ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

Profit Maximization in Hybrid IaaS Cloud

KAVITHA B

Lecturer, Department of Electronics and Communication Engineering, IRT Polytechnic College, Chennai, Tamil Nadu, India

ABSTRACT: The users without sufficient computing resources habitually enjoy the Infrastructure as a Service (Iaas) from the cloud providers. The IaaS provider, in turn, achieves immense financial gains when the demand by the clients reaches a summit level. In the related scenarios, the task scheduling emerges a significant daunting challenge for the provider to offer the Quality of Service (QoS). The earlier-launched cloud federation model to tackle the related challenge is not found to be viable, in actual practice. Hence, meta-heuristic algorithms such as Self-adaptive Learning Particle Swarm Optimization (SLPSO) and Cuckoo Search (CS) are envisioned to perk up the profit of IaaS provider with assured QoS devoid of any inter cloud agreement. In this work, the Self-adaptive Learning Particle Swarm Optimization and Cuckoo Search based scheduling technique are brought in for the purpose of scheduling. The vital challenge, here, is concerned with allotment of various user tasks to incredibly enhance the income of Infrastructure as a Service (IaaS) provider concurrently ensuring the Quality-of-Service (QoS). The improved outcomes originating from several test investigations vie with each other is establishing the fact that the SLPSO algorithm significantly augments the IaaS provider gains vis-à-vis the benefits offered by Cuckoo Search algorithm. It is also established from the test results that these optimization algorithms are further suitable in the hybrid cloud scenario.

***KEYWORDS*:** Scheduling, Self-adaptive Learning Particle Swarm Optimization, Cuckoo Search, IaaS provider, Hybrid Cloud, Quality-of-Service.

I. INTRODUCTION

With easy and nominal management endeavors, the cloud computing represents a growing technique which accesses the network and shares computing resources. With the intent to enhance its gains, three diverse cloud computing patterns such as Software as a Service (SaaS), Infrastructure as a Service (IaaS), and Platform as a Service (PaaS) are envisaged [1].

The Cloud computing is intended to maximize the utilization of allotted resources, and combine them so as to effectively deal with the titanic computation challenges [2]. It perfectly provides the computing services for the users as public utility, which is easily available to all the entities [3]. On the basis of contracts, the services are offered by the providers to the subscribers, and a charge is levied based on the services actually utilized. The client has the facility of paying for the availed service by means of the payment mechanism "pay as you go" model [4].

In the case of a cloud provider, a simple solution is to make the excess purchase of the resources well-ahead anticipating greater demand, and it tends to be cost-prohibitive. [6]. In this regard, a ray of hope is offered by a novel solution named as the Cloud Federation [5] which facilitates the providers to make deals in their sources by means of the federation agreement. Conversely, the relative federation is very difficult to achieve currently, on account of the dearth of inter-operation criteria and players' motivation to federate [7].

Zuo *et al* [11] significantly streamlined a cloud resource allotment structure to enable the utilization of the external clouds with the intent to tailor the IaaS cloud itself as adaptable. In their novel design, an IaaS cloud contained its unique private cloud, and outsourced its tasks to the cloud providers known as the external clouds (ECs) when its local resources reached a level of insufficiency.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

The PSO shines with gleaming gains of simple achievement and swift convergence, in order that the relative scheduling method is well-geared to achieve an optimal or suboptimal solution in a reduced computational time-frame for solving the titanic issues. Conversely, the regular PSO merely ensnares into local optima and does not appear to be robust for the severe problem scenarios. It was Zuo *et al* [11] who pressed the button of green light on the SLPSO technique by envisioning several learning stratagems to overshoot the malaise of the regular PSO. Even though, it guaranteed added profit with assured QoS, it resulted in an excessive expenditure in the choice of velocity approach.
The remainder of the document is orchestrated as follows. Section 2 highlights the literature work of the most modern investigation efforts. Section 3 offers a 360-degree view of the structure of the resources allotment in the hybrid cloud scenario. The SLPSO and CS based scheduling techniques are discussed in Section 4. The test outcomes lend charisma to Section 5 and the customary conclusion of the document is covered in the concluding Section 6.

