ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Analysis of Cost Optimization using Heuristic Algorithms in Cloud

Sridhar P[1], Kavitha B[2]

Senior Lecturer, Department of Computer Science and Engineering, IRT Polytechnic College

Chennai, Tamilnadu, India

Lecturer, Department of Electronics and Communication Engineering, IRT Polytechnic College

Chennai, Tamilnadu, India

ABSTRACT: The distributed computing systems have got more attention due to the increasing demand for high performance computing and data storage. The resource allocation is one of the big problems in the distributed systems when the client wants to decrease the cost for the resource allocation for their task. In order to assign the resource for the task, the client must consider the monetary cost and computational cost. Allocation of resources by considering those two costs is difficult. To solve this problem in this paper, we make the main task of client into many subtasks and we allocate resources for each subtask instead of selecting the single resource for the main task. The allocation of resources for the each subtask is completed through optimization algorithm. Here, we implement the binary particle swarm optimization (BPSO) and binary cuckoo search algorithm (BCSO) by considering monetary cost and computational cost which helps to minimize the cost of the client. Finally, the experimentation is carried out to compare BPSO and BCSO algorithms. Experimental results shows that BCSO outperforms BPSO algorithm.

I. INTRODUCTION

With the rapid development of processing and storage technologies and the success of the Internet, computing resources have become cheaper, more powerful and more ubiquitously available than ever before. This technological trend has enabled the realization of a new computing model called Cloud computing [1]. In a cloud computing environment, an Infrastructure-as-a-Service (IaaS) provider packages its physical resources (e.g. CPU, memory disk) into distinct types of virtual machines (VMs) in terms of their sizes and features, and offers them as services to the general public [2]. Also it delivers an infrastructure, platform, and software (applications) as Services that are made available to consumers in a pay-as-you-go model. In industry these services are referred to as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) respectively.

Particle swarm optimization (PSO) [3, 4], is one of the most important swarm intelligence paradigms. The PSO uses a simple mechanism that mimics swarm behaviour in birds flocking and fish schooling to guide the particles to search for globally optimal solutions. In PSO algorithm, each member is called “particle”, and each particle flies around in the multi-dimensional search space with a velocity, which is constantly updated by the particle’s own experience and the experience of the particle’s neighbours. The main idea behind the development of PSO is the social sharing of information among individuals of population. In PSO algorithms, search is conducted by using a population of particles, corresponding to individuals as in the case of evolutionary algorithms. According to the global neighbourhood, each particle moves towards its best previous position and towards the best particle in the whole swarm, called G-best model. On the other hand, according to the local variant, called L-best model, each particle moves towards its best previous position and towards the best particle in its restricted neighbourhood [5, 6].

The Cuckoo Search (CS) [7, 8], is a new meta-heuristic algorithm imitating animal behaviour. The optimal solutions obtained by the CS are far better than the best solutions obtained by efficient particle swarm optimizers and genetic

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

algorithms [7]. Cuckoo Search Algorithm is based on the brood parasitism of some cuckoo species [9]. Yang and Deb [7,8,10,11], discovered that the performance of the CS can be improved by using Levy Flights instead of simple random walk. The CS was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of host birds. Some cuckoos have evolved in such a way that female parasitic cuckoos can imitate the colors and patterns of the eggs of a few chosen host species [12]. This reduces the probability of the eggs being abandoned and, therefore, increases their re-productivity. It is worth mentioning that several host birds engage direct conflict with intruding cuckoos [13, 14]. In this case, if host birds discover the eggs are not their own, they will either throw them away or simply abandon their nests and build new ones, elsewhere [15, 16]. For simplicity in describing the cuckoo search, consider the following three idealized rules: 1) Each cuckoo lays one egg at a time, and dump its egg in randomly chosen nest; 2) The best nests with high quality of eggs will carry over to the next generations; 3) The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird [17, 18]. The effective way of choosing an optimization algorithm can further improve the results, specifically for resource allocation.

The main task of the client is divided into many subtasks, then allocates the resource for each subtask instead of selecting the single resource for the whole entire task. The total cost of the client is calculated using the fitness function taking the monetary cost and computational cost into account. Here, we utilise two optimization algorithm such as Binary Cuckoo Search Optimization (BCSO) algorithm and Binary Particle Swarm Optimization (BPSO) algorithm to assign cloud resources to the tasks so as to minimize the cost of the client.

