A NEW OPTIMIZATION METHOD FOR SECURITY-CONSTRAINED WORKFLOW SCHEDULING

Ali Abdali
*Departmentof Management, Amin Police University, Tehran, Iran
abdali9192@gmail.com

SafaMeasoomyNia
Department of Mathematics, Islamic Azad University, Urmia Branch, P.O. BOX 969, Urmia, Iran
measoomy@yahoo.com

Abstract - Broadly speaking, scheduling is a process of developing bunches of security policies and to control the certain available tasks which have to be taken care of using a computer system. Care should be exercised that a scheduler in a particular fields should be quite cautious in adapting scheduling strategy as the environment and the version of task change. In many cases, security is proven to be able to encourage relevance between two entities for a long time. This does imply that a secure scheduling algorithm, of course, can completely alleviate the failure probability during task execution in a security environment. Therefore, to move in a direction of a systematic approach, this paper, very natural, aims at introducing a brand new reliable algorithm using different quality of service (QoS) parameters to manage the scheduling large number of workflows. To do this, it combines two metaheuristic algorithms, CPSO and GA. It should be noted, however, that three different QoS such as execution cost, loadbalancing and security are employed as the most immediate performance measure to handle the scheduling process. Conclusion reached through this algorithm mention that its total execution cost will be minimized while meeting deadline and risk rate constraints. Verification of the proposed algorithm with other algorithms was taken into consideration. The accumulation of results through exercising this algorithm reveals that optimal solution to the problems is promised. Therefore, it is highly desirable to claim that obtained results from this effective approach are better than with the approach used with other algorithms such as CPSO and GA.

***Keywords*:** Cloud computing; Workflow scheduling; QoS; Meta heuristic; Security; Chaotic PSO.

1. Introduction

This also seems simple to state that Cloud computing attempts to share a huge set of virtual computer resources and computational equipment, storage, information and knowledge for scientific investigations on the internet. It refers to a large number of computational resources on demand through pay-per-use model [1]. Cloud computing gives users the permission to use applications via the internet without being installed [2]. It comprises a large amount of heterogeneous distributed resources to deliver large number of services in particular applications to its users with specific quality of service (QoS) requirements [3]. The main idea of Cloud computing is to satisfy various requirements in a large distributed systems in which cloud users access Quality of Service based on demands [4]. In cloud environment, load balancing becomes a more serious problem. In cloud computing environment, running multiple tasks in a workflow is required over the available set of resources at the same time. Load balancing would lead to a better result. Implementation and application of all the resources can be appropriate [5], therefore efficiency of the system would be followed [6].

In the cloud computing environment, Scheduling performs the most distinguished operations to reach the most benefit [7]. The maximum system throughput and high efficiency of computing is going to be full filled by task scheduling algorithm. It happens to handle accessibility of CPU memory and a good scheduling policy giving maximum use of resource [8]. Workflow scheduling is the problem of allocating each task is workflow scheduling to a suitable resource and letting the tasks to please some efficiency criterion [9]. Too often, a workflow involves a set of tasks,each might communicate with one another in the workflow [10]. Generally speaking, a workflow is determined by a Directed Acyclic Graph (DAG) in which each task is represented by a node, and each of the data or a dependency between tasks is allocated by a directed edge between the corresponding nodes [11].

As the size of data application is very big, security would be imperative for various scientific workflows and such data takes a long time on large-scale distributed infrastructures to be executed [12]. Thus, integrating trust can improve scheduling performance [13]. Having a trust-based mechanism in scheduling would increase failure ratio and reassign in cloud environments [14]. Security and efficiency isolation minimize rent cost and raise network guarantees for applications [15]. The security cloud similarity is the characterization of users, satisfaction by the virtual machine allocated by tasks [16]. In cloud computing, delivering services and resources on demand over a network requires a lot of technological subjects, consisting of automated provisioning, dynamic virtual server migration, or network security problems. It should be mentioned that, in a cloud environment, due to network latency, commercial agreements, or some security policy issues [17], all the resources may actually not be available to all customers.

In order to provide reader with the present work, some relevant works studied will be discussed: Li et al., propose trust based mechanism into the workflow scheduling algorithm that minimizes the completion of time and improves the execution success rate and user satisfaction [13]. Kumar et al., introduce a task scheduling algorithm and allocation of resources in cloud environment. Enhancing the reliability and minimizing the total cost, execution cost, total turn-around, total waiting time and total execution time needed to be focused on [17]. Liu et al., argued the variable neighborhood search particle swarm optimization (VNPSO) that helps the particles trapped in local minima which are not caught [18]. Wang et al., investigate Look Ahead Genetic Algorithm (LAGA) which employ the RD reputation to enhance the makes pan and reliability of a workflow scheduling application [19]. A Security and Cost Aware Scheduling (SCAS) algorithm is presented to minimize the total execution cost considering deadline and risk rate constraints by Li et al., in [12]. Li introduces a task scheduling algorithm based on CPSO (Chaotic PSO) that enhance the global convergence and obtains a global best solution by creating the sequence from chaotic systems and minimizes the cost of scheduling [20]. Wu presents a revised discrete particle swarm optimization (RDPSO) to increase the total execution cost and the total makes pan of the workflow application. This algorithm determines the velocity and position of particles pursuant to the characteristics of discrete variables [21]. Yang et al., in [22], propose a trust-based workflow scheduling algorithm (TBHSA) that minimizes the cost and employs a global search algorithm to reach the optimum scheduling solution.

Singh et al., present a budget constrained time minimization genetic algorithm in cloud computing environment that meets QoS constraints determined by the user. This algorithm decreases the failure rate that makes pan [23]. Marcon et al. consider an optimized and effective method in hybrid cloud environment to apply resources from private clouds as well as public clouds, that decrease tenant cost considering the workflow requirements [24]. Jianfang et al., in [16], investigate a workflow scheduling algorithm of the cloud computing environment to use discrete particle swarm optimization that improves security, completion time, cost and load balancing. Zeng et al. propose a Security-Aware and Budget-Aware workflow scheduling algorithm (SABA). This algorithm-minimizes the execution time within the user's security requirement and budget constraint in cloud computing environments where multidimensional computing resources are considered [25]. Wad et al., in [26], discuss mathematical model using load balancing mutation a particle swarm optimization (LBMPSO). It minimizes cost, round trip time, execution time, and transmission time and optimizes the reliability of cloud environment and well allocates tasks to resources. GhorbanniaDelavar et al., propose a hybrid meta-heuristic Genetic Algorithm (GMSW) to reach a suitable solution for assigning the tasks on resources [27]. For data intensive workflow applications in cloud computing Chen et al., [28] propose privacy and cost aware scheduling algorithm based on genetic algorithm which minimizes the computation cost, the cost of data transmission and the cost of data storage.

To prevent the dataset-datacenter mapping problem that minimizes the data transmission cost, Li et al., propose a novel strategy based on discrete binary PSO for scientific workflow scheduling in cloud environment [29]. A hybrid Particle Swarm Optimization for workflow scheduling in cloud computing is investigated by Sridhar. This algorithm picks out proper resources and handles load among resources and decreases the execution time [30]. An endocrine-based co-evolutionary multi-swarm for multi-objective optimization algorithm (ECMSMOO), for workflow scheduling in cloud computing system is proposed by Yao et al., [31]. It of course, optimizes objectives, such as cost, makes pan and energy consumption. A novel workflow scheduling algorithm based on PSO in cloud computing to achieve the scheduling solutions that reduce the makes pan considering the user's budget constraint is introduced by Wang et al., [32]. Verma and Kaushal in [33] enhance Bi-Objective Priority based Particle Swarm Optimization (BPSO) algorithm for scheduling workflow applications to cloud resources that need to decrease the execution cost considering the deadline constraint and the budget constraint. Jafarzadeh-Shirazi, propose a firefly task scheduling algorithm in cloud computing to reduce the communication cost and computation cost [34]. Wu et al., [35] devised a unified multi-constraint and multi-objective cloud workflow scheduling framework using Pareto optimality theory. This algorithm decreases energy consumption and improve reliability while meeting the deadline and budget constraints. Table 1 illustrates the objectives of these scheduling algorithms.