II. REVIEW OF RELATED WORKS

A QoS-based workflow scheduling technique was amazingly launched by S. Abrishami and M. Naghibzadeh [12] taking valuable cues from the concept known as the Partial Critical Paths (PCP). In the cloud computing, the task scheduling optimizing approach was kick-started by Lizheng Guo *et al.* [13].
An integer programming model was excellently designed by Xingquan Zuo *et al.* [11] for the resources allotment problem of an IasS cloud in a hybrid cloud scenario. A self-adaptive learning PSO (SLPSO)-based scheduling technique for tackling the related problem was effectively envisioned. Here, the four category of velocity modernize technique was very complicated and resulted in time complication.

III. SOLUTION FRAMEWORK AND PROBLEM DEFINTION

In the current investigation, for the purpose of generating an infrastructure as a service (IaaS) cloud which is it adaptable, a cloud resource allotment structure is envisioned to facilitate the deployment of the external clouds (ECs). Here, the IaaS cloud is endowed with its own unique private cloud and is competent to outsource its assignment to the external cloud providers, known as the external clouds (ECs) when its local resources are not adequate. In the innovative structure, the ECs furnish a public interface for generating and administering the VM instances within their proprietary infrastructure. In this regard, the private clouds and external clouds emerge as the crucial segments of the related hybrid design.

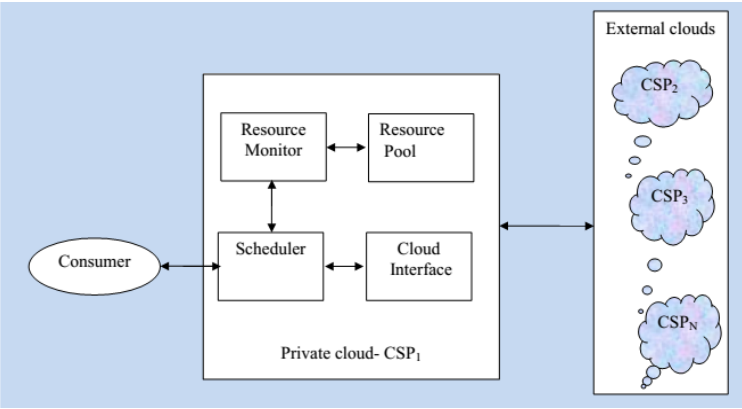

Figure 1: Overall structure of hybrid task scheduling

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

A problem may be defined as follows. Let $CSP = \{CSP_1, CSP_2, ..., CSP_n\}$ represent a set of cloud providers and CSP_1 symbolize the private cloud and $CSP_2, ..., CSP_n$ correspond to the External clouds (ECs). $V^M = \{V_1^M, V_2^M, ..., V_I^M\}$ denotes a set of virtual machines (VM) categories and $B = \{b_1, b_2, ..., b_m\}$ constitutes a set of applications. Each application $b_i (i \in \{1,2,...,m\})$ contains a harsh deadline d^i and runtime r^i, and encompasses a task set $Task_i = \{t_{i1}, t_{i2}, ..., t_{iTi}\}$.

By bringing in time slots with a granularity of 1 hr time is explicitly characterized in the Integer Programming (IP) model. Let S be the maximum number of time slots, we have $S = \max_{i \in \{1,2,...,m\}}(d^i)$ the underlying objective is to allot the m applications to $CSP_k (k \in \{1,2,...,n\})$ to maximize the profit of CSP_1. Each task has to be allotted to a $CSP_k (k \in \{1,2,...,n\})$. When a task begins to function, it should not ever be interrupted in order that its running slots are successive. In any time slot $s (s \in \{1,2,...,S\})$, resources utilized by the entire tasks performed in CSP_1 should not ever exceed the total resources of CSP_1 and from the perspective of CSP_1, all its ECs possess infinite sources.

The problem may be designed as illustrated in the ensuing IP model.

Maximize

$$\text{Profit} = \sum_{i=1}^{m}\sum_{u=1}^{I} T_i f_{iu} PC_u r^i - \sum_{i=1}^{m}\sum_{l=1}^{T_i}\sum_{u=1}^{I}\sum_{k=1}^{n} g_{ilk} f_{iu} CT_{ku} r^i \quad (1)$$