In BCSO algorithm, initialization matrix is generated based on the number of resources and number of subtasks. Each cuckoo particle is evaluated in the initialization matrix through the objective function. The cuckoo particle is updated based on levy flights and the fitness function is evaluated. This process is repeated for a given number of iterations. The fitness value is compared with the previous fitness value in each iteration and the minimum fitness value is considered to be the best solution.

In BPSO algorithm, the initialization matrix is generated as the number of resources and subtasks. A velocity matrix is generated randomly for the initialization matrix and the fitness function is evaluated. The G-best and P-best values are set with the initial P-best value and then the particle's velocity and position are updated in the matrix. The iteration starts with the computation of fitness function using the updated matrix and arrived result is the current P-best value. If this P-best value, is lesser than the previous G-best value, then G-best is updated with that new P-best value. This cycle is repeated for a given number of iterations. The final G-best value obtained after all iterations is the optimal solution.This cyclic process is repeated up to K number of iterations in order to obtain an optimal resource allocation for the client's task.

The rest of the paper is organized as follows: a brief review of some of the literature works is presented in Section 2. The algorithms for dynamic resource allocation is given in Section 3. The experimental results and the performance evaluation are provided in Section 4. Finally, the conclusion is summed up in Section 5.

II. RELATED WORKS

Here, a review of some of the works are presented for resource allocation in cloud computing. Bo Yin *et al* [19] have proposed a multi-dimensional resource allocation scheme for cloud computing that dynamically allocates the virtual resources for the cloud computing applications with reduced cost by using fewer nodes to process the applications. They adopted a two-stage algorithm using multi-constraint integer programming problem. Experimental results shows the improved resource utilization and reduced cost of data center. Daniel Warneke and Odej Kao [20] have discussed the opportunities and challenges for efficient parallel data processing in clouds and present their research project Nephele. Nephele was the first data processing framework to explicitly exploit the dynamic resource allocation offered by today's IaaS clouds for both, task scheduling and execution. Particular tasks of a job could be assigned to different types of virtual machines which were automatically instantiated and terminated during the job execution. Based on

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

these frameworks, they perform the extended evaluations of Map Reduce-inspired processing jobs on an IaaS cloud system and compared the results to the popular data processing framework Hadoop.

Yazir, *et al* [21] have investigated a dynamic autonomous resource management in cloud computing. The main contribution has two-fold. First, they adopt a distributed architecture where resource management has decomposed into independent tasks, each of which was performed by Autonomous Node Agents that were tightly coupled with the physical machines in a data center. Second, the Autonomous Node Agents carry out configurations in parallel through Multiple Criteria Decision Analysis. Simulation results showed that the promising effects in terms of scalability, feasibility and flexibility.

R.Gogulan*et al* [22] have introduced a algorithm called Multiple Pheromone Algorithm (MPA) which has belongs to Ant Colony Optimization Algorithm. The objective of MPA algorithm was to dynamically generate an optimal schedule so as to be completed the task in minimum period of time as well as utilizing the resources in an efficient way. They have three different Quality of Service (QoS) make span, cost and reliability constraints were considered as performance measure for scheduling. Finally, the algorithm was compared with normal Ant Colony Algorithm(ACO) and Genetic Algorithm (GA). With the implementation of the Multiple Pheromone Algorithm (MPA), it reached an optimal solution as well as obtained the better QoS than ACO and GA.

In cloud computing, the resource allocation for a task with the minimum cost is a challenging criterion. This problem is solved by computing the total cost using the monetary cost and the computational cost for the task. Evolutionary algorithms like Cuckoo Search and Particle Swarm Optimization can be utilized to achieve the minimum cost. Here we analyse the optimization of cost using BCSO and BPSO algorithms so as to achieve better results.