In this paper, a hybrid meta-heuristic scheduling algorithm for various scientific workflows is proposed by focusing on cost and load balance deviation. The present paper is formulated as follows: 1) regarding user satisfaction, choose appropriate virtual machine; 2) establishing a scheduling model on the cloud workflow of multi-dimensional QoS perception considering security and execution cost in the cloud workflow scheduling; 3) proposing an optimized scheduling algorithm of the cloud workflow based on CPSO algorithm.

The following is structured as follows. The system models and problem formulation is discussed in Section 2. The proposed algorithm implementation is determined in Section 3. Section 4 illustrates the experiment design and evaluation results. Finally, the conclusion and attending our future works is covered in Section 5.

Table 1. Comparison of workflow scheduling schemes

	Feature	Environment	Type of approach
[12]	total execution cost, security and deadline	Cloud environment	Meta-heuristic
[13]	completion time	Cloud environment	heuristic
[16]	security, completion time, cost and load balancing	Cloud environment	Meta-heuristic
[17]	Reliability, total cost, execution cost, total turn-around, total waiting time and total execution time	Cloud environment	heuristic
[19]	makespan and reliability	Cloud environment	Meta-heuristic
[20]	cost	Cloud environment	Meta-heuristic
[21]	execution cost and makespan	Cloud environment	Meta-heuristic
[22]	cost	Cloud environment	
[23]	failure rate and makespan	Cloud environment	Meta-heuristic
[24]	tanent cost	Cloud environment	heuristic
[25]	execution time, security and budget	Cloud environment	heuristic
[26]	decreases cost, round trip time, execution time, transmission time and reliability	Cloud environment	Meta-heuristic
[28]	computation cost, data transmission cost and the cost of data storage	Cloud environment	Meta-heuristic
[29]	data transmission cost	Cloud environment	Meta-heuristic
[30]	execution time	Cloud environment	Meta-heuristic
[31]	cost, makespan and energy consumption	Cloud environment	heuristic
[32]	makespan and budget	Cloud environment	Meta-heuristic
[33]	execution cost, deadline and budget	Cloud environment	Meta-heuristic
[34]	communication cost and computation cost	Cloud environment	Meta-heuristic
[35]	energy consumption, reliability, deadline and budget	Cloud environment	heuristic

2. Problem Statement: Objectives, Assumptions and Constraints

Our problem goes on in scheduling the large number of data scientific workflows by constraints of risk rate for minimizing the execution cost and can also balance the load on resources. In this section, we are about to explain a scientific workflow model, the cloud data center and Security model, and problem formulation, which form the foundation of our method.

2.1 Workflow model

In terms of method, a workflow is modeled by a Directed Acyclic Graph (DAG), presented by a tuple $G\ (T, E)$, where $T = \{t_1, t_2, \dots, t_n\}$ is the set of nodes that $t_i \in T$ defines a task in the workflow where each task is atomic and E is the set of edges representing constraints of priority and the data dependencies between tasks. $d_{i,j} = \left(t_i, t_j\right) \in E$, specify that $t_j$can be executed only by finishing whole task t_i being parent node. In a workflow, entry task refers to the task with no parent, defined asT_{entry}, and exit task is the task with no child, represented asT_{exit}. The structure of four scientific workflow models is illusterated in Fig. 1.

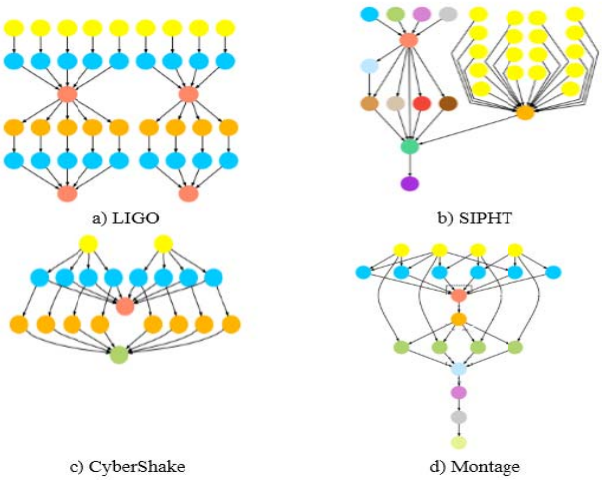


Fig. 1. The structure of four scientific workflows

2.2 Cloud data center solutions and Security models

Probably, suppose that cloud data center organizes a set of S kinds of virtual machines$VM = \{VM_1, VM_2, \dots, VM_s\}$. Each $VM_s$is defined by multiple features, containing the type of virtual machine shown as S, processing capacity denoted as p_s in a million floating point operations per second (MFLOPS) and cost per hour shown by c_s. The time unit of leasing refers to the hour and remainder time unit of usage rounded up to the next entire time unit. For example, the lease of a VM for 1 hour and 20 min will be rounded up to 2 hours, As a result, the cost of leasing is computed based on 2 hours [12].

The security and privacy problems is a big challenge in cloud computing due to its multi-tenancy essence and the outsourcing of infrastructure, sensitive data and important applications. There are three traditional attacks for cloud system: snooping, alteration and spoofing. In addition, we prepare three essential security services called confidentiality, integrity and authentication to protect the scientific cloud applications from these traditional threats.

Now, consider that each task might require all three kinds of security service with certain security levels that users determinate. For instance, sr_i is the set of security requirements of task t_i, which can be specified as a 3-tuple $sr_i = \{sr_i^a, sr_i^c, sr_i^g\}$, where a, g and c demonstrate the authentication, integrity and confidentiality service and $sl_i = \{sl_i^a, sl_i^c, sl_i^g\}$ is the set of security level services provided for task t_i. The security overhead function of the integrity service, confidentiality and authentication service, is formulated by Eq. (1) and Eq. (2).

$$SC^l(t_i) = \beta^l . sl_i^l . d_i^l \qquad l \in \{g, c\} \qquad (1)$$

$$SC^l(t_i) = \beta^l . sl_i^l, \qquad l \in \{a\} \qquad (2)$$

where $d_i^l$is the data of task $t_i$and $\beta^a = 3,\ \beta^g = 4,\ \beta^c = 1$. Then, the total security overhead $SC(t_i)$experienced by task t_i can be computed from Eq. (3).

$$SC(t_i) = \sum_{l \in \{a,g,c\}} SC^l(t_i) \qquad (3)$$

In the cloud data center, each VM can provide all the security services, and users are allowed to choose various VM types with different security levels considering their QoS requirements. Tables 2-4, show encryption algorithms, hash functions and authentication techniques, respectively.

2.3 Problem formulation

We tend to focus on choosing a proper schedule to perform a scientific workflow on cloud resources such that the total execution cost is minimized condsidering the constraints of deadline and risk rate. We create a schedule $schedule = \{TaskInfo, Total_{exeCost}, Total_{exeTime}\}$ that includes of a set of information for all tasks (e.g. security level of tasks, VM mapping of tasks, workload of tasks, and etc.), the total execution cost and the total execution.