Subject to,

$$\sum_{k=1}^{n} g_{ilk} = 1, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\} \quad (2)$$

$$\sum_{s=1}^{d^i} x_{ils} = g_{il1} r^i, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\} \quad (3)$$

$$st_{il} \geq 1, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\} \quad (4)$$

$$st_{il} \leq d^i - r^i + 1, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\} \quad (5)$$

$$(s \leq st_{il} - 1) \vee (s \geq d^i - r^i) \vee ((s \geq st_{il}) \wedge (s \leq st_{il} + r^i - 1) \wedge (x_{ils} = g_{il1})) \quad (6)$$

$$\forall s \in \{1,2,..., d^i\}, i \in \{1,2,..., m\}, l \in \{1,2,...., T_i\}$$

$$\sum_{i=1}^{n}\sum_{l=1}^{T_i}\sum_{u=1}^{I} x_{ils} f_{iu} CPU_u \leq total\_CPU, \quad \forall s \in (1,2,..., S) \quad (7)$$

$$\sum_{i=1}^{n}\sum_{l=1}^{T_i}\sum_{u=1}^{I} x_{ils} f_{iu} MEM_u \leq total\_MEM, \quad \forall s \in (1,2,..., S) \quad (8)$$

$$g_{ilk} \in \{0,1\}, \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\}, k \in \{1,2,...., n\} \quad (9)$$

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

$$x_{ils} \in \{0,1\}, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\}, s \in \{1,2,...., S\} \tag{10}$$

$$st_{il} \in \{1,2,...S\}, \quad \forall i \in \{1,2,..., m\}, l \in \{1,2,..., T_i\} \tag{11}$$

Of late, many an investigator has carried out several researches by employing diverse appraisal algorithms like the particle swarm optimization, genetic algorithm and the cuckoo search. From among them, the PSO is equipped with inherent skills to locate the most buoyant outcome. In the meantime, often, it is adversely affected by several defects involving the local optimum and the minimal convergence rate. Nevertheless, of late, various changes have been effected in the PSO to augment the performance of task scheduling procedure. In [11], the author has envisaged a resource allotment structure employing the self-adaptive learning particle swarm optimization (SLPSO) in which an IaaS provider is competent to outsource its tasks to External Clouds. In the paper, PSO is presented for the enhancement in performance.

IV. CS AND SLPSO BASED RESOURCE ALLOCATION APPROACH

An innovative resource allotment method is effectively envisioned to overwhelm the scheduling problem with the help of the Self-adaptive Learning Particle Swarm Optimization (SLPSO) and Cuckoo Search (CS) in this section. The Particle swarm optimization constitutes a population based optimization approach [7] performed for several complicated optimization problems [8, 9] inviting superior outcomes.

4.1. Standard PSO

Kennedy and Eberhart [8] [9] gained considerable name and esteem by effectively kick-starting a universal optimization approach known as the Particle Swarm Optimization (PSO) algorithm. It represents a swarm intelligence [10] algorithm which replicated the swarm character like the flocking of birds and the schooling of the fish. Conversely, the PSO is in the hunt for an optimum through each particle flying in the search space and varying its flying trajectory based on its unique best experience together with that of the neighborhood instead by means of particles experiencing genetic functions such as the selection, crossover, and mutation [14].
In the standard PSO, each individual in the swarm is deemed as a particle in a D-dimensional search space, and signified by a three tuple $\{X_i, V_i, P_i\}$. $X_i = (x_{i1}, x_{i2},..., x_{iD})$ and $V_i = (v_{i1}, v_{i2},..., v_{iD})$ indicate the position and velocity of particle i, correspondingly. $P_i = (p_{i1}, p_{i2},..., p_{iD})$ Signifies the personal best (pbest) of particle i (that is, the best position achieved by particle i), $G = (g_1, g_2,..., g_D)$ indicates the global best (gbest), that is, the best position followed by the whole swarm. The value of each element in the vector V_i can be closed to the range of $[-v_{max}, v_{max}]$ to regulate the needless roaming of particle outside the search space, and revised by [9], as

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 [x_{id}(t) - p_{id}(t)] + c_2 r_2 [x_{id}(t) - g_d(t)] \tag{12}$$

Where, $i = 1,2,..., M$ indicates the number of particles and $d = 1,2,..., D$ denotes the dimension of particles. r_1 and r_2 are uniformly apportioned arbitrary numbers in the range of is [0, 1]. c_1 and c_2 represent the learning factors [0.1] and ω characterizes the inertia weight [0.1] essential to keep aloof from unhindered enhancement of the velocity of the particle.
The particle flies in the direction of a new position as illustrated in the following Equation (13), and each value of the new position [9] has to be well-within the range of $[\min X, \max X]$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{13}$$

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

At the outset, the position and velocity of each particle in the swarm are randomly activated. Subsequently, each particle i is directed by its own flying practice (pbest) and the best particle (gbest), as modernized by (12) and (13). The process goes on until the reach of a user-defined stopping standard.
Beautifully depicted below is the roadmap with the diverse phases of the normal PSO.