III. COST OPTIMIZATION IN ALGORITHMS

In this paper, we apply optimization algorithm for dynamic resource allocation in cloud computing. Here, two optimization algorithms such as Binary Cuckoo Search Optimization (BCSO) algorithm and Binary Particle Swarm Optimization (BPSO) algorithm are utilized to schedule tasks to cloud resources, that takes into account both monetary cost and computational cost. Monetary Cost (MC) is considered to be the cost which is allotted by the service provider as the maintenance cost of the resources which includes electricity, building, cooling etc., to the client to utilize the resources to do the client's task. This cost may differ from one service provider to another for the same task. Computational Cost (CC) is also assigned by the service provider based on the computational complexity of the task assigned by client to the resource. The computational cost also differs from one service provider to another since the computational complexity includes the number of core required, required memory space, bandwidth etc., Cloud service providers invest 70% of the cost for the maintenance of the data center where as only 30% of the cost is spent on actual IT resources.

To reduce the overall cost of the resource allocation for the task of the clients, here we divide the main task into many subtasks instead of selecting the best resource among the available resources for the whole task. Best resources for each subtask from the available resources are selected and alloted through the optimization algorithms such as BPSO and BCSO. These optimization algorithms helps to reduce the monetary cost and computational cost of each subtask.

The client submits a task T and desires to complete the task with less cost and efficient computation through the cloud service providers. Consider there are n number of available resources$R=\{R_1,R_2 \ldots R_i \ldots R_n\}$ where $1 \leq i \leq n$ in the cloud to do the task T. To improve the efficiency and reduce the total cost of the task T, we divide the main task T into m number of subtasks $T=\{t_1,t_2..t_j..t_m\}$ where $1 \leq j \leq m$. To complete the task efficiently with less cost, the two costs such as computational cost and monetary cost are considered at the same time.

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Resource allocation based on the input demand is a challenging issue in cloud, since both resource allocation and input demand must be satisfied for multiple constraints. So the allocation of resources for an input demand is increasing day-by-day in cloud computing resource management system. But the major problem is handling of real time constraints such as, monetary cost and computational cost.

TABLE I
MONETARY COST OF RESOURCES FOR EACH SUBTASK

Resource / Subtask	R_1	R_2	R_i	R_{n-1}	R_n
t_1	$MC_{1,1}$	$MC_{1,2}$	$MC_{1,i}$	$MC_{1,n-1}$	$MC_{1,n}$
t_2	$MC_{2,1}$	$MC_{2,2}$	$MC_{2,i}$	$MC_{2,n-1}$	$MC_{2,n}$
t_j	$MC_{j,1}$	$MC_{j,2}$	$MC_{j,i}$	$MC_{j,n-1}$	$MC_{j,n}$
t_{m-1}	$MC_{m-1,1}$	$MC_{m-1,2}$	$MC_{m-1,i}$	$MC_{m-1,n-1}$	$MC_{m-1,n}$
t_m	$MC_{m,1}$	$MC_{m,2}$	$MC_{m,i}$	$MC_{m,n-1}$	$MC_{m,n}$

For example, let us consider that, the main task T is divided into 5 subtasks t_1, t_2, t_3, t_4, t_5 and there are seven resources R_1,R_2,R_3,R_4,R_5,R_6,R_7 available from the cloud. For each subtask the algorithm helps to select the better resource based on the monetary cost and computational cost. Normally client desires to access the resources with minimum cost, so that the client will be benefited. Here, we consider monetary cost and computational cost for computing the total cost of the resources. Each of the resources in the cloud assigns the monetary cost (MC) and computational cost (CC) for each subtask t_j. The Table I and Table II represent the monetary cost and computational cost of *m*number of subtasks with respect to *n* number of resources. Table III represent the parameters used.

TABLE II
COMPUTATIONAL COST OF RESOURCES FOR EACH SUBTASK

Resource / Subtask	R_1	R_2	R_i	R_{n-1}	R_n
t_1	$CC_{1,1}$	$CC_{1,2}$	$CC_{1,i}$	$CC_{1,n-1}$	$CC_{1,n}$
t_2	$CC_{2,1}$	$CC_{2,2}$	$CC_{2,i}$	$CC_{2,n-1}$	$CC_{2,n}$
t_j	$CC_{j,1}$	$CC_{j,2}$	$CC_{j,i}$	$CC_{j,n-1}$	$CC_{j,n}$
t_{m-1}	$CC_{m-1,1}$	$CC_{m-1,2}$	$CC_{m-1,i}$	$CC_{m-1,n-1}$	$CC_{m-1,n}$
t_m	$CC_{m,1}$	$CC_{m,2}$	$CC_{m,i}$	$CC_{m,n-1}$	$CC_{m,n}$