Table 1. Cryptographic Algorithms for Confidentiality

Cryptographic Algorithms	Security Level	μ
Seal	0.08	168.75
RC4	0.14	96.43
Blowfish	0.36	37.5
Knufu	0.4	33.75
RC5	0.46	29.35
Rijndael	0.64	21.9
DES	0.9	15
IDEA	1	13.5

Table 2. Hash Functions for Integrity

Hash Functions	Security Level	μ
MD4	0.18	23.9
MD5	0.26	17.09
RIPEMD	0.36	12
RIPEMD-128	0.45	9.73
SHA-1	0.63	6.88
RIPEMD-160	0.77	5.69
Tiger	1	4.36

Table 3. Authentication Methods

Authentication methods	Security Level	μ
HMAC-MD5	0.55	90
HMAC-SHA-1	0.91	148
CBC-MAC-AES	1	163

At first, total execution cost, TEC and total execution time, TET are computed by Eq. (4) and Eq. (5).

$$TEC = \sum_{i=0}^{n-1} c_s.\lceil LET(t_i, vm_s) - LST(t_i, vm_s)\rceil \quad (4)$$

$$TET = \max\{ExeutionTime(t_i)|t_i \in T\}, subject\ to\ TET < T_{deadline} \quad (5)$$

where c_s is the leasing cost task t_i on vm_s. LET and LST are lease end time and lease start time that computed by Eq. (6) and Eq. (7), respectively.

$$LET(t_i, vm_s) = \begin{cases} 0, & ,if\ vm_s(t_i) = vm_s(t_{i-1}) \\ ProcessingTime(t_i, vm_s) + LST(t_i) & ,else \end{cases} \quad (6)$$

$$LST(t_i, vm_s) = \begin{cases} 0, & ,if\ vm_s(t_i) = vm_s(t_{i-1}) \\ StartTime(t_i) & ,else \end{cases} \quad (7)$$

We suppose that task t_i need to be executed prior to taskt_{i-1}. First, users who intend to access input data to prevent from spoofing attack and services are going to be authenticated. Next, the output data of task t_{i-1} is transferred to task t_i, and the corresponding transfer time can be calculated by Eq. (8).

$$TransferTime(t_i) = I_i/B \quad (8)$$

whereI_i is the input data of task $t_i$which is transferred from t_{i-1}. Keep in mind that the time of transfer between two tasks, executed on the same VM, is zero similar to the lease end time and lease start time according to the above formulas. Now, consider that all types of VM have the same communication bandwidth B. The task t_i will be executed using the input data, and the execution time $ExectionTime(t_i, vm_s)$,of task t_i on vm_swill be represented as follows:

$$ExecutionTime(t_i, vm_s) = W_i/p_s \quad (9)$$

Here, W_i is the workload of task t_i and p_s is the computing capacity of vm_s. The processing time, the end time and the start time are calculated by Eqs. (10)-(12) as well as.

$$ProcessingTime(t_i, vm_s) = TransferTime(t_i) + EexecutionTime(t_i) + SC(t_i) \quad (10)$$

$$EndTime(t_i) = ProcessingTime(t_i, vm_s) + StartTime(t_i) \quad (11)$$

$$StartTime(t_i) = \begin{cases} 0, & ,if\ t_i\ has\ no\ parents \\ \max\{EndTime(t_i) | t_j \in pre(t_i)\} & ,else \end{cases} \quad (12)$$

All the tasks that are going to be send to cloud environment will be managed using Load balancing. Tasks are assigned onto load balancing deviation used to react to fair utilization efficiency obtained by its own abilities of virtual machine resources, whose value can be obtained by Eq. (13).

$$\sigma_{LoadBalancing} = \sqrt{\frac{\sum_{j=1}^{m}\left(LB_j - \overline{LB_j}\right)^2}{m-1}} \quad (13)$$

where$\overline{LB_j}$ is the average of the load balancing factor LB_j. Finally, the fitness function is shown below.

$$Fitness = \alpha.minimize(TEC) + (1-\alpha).minimize\left(\sigma_{LoadBalancing}\right) \quad (14)$$

where α is the balance factor in a range of [0,1] which identifies the user preference for cost and load balancing.

3. The proposed optimization algorithm

The proposed algorithm help us for a speed to reach the optimization with respected to the execution cost and load balancing considering deadline and meet risk rate constraints in cloud computing. This algorithm possess of two algorithms, CPSO and Genetic algorithm. Later, we describe the PSO, CPSO and Genetic algorithm, briefly.

3.1 Particle Swarm Optimization algorithm

Kennedy and Eberhart[36] in 1995 introduced Particle swarm optimization (PSO) which refers to a population-based optimization technique PSO patterns the social behavior of birds and fish [37]. In PSO, each particle in the population is splitting into two vectors, a velocity vector and a position vector. Within each iteration, velocity and position are updated by learning from the particle's own experience best position, and yet the best position is discovered by the whole swarm [38].Position and velocity are updated by Eq. (15) and Eq. (16).

$$v_i(t) = wv_i(t-1) + c_1.r_1.\left(x_{pbest,i} - x_i(t-1)\right) + c_2.r_2.(x_{gbest} - x_i(t-1)) \quad (15)$$

$$x_i(t) = v_i(t) + x_i(t-1) \quad (16)$$

In Eq. (14), w is a parameter of PSO called the inertia weight, c_1 and c_2 are acceleration coefficients, and $r_1$and $r_2$are random numbers with uniform distribution in (0, 1), $pbest_i$ is the experienced best position of the $i-th$ particleand $gbest$ is the best position of the whole swarm so far.

3.2 Chaotic Particle Swarm Optimization algorithm

An important pint to me mentioned here is that the parameters of PSO has high impact on optimum solution efficiently. Chaotic sequence is combined with random sequence in PSO [39] to reach the high- performance and not being trapped in local minima. The process of the chaotic local search is defined as follows:

$$cx_i^{iteration} = 4cx_i^{iteration-1}\left(1 - cx_i^{iteration-1}\right), \quad i = 1,2,\dots,n. \quad (17)$$

Where $cx_i$is the ith chaotic variable, and to convert the chaotic variables to decision variables$x_i^{iteration}$using the following equation.

$$x_i^{iteration} = x_{min,i} + cx_i^{iteration}\left(x_{max,i} - x_{min,i}\right), \quad i = 1,2,\dots,n. \quad (18)$$

And the adaptive inertia weight factor depends on the optimization value of fitness calculation and is evaluated by Eq. (18).

$$w = \begin{cases} w_{min} + \dfrac{(w_{max} - w_{min})(Fitness - Fitness_{min})}{Fitness_{avg} - Fitness_{min}}, & if\ Fitness \le Fitnes \\ w_{max}, & if\ Fitness > Fitness_{avg} \end{cases}$$

3.3 Genetic Algorithm

Individuals in the current population are manipulated by Genetic algorithm and produces new individuals. Crossover and mutation are of two operators for the scheduling problems in a GA [40]. Combining individual and couple therapy is important. Crossover combines two chromosomes and produces next generation chromosomes. Mutation will produce the small variations at each chromosome [41].

3.4 The proposed optimization algorithm

In this paper, we aim to select the proper solution in a way that that optimizes the scheduling performance and minimize total execution cost and balances the load on resources while satisfying constraints within deadline and risk rate. The best feasible security is specified to make the level of user satisfaction rich. In PSO, each single solution is encoded as a particle. In workflow scheduling, each single schedule appears in a particle form and contains the tasks of application and the related candidate resources. Firstly, the tasks of the workflow are splitted for execution; secondly, the tasks should be mapped on feasible VMs from a set of available VMs. For instance, the types of available VMs that are used in this paper are shown in Fig. 2. Following this section, task scheduler and the algorithm stages are described.