Step 1: *Arbitrarily activate the position and velocity of the entire particles.*

Step 2: *Estimate the fitness values of the entire particles, with t each particle's pbest as its current position and gbest as the best one among the entire particles.*

Step 3: *Modernize the velocity and position of each particle employing (12) and (13).*

Step 4: *Compute the fitness values of the entire particles.*

Step 5: *Modernize the pbest. For each particle, if the fitness value of its new position is superior to that of its pbest, then substitute its pbest by the new position.*

Step 6: *Modernize gbest. For each particle, if the fitness value of its new position is superior to that of the gbest, then substitute the gbest by the new position.*

Step 7: *If the stopping standard is satisfied, output the gbest and its fitness value. Or else, return to Step 3.*

4.2 SLPSO based Scheduling

Based on the work in [15], in this paper a Self-adaptive Learning PSO-based scheduling approach is proposed to solve the DCTS problem. Four modified velocity updating strategies are used to improve the global search capability and robustness.

4.2.1 ***Velocity Updating Strategies***

a) Strategy 1: The first strategy used is the difference based velocity updating strategy (DbV) [15] that can avoid progressive velocity changing. This strategy makes each particle explore in a wider range by updating the velocity based on the difference information of particles, and the velocity is updated by

$$v_{id}(t) = x_{kd}(t) - x_{jd}(t) \quad (14)$$

$$v_{id}(t+1) = cv_{id}(t) + c[p_{id}(t) - x_{id}(t)] \quad (15)$$

where c is a standard normally distributed random number in the range of [0, 1]; $x_{kd}(t)$ *and* $x_{jd}(t)$ are the dth variables of two randomly selected particles from the swarm in the tth generation, respectively. Equation (14) is used to obtain the different information of two selected particles.

b) Strategy 2: The second strategy is the one used in standard PSO, to learn the global information by some particles to speed up algorithm convergence.

c) Strategy 3: The third strategy is the one used in the comprehensive learning particle swarm optimizer (CLPSO) [16], in which each dimension of particle learns from its *pbest* or other particle's *pbest* by a tournament selection method. To reduce the complexity, we randomly select a particle's *pbest* for all dimensions, instead of selecting a particle's *pbest* for each dimension. The velocity is updated as follows:

$$v_{id}(t+1) = wv_{id}(t) + cr[p_{kd}(t) - x_{id}(t)] \quad (16)$$

where is uniformly distributed real number in [0, 1]; $p_{kd}(t)$ is the pbest of a randomly selected particle.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

d) Strategy 4: It has been investigated in [16] that exploration range of particles in CLPSO is very large for almost all problems, resulting in a slow convergence. Therefore, Wang *et al.* [15] develop a new variant that searches a relatively smaller region, named PSO-CL-pbest. To reduce this strategy's complexity, we randomly select a particle's *pbest* for all dimensions. The velocity updating strategy is as follows:

$$v_{id}(t+1) = wv_{id}(t) + 0.5c[p_{kd}(t) - x_{id}(t) + p_{id}(t) - x_{id}(t)] \quad (17)$$

where is an uniformly distributed real number in [0, 1] and $p_{kd}(t)$ is the *pbest* of a randomly selected particle.

4.2.2 *Selection of Velocity Updating Strategies*: Each velocity updating strategy has an execution probability $ep_u(u=1,\ldots,4)$, which is the probability of choosing the strategy to update a particle. Initially, we assign equal probabilities for all the four strategies, namely $ep_u=0.25\ (u=1,\ldots,4)$, and set their accumulators as $A_u=0\ (u=1,\ldots,4)$. Every a fixed number of generations (that is, f_g), execution probability of the uth strategy is updated by

$$ep'_u = (1-\alpha)epu + \alpha\, A_u / f_g \quad (18)$$

$$epu = ep'_u / (ep'_1 + \ldots + ep'_4) \quad (19)$$

where learning rate is used to control the learning speed of execution probabilities; A_u is divided by f_g to ensure that this term is smaller than 1. This updating method ensures that execution probabilities of the strategies able to produce higher quality solutions are increased along with swarm evolution. This adaptive strategy of velocity
updating is able to increase the probability of choosing suitable velocity updating strategies and at the same time decrease those unsuitable.