TABLE III
PARAMETERS USED

Symbol	*Functionality of the symbol*
n	*Number of resources*
m	*Number of subtasks*
MC	*Monetary cost*
CC	*Computational cost*
K	*Iteration counter of BCSO*
τ	*Iteration counter of BPSO*
K	*Maximum number of iterations*
$f(t^k_j)$	*Fitness of j^{th} subtask at k^{th} iteration*
CC_{ji}	*Communication cost of j^{th} task with i^{th} resource*
MC_{ji}	*Monetary cost of j^{th} task with i^{th} resource*

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Algorithm:

Input: *m* number of subtasks, *n* number of resources, monetary cost, computational cost of *n* number of resources for *m* number of subtasks

Output: Minimum cost of resource allocation for each subtask.

3.1 Binary Cuckoo Search Optimization

BCSO consists of four steps such as generation of initialization matrix, computing fitness function, particle updation and position updation which are explained in this section.

3.1.1 Initialization Matrix

Initially the value of iteration counter *k* is assigned as zero. After the every iteration the value of the iteration counter gets increase by one. Generate the initialization matrix *n* x *m*where *n* and *m* represent, the number of resources and number of subtasks. In the initialization matrix if the resource R_i is allocated to the subtask t_j in k^{th}, it is represented as '1', otherwise '0'

3.1.2 Fitness Function

To evaluate fitness of each cuckoo particle from the initialization matrix, the fitness function is computed. The fitness function includes the monetary cost and computational cost of the resources (dimension) for each subtask (cuckoo particle). Equation (1) is used to calculate the fitness value based on the monetary cost and the computational cost.

$$f\left(t_j^k\right) = CC_{ji} + MC_{ji} \quad (1)$$

In the equation (1), the value of $f(t^k_j)$ represents the total cost of j^{th}subtask at the i^{th} resource in the k^{th} iteration.

As an example, in the table 5, the subtask $t_1$is assigned to the resource R_4. Based on this data, we select the values MC_{14} and CC_{14} as the value of the monetary cost and the computational cost of the subtask t_1 with respect to the resource R_4 and the computed fitness values are shown in table 6.

3.1.3 Cuckoo Particle Updation

In nature, animals search for food in a random or quasi-random manner. Generally, the foraging path of an animal is effectively a random walk because the next move is based on both the current location/state and the transition probability to the next location. The chosen direction implicitly depends on a probability, which can be modelled mathematically

Various studies have shown that the flight behaviour of many animals and insects demonstrates the typical characteristics of Levy flights [19]. A Levy flight is a random walk in which the step-lengths are distributed according to a heavy-tailed probability distribution. After a large number of steps, the distance from the origin of the random walk tends to be a stable distribution. Each dimension of the every cuckoo particle is updated through levy flights. The equations (2, 3, 4, 5 and 6) are used to update the every dimension of each particle of the initialization matrix. The values of γ, β, *rand*are the random values between 0 and 1. The value of *R(n)* in the equation (3) represents the random value between *0 to n*where *n*represents the total number of available resources to do the *m* number of subtasks.

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

$$\sigma = \left[\frac{\gamma(1+\beta)\sin\left(\frac{\pi * \beta}{2}\right)}{\gamma\left(\frac{1+\beta}{2}\right) * (\beta) * \left(2^{\frac{\beta-1}{2}}\right)} \right]^{\frac{1}{\beta}} \quad (2)$$

$$U = R(n) * \sigma \quad (3)$$

$$V = R(n) \quad (4)$$

$$step\left(x_{jd}^{k}\right) = \frac{U}{\sim V^{1/\beta}} \quad (5)$$

$$step\ size\left(x_{jd}^{k}\right) = rand * step \quad (6)$$

Here $x^{k}_{jd}$is the j^{th}particle position at the k^{th} iteration in the dimension *d*, where d represents the resource number, *U* and *V* are normally distributed stochastic variables with standard deviation (σ). Equation (6) represents the calculation of step size value necessary to update the dimension.