VMs Properties

	CPU	*Processing capacity*	*Memory (GiB)*	*Cost per hour* ($)
VM_1	2	8800	3.75	0.105
VM_2	4	17600	7.5	0.210
VM_3	8	35200	15	0.420
VM_4	16	70400	30	0.840
VM_5	32	140800	60	0.680
VM_6	2	8800	15	0.175
VM_7	4	17600	30.5	0.350
VM_8	8	35200	61	0.700
VM_9	16	70400	122	0.400
VM_{10}	32	140800	244	0.800
VM_{11}	4	17600	30.5	0.69
VM_{12}	8	35200	61	1.38
VM_{13}	16	70400	122	2.76
VM_{14}	32	140800	244	5.52

Fig. 1. Virtual machine properties

3.4.1 Task scheduler system

In simple terms, provisioning a schedule is done on the basis of particle's position, that concerned with the available association between tasks and VMs and security certain degree of each single task. In this algorithm, each task is assigned to a VM from a virtual machine list of available resources. The pseudo-code to convert a particle's position into a schedule is illustrated in Fig. 2. Initially, we set the scheduling parameter. Then, we begin decoding the particle's position and constructing the scheduling. To meet this, we decide which task and which VM type and security levels are associated to the current particle and its value. The value at the beginning of theperforming task is calculated. As the VM is leased, based an hourly based pricing model, it may leave as much as idle time as possible. Finding a solution to this problem and minimizing execution cost, if the task t_{i-1} is the predecessor of task t_i, and processing time of task t_{i-1} has remindertime unit. Thus, the VM will be idle which can be reused by task t_i. The value of the end time of the task is computed in terms of the total processing time and the start time of the task by Eq. (11). Then, we evaluate the lease start time and lease end time of the task. And finally, we calculate the total execution cost and total execution time by Eqs. (4) and (5). After this, the algorithm can create and return the schedule associated to the given particle's position.

Task Scheduler

Begin

1. *Initialize scheduling parameters;*
2. ***For*** *each available task* t_i *in T*
3. *Compute* $StatTime(t_i)$ *by Eq. (12); // Suppose the start time of workflow is zero*
4. *Compute* $ProcessingTime(t_i)$ *by Eq. (10);*
5. *Compute* $EndTime(t_i)$ *by Eq. (11);*
6. *Compute* $LET(t_i)$ *by Eq. (6);*
7. *Compute* $LST(t_i)$ *by Eq. (7);*
8. ***End*** *for*
9. *Calculate* $TEC(t_i)$ *by Eq. (7);*
10. *Calculate* $TET(t_i)$ *by Eq. (7);*
11. *Calculate Load Balancing deviation by Eq. (13).*
12. *Return Schedule;*

End

Fig. 2. The pseudo code of workflow scheduling generation

3.4.2 The PSO-GA algorithm

At first, the initial position and velocity of population is made randomly. In each iteration, the position and velocity of all particles are updated and the chaotic sequence to the position of all particles are applied, and their fitness is computed. The number of particles, $nPop$,specifies the number of computations required to update the position and velocity of particles. The fitness function complexity is based on the schedule generation algorithm and depends on the number of tasks. Then, $pbest$ and $gbest$ are calculated. In the next step, genetic algorithm operators, crossover and mutation, are applied. In crossover, a two-point crossover is selected. Two parents and their two genes are used for crossover and then randomly selected. Then, two other solutions, by a change in resource sections of the selected genes, are created. Particles are sorted, based on the value of fitness function and extra particles up to $nPop$ deleted, and global best position is seen. You need to repeat these steps till the maximum number of iterations is created. Generally, Fig. 4 illustrates the pseudo code of this algorithm.

Before crossover

T_1	T_2	T_3	T_4	...	T_n
VM_2	VM_5	VM_{37}	VM_{41}	...	VM_5

T_1	T_2	T_3	T_4	...	T_n
VM_8	VM_9	VM_{66}	VM_{85}	...	VM_3

After crossover

T_1	T_2	T_3	T_4	...	T_n
VM_2	VM_9	VM_{66}	VM_{41}	...	VM_5

T_1	T_2	T_3	T_4	...	T_n
VM_8	VM_5	VM_{37}	VM_{85}	...	VM_3

Fig. 3. Crossover operation method in proposed algorithm

Pseudo code of proposed algorithm using CPSO and Genetic Algorithm

```
Begin
1.  Set the number of particles to nPop;
2.  Set the maximum number of iteration maxIt = 100;
3.  Set the personal best and global best pbest = 0; gbest = 0;
4.  Set the parameters of CPSO w, c1 and c2;
5.  Set the parameters of Genetic Algorithm crossover_Rate and mutation_Rate;
6.  Initialize the population of particles with random positions and velocities;
7.  For each particle i=1 to nPop
8.      Randomly initialize x_i and v_i ;
9.      x_pbest,i = x_i;
10.     End if
11. End For
12. j = 0;    // number of iteration
13. while the maximum number of iterations has not been reached // j < maxIt
14. For each particle i=1 to nPop
15.     Evaluate each particle's fitness function by Eq. (14);
16.     Update the position and velocity of the particle by Eqs. (15) and (16);
17.     Apple the chaotic sequence to position of the particle by Eq. (17);
18.     Keep particle within the search space based on its boundaries;
19.     if Fitness_ij < pbest_ij then
20.         x_pbest,i = x_ij;
21.         pbest_ij = Fitness_ij;
22.     End if
23.     if Fitness_ij < gbest then
24.         x_gbest = x_ij;
25.         gbest = Fitness_ij;
26.     End if
27. End for
28. For each i=1 to nCrossover   //(nCrossover = Round((pCrossover * nPop)/2) * 2 )
29.     Apply Crossover operator;
30. End for
31. For each i=1 to nMutation    //(nMutation = pMutation * nPop )
32.     Apply Mutation operator;
33. End for
34. Sort population with the value of fitness function;
35. Delete extra particles up to nPop;
36. if pbest_0 < gbest then
37.     gbest = pbest_0
38. End if
39. Increment the number of iteration j
40. End while
End
```

Fig. 4. The pseudo code of proposed algorithm

After this step, the algorithm will be able to establish and return the schedule associated to the given particle's position. Finally, Algorithms 1 and 2 are combined to produce a near optimal scheduling. In step 15 of Algorithm 2, instead of calculating the fitness value of the particle, we provide the scheduling as outlined in Algorithm 1.

4. Performance evaluation

4.1 Environment

We apply our proposed algorithm in visual studio C#.net on a Windows 8 desktop PC equipped with Intel Core i7 CPU. In the experiments, the performance of the proposed algorithm is evaluated using a CPSO proposed by Li et al. in [20] and genetic algorithm. Four scientific workflows, namely LIGO, SIPHT, CyberShake and montages are employed to run this algorithm. The laser interferometer gravitational wave observatory (LIGO) found as a project to reveal gravitational waves through the network of gravitational-wave detectors [42]. The sRNA identification protocol using high-throughput technology (SIPHT) program uses a workflow to automate the search for sRNA encoding-genes [43]. The CyberShake workflow is used to characterize earthquake hazards in a region[43]. Montage consists of an image application that creates mosaics of the sky in astronomy research [42].