4.3. Cuckoo Search

The Cuckoo search algorithm [17] represents the meta-heuristic search technique which was flagged off by Yang and Deb in 2010. In the search procedure, the following three vital rules are habitually employed.

a. According to the first rule, at a particular point of time, a cuckoo is competent to lay only one egg, which is thereafter discarded in an arbitrarily selected nest.
b. As per the second rule, the best nests with superior eggs are preserved for use in the successive next generation.
c. The third rule makes it clear that in the course of the entire search procedure, the number of accessible host nests continues to be a fixed number, and the host bird locates the egg laid by a cuckoo with a probability.

4.4 Solution representation

One of the most vital factors in the cloud computing is concerned with the representation of a solution for the purpose of task scheduling. Each application contains several tasks, in order that the entire applications may be treated as a set of tasks [11],

i.e., $T = \{t_1, t_2, \ldots, t_{TN}\}\left(TN = \sum_{j=1}^{w} T_j\right)$. A particle is indicated as a $TN(D = TN)$ dimension vector and each dimension (position) represents a task. A position with a greater value reveals the fact that the task represented by this position contains a superior priority and has to be picked first and allotted in the scheduling procedure.

Table I: Particle coding and decoding

Dimension	**1**	**2**	**3**	**4**	**5**
Position value	6.14	6.24	5.47	4.12	4.25
Priority	2	1	3	5	4

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

The ranked-order value (ROV) rule [18] is performed to generate a particle $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$ into a variant of tasks $T = \{t_1, t_2, ..., t_D\}$ to appraise the relative particle [19]. For example, for a problem with five tasks $(D = 5)$, the i^{th} particle is illustrated by $X_i = (6.14, 6.24, 5.47, 4.12, 4.25)$. The position x_{i2} contains the maximum value, in order that the task signified by x_{i2} is allotted a rank value one, as demonstrated in Table I; next $x_{i1} = 4.25$ is allotted a rank value two. Similarly, the rank values of 3, 4, and 5 are allotted to x_{i3}, x_{i5} and x_{i4}, respectively. Hence, it is possible to achieve a priority series of tasks, $Priority = \{2,1,3,5,4\}$.

4.5 Fitness Function

An appraisal function is needed while performing the SLPSO and CS to optimize the task permutation to substantially increase the profit of CSP_1, and also to estimate the fitness value of each particle in the swarm.

Each task in the set T is allotted to one $CSP_k (k = 1,2,...,n)$ based on its priority declared by the code of a particle. An effort is made to allot each task to CSP_1 as the cost of CSP_1 is found to be the minimum among all clouds. If CSP_1 contains accessible resources to satisfy the requirements of a task demand during its runtime, then the task is allotted to CSP_1; If not, the task is allotted to an EC with low cost which is chosen as illustrated in the following Equation 20.

$$E_{(l)}^{C} = \arg\min_{k \in \{2,...n\}} \left\{ \sum_{u=1}^{I} f_{app}(l) u CT_{ku} \right\}, \quad l \in \{1,2,..., D\} \tag{20}$$

Here, $app(l)$ refers to the application to which the l^{th} task belongs. Subsequent to the allotment of the entire tasks, the profit of CSP_1 is taken as the fitness value of the particle.

a) **Initialization:** Let total_cost = 0; avail_cpu$_s$ = total_CPU, avail_mem$_s$=total_MEM, $s \in \{1,2,...,S\}$

b) Evaluate the total income by means of Equation 21 given hereunder.

$$total\_income = \sum_{l=1}^{D} \sum_{u=1}^{I} f_{app(l)u} \; PC_u r_{app(l)} \tag{21}$$

c) Compute the total cost:

Arrange the task set T in an increasing order in accordance with the rank values illustrated by the code of a particle.

Let the l^{th} task in T be t_l and its start time be st_l, $l \in \{1,2,...,D\}$.