3.1.4 Position Updation in BCSO Algorithm

The dimension of the each cuckoo particle is updated based on step size value of that particle. The step size value of the each dimension of every cuckoo particle is applied into the sigmoid function, given in the equation (7) and the updated value becomes "0" or "1" through the condition which is represented in the equation (8).

$$S\left(step\ size\left(x_{jd}^{k}\right)\right) = \frac{1}{1+e^{-step\ size\left(x_{jd}^{k}\right)}} \quad (7)$$

$$if\ \left[(rand\ (0,1)) < S\left(step\ size\left(x_{jd}^{k}\right)\right)\right]\ then\ \ x_{jd}^{k} = 1 \quad (8)$$

$$else\ \ x_{jd}^{k} = 0$$

While updating the particle in each iteration, if the subtask is assigned to more than 1 resource at a time, the fitness value of each resource is calculated first and then the resource having a minimum fitness value is selected.

BCSO algorithm

Set k=0
Generate initialization matrix n x m
For each particle
Get the value of MC & CC
Calculate fitness f (t $^{k}_{j}$) = MC_{ji} + CC_{ji}
Call Particle updation of BCSO
End for

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Subroutine: Particle updation of BCSO

Set k = k + 1
Calculate σ
Calculate U and V
Calculate step
Calculate step size
Apply the step size value of particle to sigmoid function
Generate rand (0,1)
If sigmoid function>1
Particle become 1
Else
Particle become 0

3.2Binary Particle Swarm Optimization:

There are four steps involved in BPSO. They are, generating initialization matrix, generating velocity matrix, computing fitness values, velocity updation and position updation which are explained in this section briefly.

3.2.1 Initialization Matrix

The BPSO algorithm process initialization matrix to find the global best value (G- best) and local best (P-best).

3.2.2 Velocity Matrix

The updating process of the BPSO algorithm is based on the velocity matrix *n* x *m* which is generated randomly as the same size of the initialization matrix. The generation of velocity matrix is completed before the calculation of fitness function. The velocity of the particle is represented by v^{τ}_{jd}in which *j*represents number of subtask, *d*represents dimension of the particle and τ represents the iteration counter value. The value of the iteration counter τ is initiated from 0 and is incremented by two (i.e. 0, 2, 4, 6) every time.

3.2.3 Fitness Function

The fitness function includes the monetary cost and computational cost of the resources for each subtask. The fitness function used in the BCSO algorithm is also used in BPSO algorithm as represented in the equation (1). Fitness values for the initialization matrix of the BPSO algorithm is given in table 7.

3.2.4Velocity Updation

After calculating the fitness of every particle of the matrix, the next step is to update particle position. Before that velocity of the particle should be updated. The equations (9) and (10) are used to update the (particle) velocity. *w* is used to control the impact of the previous velocities on the current velocity.

$$\Delta v^{\tau}_{jd} = c_1 r_1 \left(pb^{k}_{jd} - x^{k}_{jd}\right) + c_2 r_2 \left(gb^{k}_{jd} - x^{k}_{jd}\right) \quad (9)$$

$$v^{\tau+2}_{jd} = w[v^{\tau}_{jd}] + \Delta v^{\tau}_{jd} \quad (10)$$

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

where

Δv_{jd}^{τ} - *change in the Velocity of* j^{th} *particle at* τ^{th} *iteration in dimension d*

c_1, c_2 - *Acceleration constants*

r_1, r_2 -*Random numbers in the interval [0,1]*

pb_{jd}^{k} - *P-best value of* j^{th} *particle at* k^{th} *iteration in dimension d*

x_{jd}^{k} - *Position of* j^{th} *particle at* k^{th} *iteration in the dimension d*

gb_{jd}^{k} - *G -best value of* j^{th} *particle at* k^{th} *iteration in dimension d*

v_{jd}^{τ} - *Updated Velocity of* j^{th} *particle at* τ *iteration in dimension d*

$v_{jd}^{\tau+2}$ - *Updated Velocity of* j^{th} *particle at* $\tau+2$ *iteration in dimension d*

w - Inertia weight

3.2.5 Position UpdationIn BPSO Algorithm

With the help of updated velocity of each dimension of the particle, the BPSO algorithm updates the every dimension of the particle. The equations (11) and (12) are used to update the each dimension of the particle matrix. The updated matrix is given as the next input of BCSO algorithm.

$$S\left(v_{jd}^{\tau+2}\right)=\frac{1}{1+e^{-\left(v_{jd}^{\tau+2}\right)}} \qquad (11)$$

$$if\ \left[(rand\ (0,1))<S\left(v_{jd}^{\tau+2}\right)\right]\ then\ \ x_{jd}^{\tau}=1$$
$$else\ \ x_{jd}^{\tau}=0 \qquad (12)$$

While updating the particle in each iteration, if the subtask is assigned to more than 1 resource at a time, the fitness value of each resource is calculated at first and then the resource having a minimum fitness value is selected. The algorithm stops when a predefined number of iterations are completed.