The experiments are implemented with 14 VMs and 100 tasks. In addition, suppose that the workload of each task is in the range [5000, 50000] MFLOPS and the output size is in the range [10, 100] GB only for the experiments purposes, and the bandwidth among VMs is constant, 0.1 GB/s. Also, suppose there are many alternative security methods or algorithms provided for users to figure out the authentication service, integrity service and confidentiality service, as illusterated in Tables 2-4. The parameters used are those listed in Table 4. We have set other parameters considering [12] and [44]. The accomplished experiments evaluate the workflow scheduling cost with the deadline, risk rate constraints, and load balance deviation. To examine the execution cost of the proposed algorithm under the different numbers of iterations, particles and tasks, in this section, we scale the task sizes from 25 to 100 while the iterations are from 100 to 300 and particle sizes are 40. The results of comparisons between different algorithms to improve execution cost are shown in Figs. 5 and 6. The results demonstrate that CPSO-GA minimizes total execution cost. Also, consider load balancing and achieve security.

According to our findings, each single task will be assigned to suitable VM type, which can meet the deadline and risk rate constraints.In this algorithm, security levels of all tasks are equal to 1. Thus, the risk rate of each workflow is always 0. If all tasks are without security services, the risk rate of each workflow is always 1. The aim of the proposed algorithm is to decrease the total workflow execution cost while meeting the deadline and risk rate constraints and load balance deviation. This section presents the comparative evaluation CPSO-GA with two algorithms, CPSO, GA and demonstrates and evaluates the execution cost and load balancing.

Table 4. Simulation Parameters

Parameter	**Value**
Number of tasks in application	25~100
Number of VMs	14
Number of population	40
Number of iteration	100~300
The bandwidth among resources	0.1
The workload of each task(MFLOPS)	5000~50000
The output sizeof each task (GB)	10~100

4.2 The fitness evaluation

Of course, the results of proposed algorithm and other algorithms are involved in this section. The distinction between them is almost straightworward. We utilize 100 tasks as the number of iterations are 100 and 300. The comparative evaluationof three algorithmss on various workflow for iteration 100 and 300 is demonstrate in Table 4 and 5. Also, the fitness between the three algorithms for large size of workflows, Montage, LIGO, SIPHT and CyberShake are compared in figure 6 and 7. The obtained results show that fitness value of proposed algorithm is lower than CPSO and GA. Therefore, CPSO-GA is announced to be better than other algoritms.

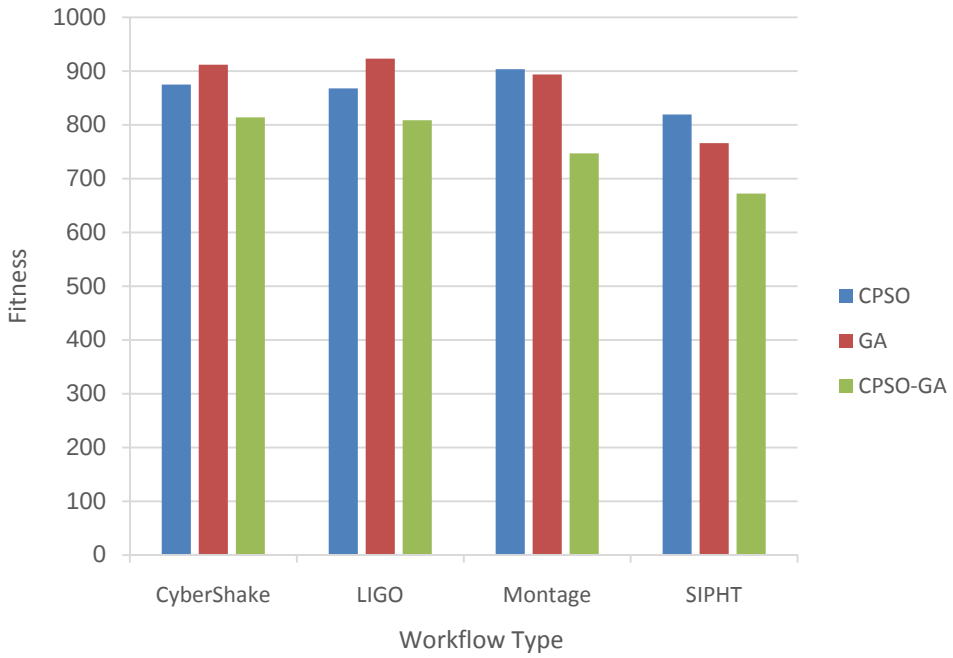


Fig. 6. The result of three algorithms for iteration 100 on various workflows

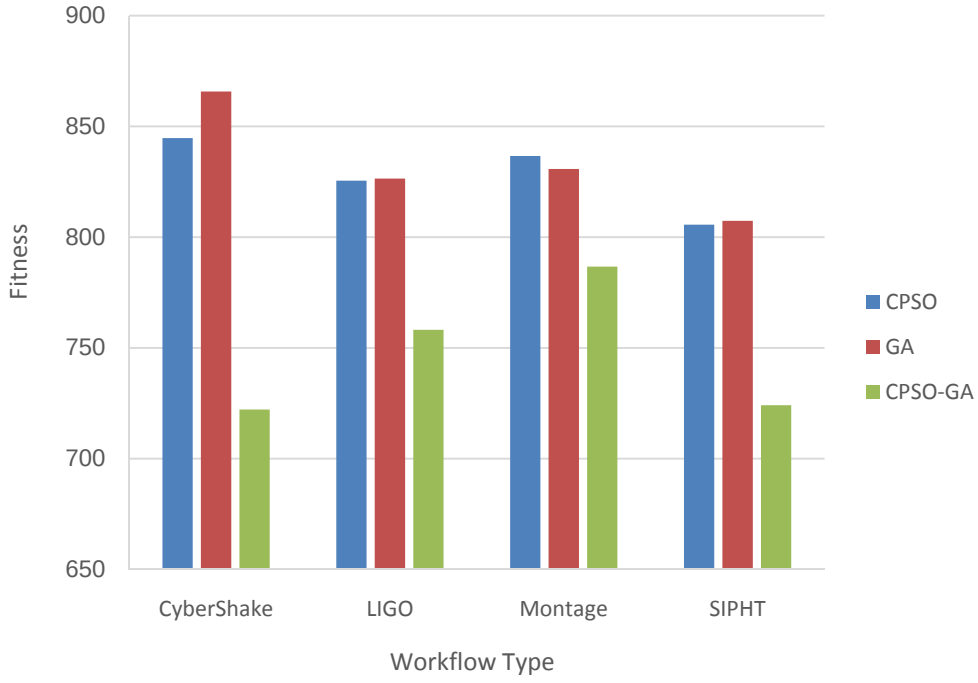


Fig. 7. The result of three algorithms for iteration 300 on various workflows

Table 4. Comparison of fitness on various workflow for iteration 100

Iteration / workflow & algorithm		1	20	40	60	80	100
Cyber Shake	CPSO	942.12	930.17	916.61	874.87	874.87	874.87
	GA	1013.21	911.96	911.96	911.96	911.96	911.96
	CPSO-GA	904.81	894.18	839.58	839.58	814.02	814.02
LIGO	CPSO	939.72	925.70	923.36	867.79	867.79	867.79
	GA	1160.11	1017.79	923.18	923.18	923.18	923.18
	CPSO-GA	1043.87	929.41	905.30	905.30	808.60	808.60
Montage	CPSO	953.31	913.14	913.14	913.14	913.14	903.54
	GA	990.09	934.05	934.05	917.72	893.64	893.64
	CPSO-GA	1015.81	829.72	829.72	747.06	747.06	747.06
SIPHT	CPSO	996.53	848.48	848.48	819.23	819.23	819.23
	GA	999.03	922.50	893.50	765.94	765.94	765.94
	CPSO-GA	923.75	858.12	858.12	672.34	672.34	672.34