For $(l = 1 \; to \; D)$

While $(st_l \le d_{app(l)} - r_{app(l)} + 1)$

$IsPC = true$

For each $s \in \{st_l, ..., st_l + r_{app(l)} - 1\}$

If $\left(\sum_{u=1}^{I} CPU_u f_{app(l)u} \ge avail\_cpu_s \; or \; \sum_{u=1}^{I} MEM_u f_{app(l)u} \ge avail\_mem_s \right)$

$IsPC = false$

Break for.

End if

End for

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

If $(IsPC = true)$

Evaluate the cost for task t_l by

$$\cos t_l = \sum_{u=1}^{I} CT_{1u} f_{app(l)u} r_{app(l)}$$

Update $avail\_cpu_s$ and $avail\_mem_s$ for each $s \in \{st_l, ..., st_l + r_l - 1\}$

Break while

End if

$st_l = st_l + 1$

End While

If $IsPC = false$

Choose the EC with minimum cost employing (14).

Compute the cost t_l of task by

$$\cos t_l = \sum_{u=1}^{I} CT(EC_l)_u f_{app(l)u} r_{app(l)}$$

End if

$total\_\cos t = total\_\cos t + \cos t_l$

End for

d) $Output\ profit = total\_income - total\_\cos t$

4.6 Cuckoo operator

Each particle is modernized by the successive cuckoo search operator in the scheduling [17].

$$x^{(t+1)} = x_i^{(t)} + \alpha^{Levy(\lambda)} \quad (22)$$

Where;

$x_i \rightarrow i^{th}$ Solution

$x^{(t+1)} \rightarrow$ New solution

$\alpha \rightarrow$ Step size

Here, α $(\alpha > 0)$ indicates a step scaling size. The relative constraint has to be linked to the scales of issue for which the attempt to locate a solution. In a lion's share of the instances, α can be set either to 1 or to a certain different constant.

4.7 Stopping criterion

The technique concludes its performance only after attainment of the maximum number of iterations and the particle possessing the best fitness value is shortlisted and treated as the best feature to task scheduling. Subsequent to the achievement of the best fitness, the chosen task is allotted for the cloud computing procedure.

V. EXPERIMENTAL RESULTS AND ANALYSIS

5.1 Experimental Design

Two types of resources such as the CPU and memory are shortlisted as they are the two most characteristic configurations for choosing a VM instance. The task scheduling is performed in the Java (jdk 1.6) software and a number of tests have been carried out on a PC with Windows XP Operating system at 2 GHz dual core PC machine

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

with 2 GB main memory running a 64-bit version of Windows 2007. Further, the task scheduling approach is replicated by employing the CloudSim 3.0.
Parameters of problem instance are illustrated in Table II. The relative constraints are illustrated in Table III and IV.

Table II - Parameters of problem instance

Application		**Cloud resources**	
Parameters	**Values(integer)**	**Resources**	**Number**
Number of tasks	~unit[1,5]	CPU	20
VM instance type	~unit[1,3]	Memory	40GB
Deadline (hours)	~unit[1,5]		
Runtime (hours)	~unit[1,Deadline]		

Table III - Parameters of problem instance

Application		**Cloud resources**	
Parameters	**Values(integer)**	**Resources**	**Number**
Number of tasks	~unit[1,50]	CPU	512
Deadline (hours)	~unit[1,168]	Memory	1024GB
Runtime (hours)	~unit[1,Deadline]		

Table IV - Parameters of algorithm

Pop_size	**maxGen**	**minX**	**maxX**	**C1**	**C2**	V_{Max}	Ω
2D (Dimension)	1000	0	D	0.1	0.1	D/2	0.2

5.2 Experiment Results

The SLPSO technique is able to achieve the maximum profit in 24 assessments which is superior to CS. The standard profit attained in the 24 runs of the CS and SLPSO and their average runtime are detailed in Table V and VI.

Table V - Comparison for average profit

Algorithm	**Average Profit**
CS	1388.75
SLPSO	1849.93

Table VI - Comparison for average runtime

Algorithm	**Average Runtime**
CS	28.84
SLPSO	32.73

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

It is crystal clear that the average profit achieved by SLPSO is superior to CS technique, also SLPSO technique consume largest duration of time compared to CS technique. This is due to the overhead of selecting the velocity updating strategy.