BPSO algorithm

Get the updation matrix from BCSO algorithm as initialization matrix n x m
Set $\tau = 0$
Generate the velocity matrix of the initialization matrix n x m
For each particle
Get the value of MC & CC
Calculate fitness $f(t^{k}_{j}) = MC_{ji} + CC_{ji}$
Update G-best & P-best
Call Particle updation of BPSO
End for

Subroutine: Particle updation of BPSO

Set $\tau = \tau + 2$
Calculate Δv
Calculate updated velocity

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Apply updated velocity to sigmoid function
Generate rand (0,1)
If sigmoid function>1
Particle become 1
Else
Particle become 0

IV. IMPLEMENTATION AND RESULT ANALYSIS

The experimental results for the optimization algorithms for dynamic resource allocation are described in this section. In this paper, we compare BPSO algorithm with the BCSO algorithm.

The algorithm are implemented with Cloudsim version 2.1.1. The two synthetic datasets are generated in times of interval based on the number of subtask and available resources. Also each of the synthetic dataset consists of monetary cost and computational cost of each task with respect to all resources.

In this paper, our main aim is to allocate the resource with minimum amount of monetary cost as well as computational cost. Here the total cost of the task is taken as the evaluation metrics. The total cost is calculated by the summation of fitness value of every subtask. The evaluation is carried out with the total cost by varying the number resources and number of subtasks. Equation (13) is used to find the total cost (TC) of the task.

$$TC = \sum_{j=1}^{m} f(t_j) \qquad (13)$$

The performance evaluation of the optimization algorithms is prepared by comparing with the binary cuckoo search algorithm and binary particle swarm optimization.

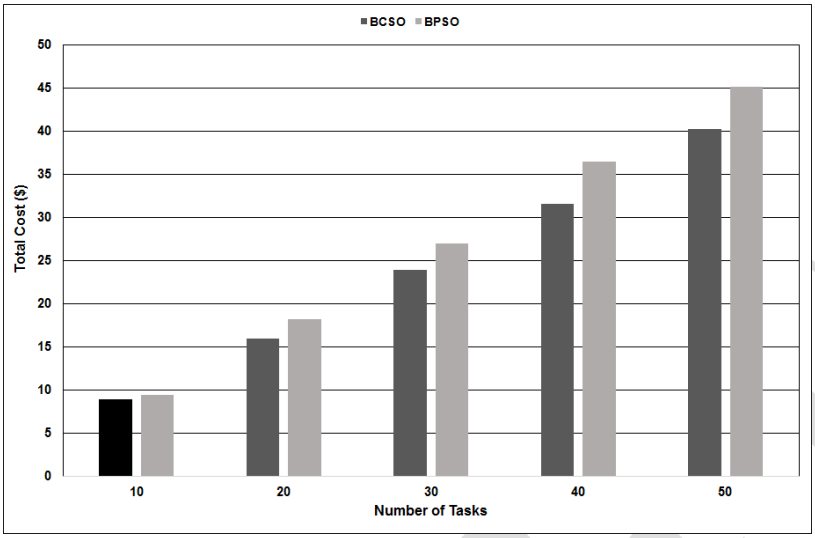
Fig 1(a) and 1(b) represents the performance of the BCSO and BPSO algorithm for 25 resources, 35 iterations for the datasets D1 and D2 respectively with varying subtasks. As the number of subtasks increases, the total cost of the task keeps increasing in the algorithms. At any level of subtasks, for a given number resources the cost achieved by BSCO is less than the cost achieved by BPSO. For both the datasets D1 and D2, the cost of BCSO algorithm is less than BPSO algorithm.

