M, Liu X, Zheng J. A grid-based evolutionary algorithm for many-objective optimization. IEEE Transactions on Evolutionary Computation. 2013 Jan 1;17(5):721-36.
[56] Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE transactions on Evolutionary Computation. 2002 Aug 7;6(1):58-73.
[57] Gandibleux X, editor. Multiple criteria optimization: state of the art annotated bibliographic surveys. Springer Science & Business Media; 2006 Apr 11.
[58] Zhu Z, Zhang G, Li M, Liu X. Evolutionary multi-objective workflow scheduling in cloud. IEEE Transactions on parallel and distributed Systems. 2015 Jun 17;27(5):1344-57.
[59] Juve G, Chervenak A, Deelman E, Bharathi S, Mehta G, Vahi K. Characterizing and profiling scientific workflows. Future Generation Computer Systems. 2013 Mar 1;29(3):682-92.
[60] Yao G, Ding Y, Ren L, Hao K, Chen L. An immune system-inspired rescheduling algorithm for workflow in Cloud systems. Knowledge-Based Systems. 2016 May 1;99:39-50.
[61] Yen GG, He Z. Performance metric ensemble for multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation. 2013 Jan 16;18(1):131-44.
[62] Lim KS, Buyamin S, Ahmad A, Shapiai MI, Naim F, Mubin M, Kim DH. Improving vector evaluated particle swarm optimisation using multiple nondominated leaders. The Scientific World Journal. 2014; 2014.
[63] Gill SS, Tuli S, Xu M, Singh I, Singh KV, Lindsay D, Tuli S, Smirnova D, Singh M, Jain U, Pervaiz H. Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges. Internet of Things. 2019 Sep 19:100118.

Sahar Saeedi: Software, Writing - Original Draft, Investigation
Reihaneh Khorsand : Conceptualization, Methodology, Writing - Review & Editing
Somaye Ghandi Bidgoli: Resources, Supervision, Validation
Mohammadreza Ramezanpour : Project administration, Visualization

Highlights:

- Proposing four greedy heuristic methods to generate uniformly distributed particles
- Applying an efficient approach to compute the velocity of particles.
- Applying a roulette wheel selection process to select the social leader.
- Applying an efficient approach to choose the new cognitive leader.