The utilization rate of CPU or Memory at the s_{th} time slot for the private cloud is effectively evaluated by means of the equation (19)

$$M(s) = \sum_{i=1}^{D} \mathrm{Re}\, s_i \, E_{is} / Total - \mathrm{Re}\, s \quad ; \quad s \in \{1,2,...,S\} \tag{23}$$

Where;

$\mathrm{Re}\, s_i$ → Denotes the number of CPU or the size of memory demanded by task T_i

$Total - \mathrm{Re}\, s$ → Represents the total CPU or memory in the private cloud

In case task T_i functions in the S^{th} time slot, then $E_{is} = 1$; or else $E_{is} = 0$; the average utilization rate of CPU or Memory is evaluated by means of the Equation (20)

$$AU = \sum_{s=1}^{S} \frac{M(s)}{S} \tag{24}$$

The maximum CPU utilization and memory utilization rate is directly proportional to high system performance. Figure 2 shows the performance plot of CPU utilization rate for problem instance. When analyzing figure 2, we obtain the CPU utilization rate 0.758 of SLPSO which is high compare to the CS approach. Moreover, figure 3 shows the Performance plot of memory utilization for problem instance. Here, the SLPSO approach achieves the maximum memory utilization rate of 0.705 where as it is 0.589 for task scheduling using cuckoo search (CS) algorithm.