ISSN: 2319-8753

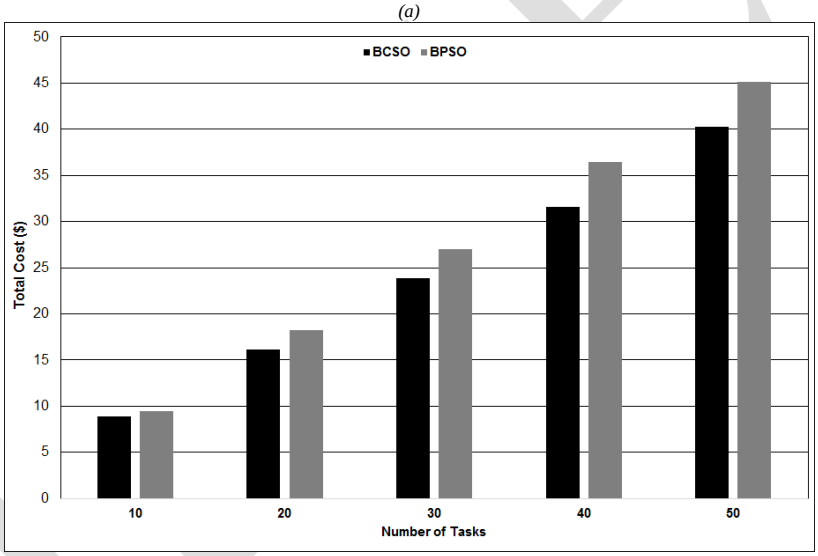
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014



(a)



(b)

TABLE IV

AVERAGE PERFORMANCE OF THE THREE ALGORITHMS

Number of resources	D1 dataset		D2 dataset	
	BCSO	BPSO	BCSO	BPSO
25	23.84	26.96	24.48	27.31
50	22.99	26.84	23.73	26.92
100	22.64	26.54	23.44	26.83

DOI: 10.15680/IJIRSET.2014.0309090

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

Table IV represents the average performance of the BCSO and BPSO algorithms for the numbers of resources, n=25, 50 and 100 and 35 number of iterations. From the table it is clear that BCSO outperforms BPSO irrespective of number of resources and iterations.

V. CONCLUSION

We comparedtwo different optimization algorithms for dynamic resource allocation in cloud computing. BCSO algorithm generates the initialization matrix as per the number of subtasks and resources. Subsequently it calculates the fitness value for each particle of the initialization matrix based on the monetary cost and computational cost. The fitness of the particle is evaluated in BPSO algorithm.Then the initialization matrix of the BCSO algorithm gets updated through the levy flights and sigmoid function. The BPSO algorithm generates the velocity matrix randomly for the initialization matrix and then calculates the fitness value. Based on the arrived fitness value, the values of G-best and P-best are updated. The BPSO algorithm updates the matrix through the updated velocity and sigmoid function. This process is repeated up to *k* number of iterations. The experimentation is carried out and the results of the BCSO and BPSO algorithms are compared. The BCSO algorithm gives better cost optimized results compared to BPSO algorithm. Hybrid combination of other evolutionary algorithms can be carried out for further scope.

REFERENCES

[1] Qi Zhang, Lu Cheng and RaoufBoutaba, "Cloud computing: state-of-the-art and research challenges", Journal International Service Applications, vol.1, no.6, pp. 7-18, 2010.

[2] Qi Zhang, Quanyan Zhu and RaoufBoutaba, "Dynamic Resource Allocation for Spot Markets in Cloud Computing Environments", Utility and Cloud Computing (UCC), pp.178-185, Dec 2011.

[3] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. Neural Networks., Perth, Australia, 1995, vol. 4, pp. 1942–1948.

[4] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proc. 6th Int. Symp. Micromachine Human Sci., Nagoya, Japan, 1995, pp. 39–43.

[5] J. Kennedy, R. C. Eberhart, and Y. H. Shi, "Swarm Intelligence", San Mateo, CA: Morgan Kaufmann, 2001

[6] S.Y. Ho, H.-S. Lin, W.-H. Liauh, and S.J. Ho, "OPSO: Orthogonal Particle Swarm Optimization and its Application to Task Assignment Problems," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 38, no. 2, pp. 288–298, Mar. 2008.

[7] Zhi-Hui Zhan, Jun Zhang, Yun Li,Member, Henry Shu-Hung Chung, "Adaptive Particle Swarm Optimization" IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 39, no. 6, pp. 1362-1381, 2009.