Table 5. Comparison of fitness on various workflow for iteration 300

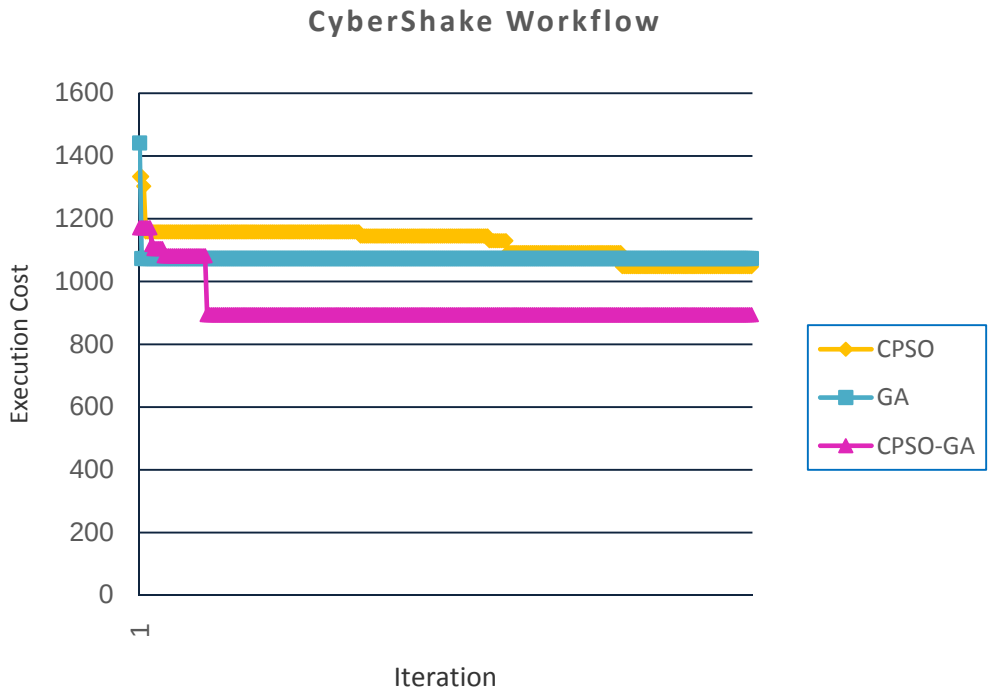
Iteration / workflow & algorithm		1	50	100	150	200	250	300
Cyber Shake	CPSO	1074.60	933.83	933.83	923.00	923.00	844.74	844.74
	GA	1159.47	865.78	865.78	865.78	865.78	865.78	865.78
	CPSO-GA	944.07	722.18	722.18	722.18	722.18	722.18	722.18
LIGO	CPSO	1005.80	925.70	923.36	867.79	867.79	867.79	867.79
	GA	1160.11	951.88	825.50	825.50	825.50	825.50	825.50
	CPSO-GA	1092.56	826.48	826.48	826.48	826.48	826.48	826.48
Montage	CPSO	1119.73	941.20	855.24	845.95	845.79	845.79	836.62
	GA	1016.53	863.79	863.79	863.79	830.76	830.76	830.76
	CPSO-GA	995.44	995.44	818.74	786.70	786.70	786.70	786.70
SIPHT	CPSO	926.63	805.66	805.66	805.66	805.66	805.66	805.66
	GA	906.44	834.61	834.61	834.61	834.61	818.86	807.36
	CPSO-GA	973.66	770.39	770.39	724.16	724.16	724.16	724.16

4.3 The execution cost evaluation

Due to the importance of this paper, it will be mentioned that the proposed algorithm attempts to minimize the total workflow execution cost as meeting the deadline and risk rate constraints.

Fig. 8 shows the comparison of cost for CPSO, GA and our proposed algorithms when the number of iterations are 300 on different workflows

Under the same parameter setting, the LIGO workflow has the most cost, the Montage workflow has a moderate level of cost and SIPHT and Cyber Shake workflow show the lowest execution cost. Therefore, our proposed approach can decrease the total execution cost more than CPSO and GA algorithms.


CyberShake Workflow
1600
1400
1200
1000
800
600
400
200
0
Execution Cost
1
Iteration
CPSO
GA
CPSO-GA

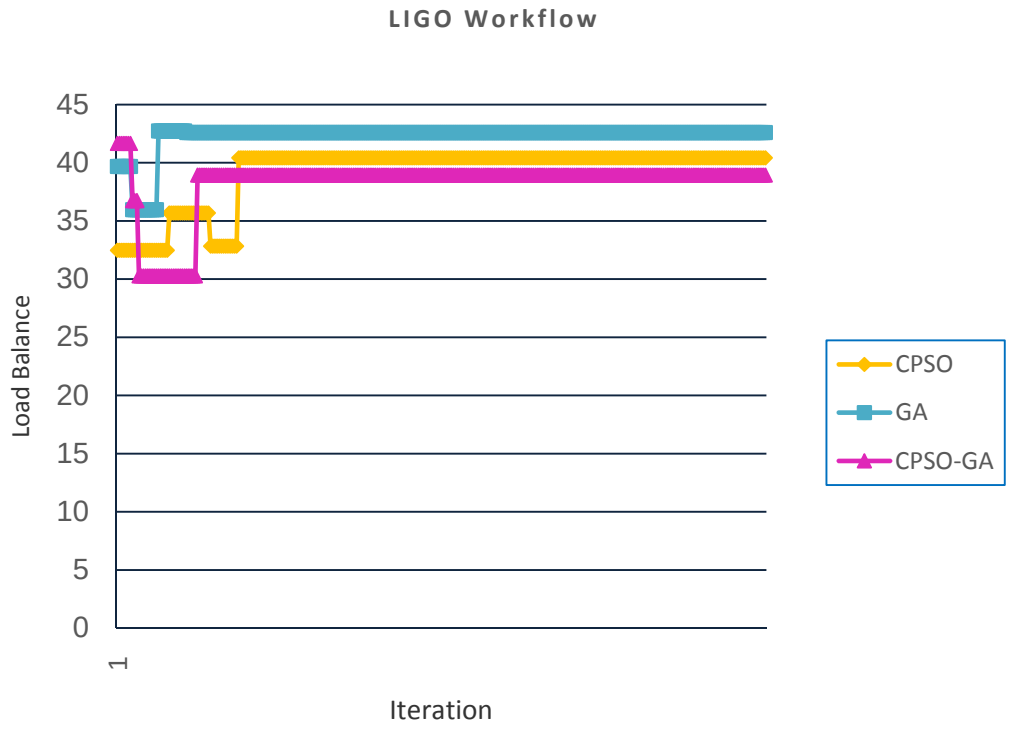

LIGO Workflow
1400
1200
1000
800
600
400
200
0
Execution Cost
1
50
100
150
200
250
300
Iteration
CPSO
GA
CPSO-GA