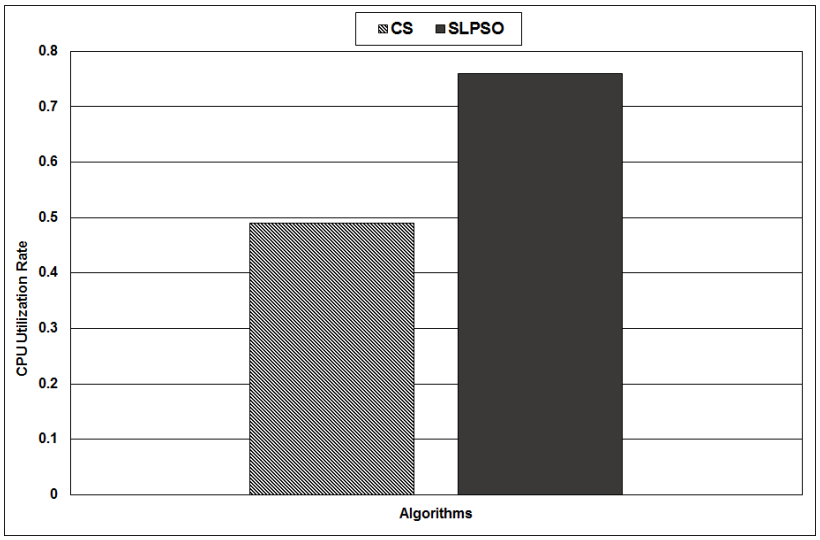

Figure 2 Performance plot for CPU utilization for problem instance

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

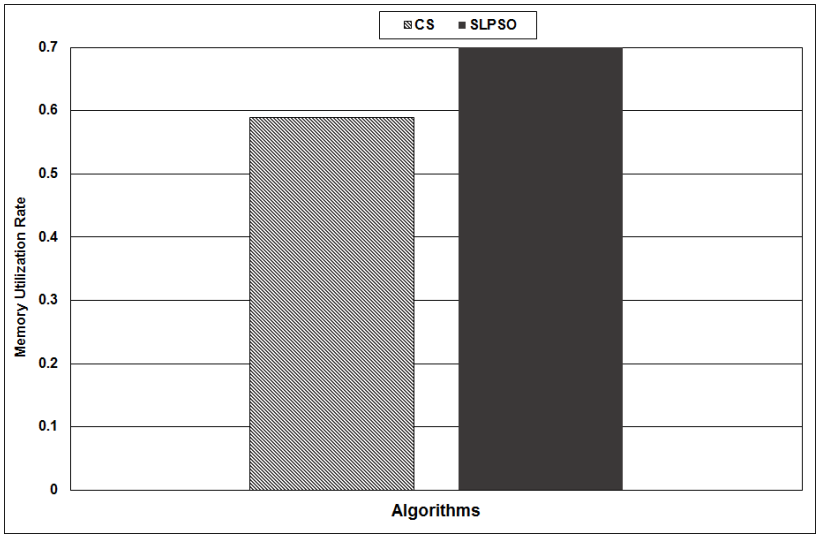

Figure 3 - Performance plot of memory utilization for problem instance

VI. CONCLUSION

In the hybrid cloud scenario, an optimal task scheduling is flagged off devoted for effectively tackling the resource allocation challenge of an IaaS cloud, in the current investigation. Here, SLPSO and cuckoo search-based scheduling methods are envisaged for solving the issue. The cardinal issue is concerned with the allotment of users' tasks to scale up the revenue of Infrastructure as a Service (IaaS) provider to the maximum possible, simultaneously ensuring the Quality-of-Service (QoS). By employing two optimization approaches such as the SLPSO and Cuckoo Search, a better quality scheduling solution is acheived. The generated solution is skilled enough to ensure user-level (QoS) and perk up the reliability of the IaaS providers together with an enhancement in financial gains. As regards the task scheduling issue, the test outcomes illustrated unanimously that the SLPSO based scheduling technique is better than CS approach.

REFERENCES

1. Arshad, J., Townend, P. and Xu, J. (2013) A Novel Intrusion Severity Analysis Approach for Clouds. Future Generation Computer Systems, 29, 416-428.
2. Malathi, M. (2011), "Cloud Computing Concepts," 3rd International Conference on Electronics Computer Technology (ICECT).
3. Rabai, L.B.A., et al. (2013) A Cyber security Model in Cloud Computing Environments. Journal of King Saud University-Computer and Information Sciences, 25, 63-75.
4. Khorshed, M.T., Ali, A.B.M.S. and Wasimi, S.A., (2012), "A Survey on Gaps, Threat Remediation Challenges and Some Thoughts for Proactive Attack Detection in Cloud Computing," Future Generation Computer Systems, 28, 833-851.
5. A.N.Toosi,R. N.Calheiros, P.K.Thulasiram, and R.Buyya, "Resource provisioning policies to increase IaaS provider's profit in a federated cloud environment," in Proc. IEEE Int. Conf. High Perform. Comput. Commun., Banff, Canada, 2011, pp. 279–287.
6. D. Breitgand, A. Maraschini, and J. Tordsson, Policy-Driven Service Placement Optimization in Federated Cloud, IBM Research Report, 2011.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 3, March 2018

7. S. Ortiz, "The problem with cloud-computing standardization", Computer, vol. 44, no. 7, pp. 13–16, 2011
8. J. Kennedy and R. C. Eberhart, "Particle swarm optimization", in Proc. IEEE Int. Conf. Neural Network, vol. 4., 1995, pp. 1942–1948.
9. R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proc. 6th Int. Symp. Micromach. Human Sci., 1995, pp. 39–43.
10. J. Kennedy, R. C. Eberhart, and Y. H. Shi, Swarm Intelligence. San Mateo, CA: Morgan Kaufmann, 2001.
11. Xingquan Zuo, Guoxiang Zhang, and Wei Tan, "Self-Adaptive Learning PSO-Based Deadline Constrained Task Scheduling for Hybrid IaaS Cloud," IEEE Transactions on Automation Science and Engineering, Vol. 11, No. 2, April 2014
12. S. Abrishami, M. Naghibzadeh,"Deadline-constrained workflow scheduling in software as a service Cloud,"Scientia Iranica, vol.19, no. 3, pp.680–689, 2012.
13. Lizheng Guo, Shuguang Zhao, Shigen Shen,Changyuan Jiang, "Task Scheduling Optimization in Cloud Computing Based on Heuristic Algorithm, "Journal Of Networks, Vol. 7, No. 3, March 2012.
14. Y. H. Shi and R. C. Eberhart, "Comparison between genetic algorithms and particle swarm optimization," in Proc. 7th Int. Conf. Evol. Program., LNCS 1447. Mar. 1998, pp. 611–616.
15. Y.Wang, B. Li, T.Weise, J.Wang, B.Yuan, and Q. Tian, "Self-adaptive learning based particle swarm optimization," *Inf. Sci.*, vol. 181, no. 20, pp. 4515–4538, 2011.
16. J. J. Liang, A. K. Qin, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal function," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.
17. X.-S.Yang and S. Deb, "Cuckoo search via Lévy flights", World Congress on Nature & Biologically Inspired Computing IEEE Publications, pp. 210–214, 2009.
18. B. Liu, L. Wang, and Y. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," IEEE Trans. System, Man, Cybern., Part B: Cybern., vol. 37, no. 1, pp. 985–997, 2007.
19. J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," ORSA J. Computing, vol. 6, no. 2, pp. 154–160, 1994.