[8] M. FatihTaşgetiren and Yun-Chia Liang "A Binary Particle Swarm Optimization Algorithm for Lot Sizing Problem", Journal of Economic and Social Research, vol. 5, no. 2, pp. 1-20, 2003

[9] Yang XS, Deb S, "Cuckoo Search via Levy Flights", Proceedings of World Congress on Nature and Biologically Inspired Computing, India, pp. 210-214, 2009.

[10] Yang XS, Deb S, "Engineering Optimization by Cuckoo Search", International journal of Mathematical Modeling and Numerical Optimisation. Vol. 1, No. 4, pp. 330–343, 2010.

[11] IvonaBrajevic, Milan Tuba, NebojsaBacanin, "Multilevel Image Thresholding Selection Based on the Cuckoo Search Algorithm", Advances in Sensors, Signals, Visualization, Imaging and Simulation, pp. 217-222, 2012.

[12] AbdesslemLayeb,SerielRayeneBoussalia, "A Novel Quantum Inspired Cuckoo Search Algorithm for Bin Packing Problem", International Journal of Information Technology and Computer Science, vol. 4, no. 5, pp. 58-67, May 2012.

[13] EhsanValian, ShahramMohanna and Saeed Tavakoli, "Improved Cuckoo Search Algorithm for Feed Forward Neural Network Training", International Journal of Artificial Intelligence and Applications, vol.2, no.3, pp. 36-43, July 2011

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2014

[14] EhsanValian, ShahramMohanna and Saeed Tavakoli, "Improved Cuckoo Search Algorith Global Optimization", International Journal of Communications and Information Technology, vol.1, no.1, pp. 31-44, Dec. 2011.

[15] Jati, GilangKusuma, Manurung, HisarMaruli, Suyanto "Discrete Cuckoo Search for Traveling Salesman Problem", 7th International Conference on Computing and Convergence Technology (ICCCT), pp. 993 - 997, 2012.

[16] Valian; S. Mohanna; S. Tavakoli, "Improved Cuckoo Search Algorithm for Global Optimization" vol.1, no.1, pp. 31-44, 2011.

[17] ManianDhivya, MurugesanSundarambal, LoganathanNithisshAnand "Energy Efficient Computation of Data Fusion in Wireless Sensor Networks Using Cuckoo Based Particle Approach", Int. J. Communications, Network and System Sciences, vol. 4, pp. 249-255, 2011.

[18] R.G. Babukartik and P. Dhavachelvan, "Hybrid Algorithm using the advantage of ACO and Cuckoo Search for Job Scheduling", International Journal of Information Technology Convergence and Services, vol.2, no.4, pp. 25-34, August 2012.

[19] KhairulNajmy Abdul Rani, Mohd. FareqAbdMalek and NeohSiew-Chin, "Nature-inspired Cuckoo Search Algorithm for Side Lobe Suppression in a Symmetric Linear Antenna Array", Radioengineering, vol. 21, no. 3, pp. 865-874, September 2012

[20] AminrezaNoghrehabadi, Mohammad Ghalambaz, Mehdi Ghalambaz and Amir Vosough "A hybrid Power Series - Cuckoo Search Optimization Algorithm to Electrostatic Deflection of Micro Fixed-fixed Actuators", International Journal Of Multidisciplinary Sciences and Engineering, vol. 2, no. 4, pp. 22-26, July 2011

[21] Bo Yin, Ying Wang, LuomingMeng and XuesongQiu, "A Multi-dimensional Resource Allocation Algorithm in Cloud Computing", Journal of Information and Computational Science, vol. 9, no. 11, pp. 3021–3028, 2012.

[22] Daniel Warneke and Odej Kao, "Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud", IEEE Transactions on parallel and distributed systems, vol.3, no.3, pp.1-3, January 2011.

[23] Yazır, YağızOnat; Matthews, Chris; Farahbod, Roozbeh; Guitouni, Adel; Neville, Stephen; Ganti, Sudhakar; Coady, Yvonne, "Dynamic resource allocation in computing clouds through distributed Multiple Criteria Decision Analysis", in proceedings of IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 91 - 98, 2010.

[24] R.Gogulan, A.Kavitha, and U.Karthick Kumar, "A Multiple Pheromone Algorithm for Cloud Scheduling with Various QOS Requirements", IJCSI International Journal of Computer Science Issues, vol. 9, issue 3, May 2012.