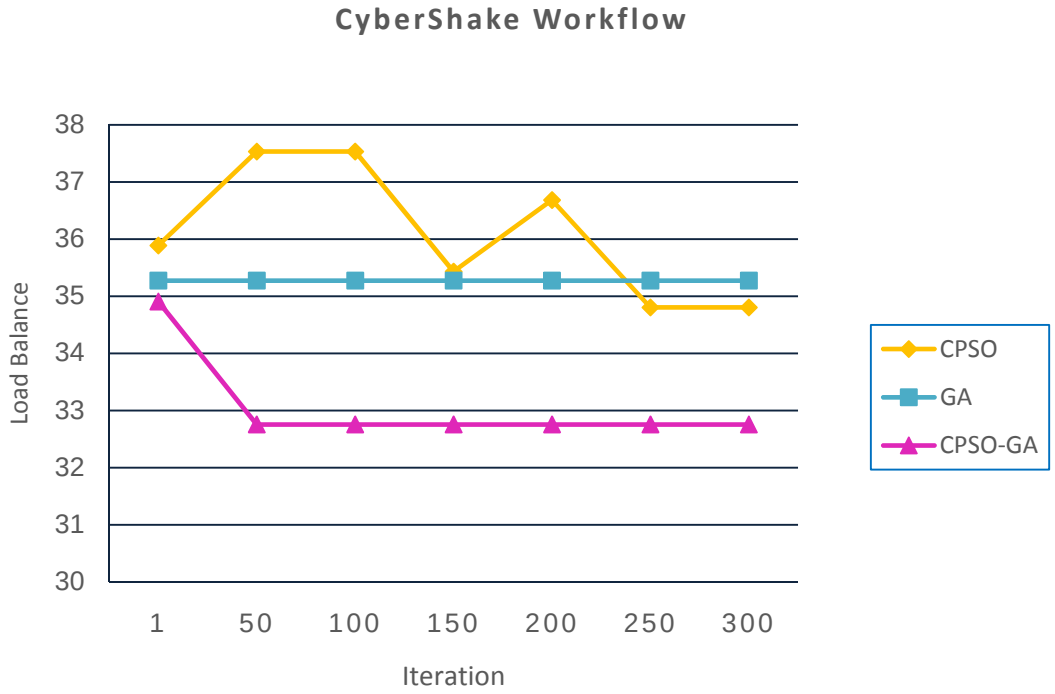
Fig. 8. Comparison of execution cost for iteration 300 on various workflows

4.4 The load balancing evaluation

As discussed before, load balance deviation is a performance metric that will be considered here. The load balancing mechanism and developing suitable task mappings are two vital issues that need to be handled to propel.

The comparison of load balance deviation for three algorithms, CPSO, GA is shown in Fig. 9 and the proposed algorithm when the number of iterations are 300 on different workflows. Under the same parameter setting, the Cyber-Shake workflow has the lowest value, the Montage workflow has a moderate level of load balance value and SIPHT and LIGO workflow show the most value for load balance.


LIGO Workflow
Load Balance
45
40
35
30
25
20
15
10
5
0
1
Iteration
CPSO
GA
CPSO-GA


CyberShake Workflow
Load Balance
38
37
36
35
34
33
32
31
30
1
50
100
150
200
250
300
Iteration
CPSO
GA
CPSO-GA

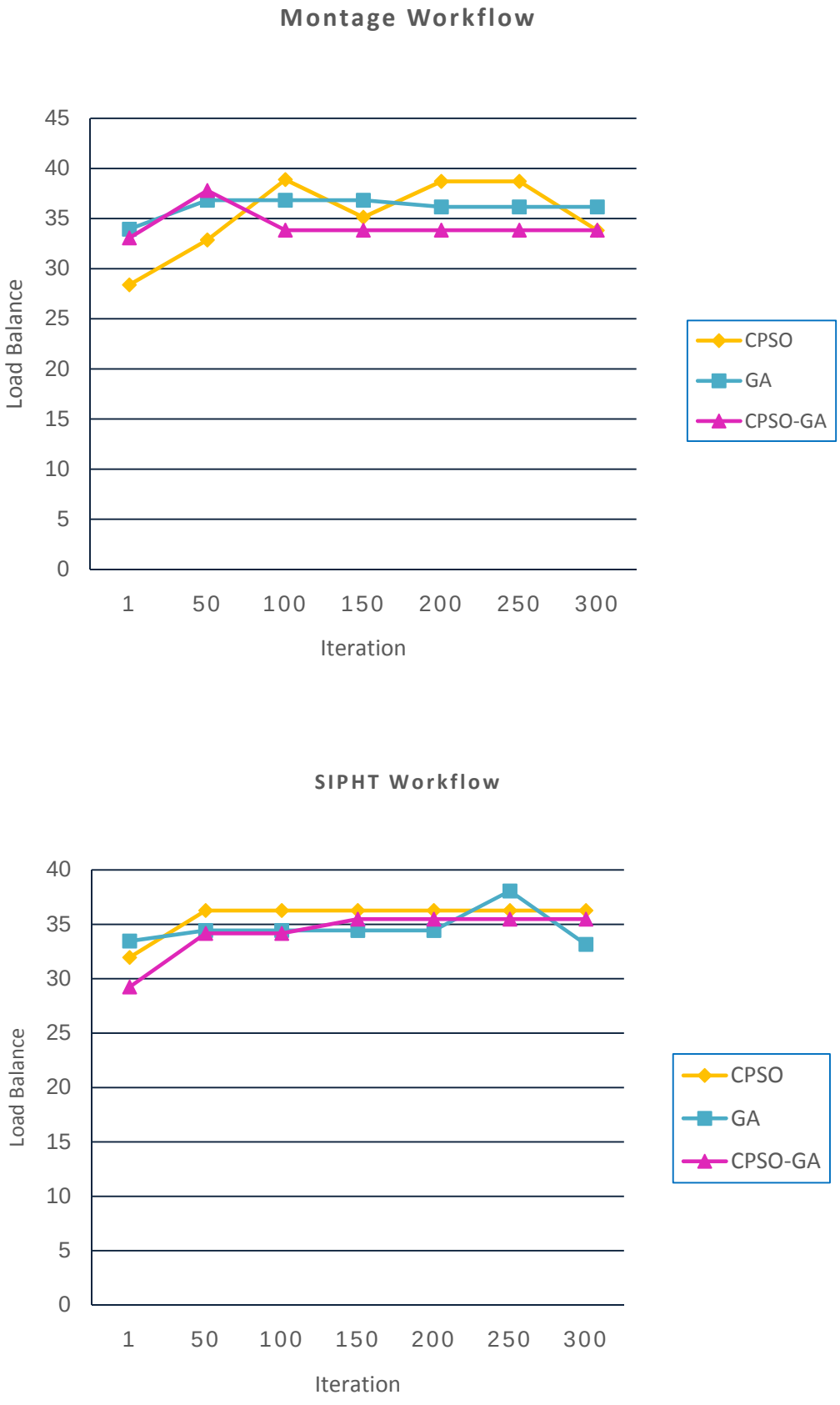


Fig. 9. Comparison of load balance deviation for iteration 300 on various workflows

5. Conclution

For a research to display dependable results, it must possess certain attributes. In this paper the desired attribute referred to a hybrid meta-heuristic scheduling algorithm. optimizing the performance of the schedule , minimizing execution cost and load balance deviation can be dealt with using proposed algorithm. Following the procedure on this algorithm, setting certain solutions for all user supper Qos constraints has been possible. In order to consolidate the validity and the security of the process, CPSO and GA algorithms are utilized.The proposed algorithm is evaluated in the case of large scientific workflows.By obtaining results through such instruments many important discoveries have been made. To investigate the validity, truthfulness, or fallacy of the proposed procedure, the comparison of this algorithm was established with CPSO and GA algorithms under the same QoS constraint and pricing model.This dose imply that CPSO has tendency to result in a much faster convergence speed than PSO for different dimensions of the input data in the search space.Considering the above-mentioned obtaining experimental results, it can be concluded that the performance of the proposed algorithm is much better than CPSO and GA algorithms. For other existing work, we canbe interested in schedules that minimize the overall budget. That is, both makespan and reliability can be minimized as well. In order to provide the reader with the trend in the acquisition of the method, appealing to more efficient procedures of optimization to solve the task-resource scheduling problem is going to be useful.

References

[1] Singh, L.; Singh, S. (2013): A survey of workflow scheduling algorithms and research issues.
[2] Kaur, N., Aulakh, T.S.;Cheema, R.S. (2011): Comparison of workflow scheduling algorithms in cloud computing. International Journal of Advanced Computer Science and Applications, 2(10).
[3] Masdari, M. (2016): Towards workflow scheduling in cloud computing: a comprehensive analysis. Journal of Network and Computer Applications, 66: p. 64-82.
[4] Jang, S.H. (2012): The study of genetic algorithm-based task scheduling for cloud computing. International Journal of Control and Automation, 5(4): p. 157-162.
[5] Kaur, R.;Ghumman, N.: Hybrid Improved Max Min Ant Algorithm for Load Balancing in Cloud.
[6] Kansal, N.J.;Chana, I. (2012): Cloud load balancing techniques: A step towards green computing. IJCSI International Journal of Computer Science Issues, 9(1): p. 238-246.
[7] Shimpy, E.; Sidhu, M.J. (2014): Different scheduling algorithms in different cloud environment. International Journal of Advanced Research in Computer and Communication Engineering, 3(9).
[8] Agarwal, D. and S. Jain, Efficient optimal algorithm of task scheduling in cloud computing environment. arXiv preprint arXiv:1404.2076, 2014.
[9] Bala, A.; Chana. I. (2011): A survey of various workflow scheduling algorithms in cloud environment. in 2nd National Conference on Information and Communication Technology (NCICT).
[10] Chawla, Y.; Bhonsle, M. (2012): A study on scheduling methods in cloud computing. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 1(3): p. 12-17.
[11] Abrishami, S.; Naghibzadeh, M. (2012): Deadline-constrained workflow scheduling in software as a service cloud. ScientiaIranica, 19(3): p. 680-689.
[12] Li, Z. (2016): A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds. Future Generation Computer Systems.
[13] Li, W. (2012): Trust-based and QoS Demand Clustering Analysis Customizable Cloud Workflow Scheduling Strategies. in Cluster Computing Workshops (CLUSTER WORKSHOPS), IEEE International Conference on. 2012. IEEE.
[14] Sumathi, D.;Poongodi, P. (2015): An Improved Scheduling Strategy in Cloud Using Trust Based Mechanism.
[15] 15. Marcon, D.S. (2013): Trust-based grouping for cloud datacenters: improving security in shared infrastructures. in IFIP Networking Conference, IEEE.
[16] Jianfang, C.; Junjie, C. (2014): and Z. Qingshan, An optimized scheduling algorithm on a cloud workflow using a discrete particle swarm. Cybernetics and Information Technologies, 14(1): p. 25-39.
[17] Kumar, S.S.;Balasubramanie, P. (2012): Dynamic scheduling for cloud reliability using transportation problem. Journal of Computer Science, 8(10): p. 1615.
[18] Liu, H., et al., Swarm scheduling approaches for work-flow applications with security constraints in distributed data-intensive computing environments. Information Sciences, 2012. 192: p. 228-243.
[19] Wang, X., (2011): Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Future Generation Computer Systems, 27(8): p. 1124-1134.
[20] Li, X.,;Xu, J. (2015): and Y. Yang, A chaotic particle swarm optimization-based heuristic for market-oriented task-level scheduling in cloud workflow systems. Computational intelligence and neuroscience, p. 81.
[21] Wu, Z. (2010): A revised discrete particle swarm optimization for cloud workflow scheduling. in Computational Intelligence and Security (CIS), 2010 International Conference on. IEEE.
[22] Yang, Y.;Peng, X. (2015): and J. Cao, Trust-Based Scheduling Strategy for Cloud Workflow Applications. Informatica, 26(1): p. 159-180.
[23] Singh, L; Singh. S. (2014): A Genetic Algorithm for Scheduling Workflow Applications in Unreliable Cloud Environment. in International Conference on Security in Computer Networks and Distributed Systems, Springer.
[24] Marcon, D.S. (2013):. Workflow specification and scheduling with security constraints in hybrid clouds. in Cloud Computing and Communications (LatinCloud), 2nd IEEE Latin American Conference on, IEEE.
[25] Zeng, L.; Veeravalli, B.; Li, X. (2015): SABA: A security-aware and budget-aware workflow scheduling strategy in clouds. Journal of Parallel and Distributed Computing, 75: p. 141-151.
[26] Awad, A.; El-Hefnawy, B. (2015): and H. Abdel_kader, Enhanced Particle Swarm Optimization for Task Scheduling in Cloud Computing Environments. Procedia Computer Science, 65: p. 920-929.
[27] Delavar, A.G.; Aryan, Y. (2012): A goal-oriented workflow scheduling in heterogeneous distributed systems. Int J ComputAppl, 52(8): p. 27-33.
[28] Chen, C. (2015): A Hybrid Genetic Algorithm for Privacy and Cost Aware Scheduling of Data Intensive Workflow in Cloud, in Algorithms and Architectures for Parallel Processing, Springer. p. 578-591.

[29] Li, X. (2014): A New Particle Swarm Optimization-Based Strategy for Cost-Effective Data Placement in Scientific Cloud Workflows, in Future Information Technology, Springer. p. 115-120.
[30] Sridhar, M.; Gabu. G. (2015): Hybrid Particle Swarm Optimization scheduling for cloud computing. in Advance Computing Conference (IACC), IEEE International.
[31] Yao, G. (2016): Endocrine-based coevolutionary multi-swarm for multi-objective workflow scheduling in a cloud system. Soft Computing, p. 1-14.
[32] Wang, X. (2015): Scheduling Budget Constrained Cloud Workflows With Particle Swarm Optimization. in 2015 IEEE Conference on Collaboration and Internet Computing (CIC), IEEE.
[33] Verma, A.;Kaushal, S. (2015): Cost Minimized PSO based Workflow Scheduling Plan for Cloud Computing.
[34] Jafarzadeh-Shirazi, O.; Dastghaibyfard, G.; Raja, M.M. (2015): TASK SCHEDULING WITH FIREFLY ALGORITHM IN CLOUD COMPUTING. Science International, 27(1).
[35] Wu, F. (2015): Unified Multi-constraint and Multi-objective Workflow Scheduling for Cloud System. in International Conference on Algorithms and Architectures for Parallel Processing. Springer.
[36] Kaur, T.;Pahwa, S. (2013): An Upgraded Algorithm of Resource Scheduling using PSO and SA in Cloud Computing. International Journal of Computer Applications, 74(8).
[37] Salman, A.; Ahmad, I.; Al-Madani, S. (2002): Particle swarm optimization for task assignment problem. Microprocessors and Microsystems, 26(8): p. 363-371.
[38] Chen, W.-N. and J. Zhang. A set-based discrete PSO for cloud workflow scheduling with user-defined QoS constraints. in Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on. 2012. IEEE.
[39] Liu, B. (2005): Improved particle swarm optimization combined with chaos. Chaos, Solitons& Fractals, 25(5): p. 1261-1271.
[40] Yu, J.;Buyya, R. (2006): Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms. Scientific Programming, 14(3-4): p. 217-230.
[41] Ravichandran, S.;Naganathan, D.E. (2013): Dynamic Scheduling of Data Using Genetic Algorithm in Cloud Computing. International Journal of Computing Algorithm, 2(01): p. 127-133.
[42] Bittencourt, L.F.; Madeira,E.R.;Da-FonsecaN.L. (2012): Scheduling in hybrid clouds. Communications Magazine, IEEE, 50(9): p. 42-47.
[43] Bharathi, S. (2008): Characterization of scientific workflows. in Workflows in Support of Large-Scale Science, Third Workshop on. 2008. IEEE.
[44] Xie, T.; Qin, X. (2006): Scheduling security-critical real-time applications on clusters. IEEE transactions on computers, 55(7): p. 864-879.