International Journal of Soft Computing 10 (1): 37-45, 2015
ISSN: 1816-9503
© Medwell Journals, 2015

A Multi-Objective Cat Swarm Optimization Algorithm for Workflow Scheduling in Cloud Computing Environment

Saurabh Bilgaiyan, Santwana Sagnika and Madhabananda Das
School of Computer Engineering, KIIT University, 751024 Bhubaneswar, India

Abstract: As the world is progressing towards faster and more efficient computing techniques, cloud computing has emerged as an efficient and cheaper solution to such increasing and demanding requirements. Cloud computing is a computing model which facilitates not only the end-users but also organizational and other enterprise users with high availability of resources on demand basis. This involves the use of scientific workflows that require large amount of data processing which can be costly and time-consuming if not properly scheduled in cloud environment. Thus, scheduling has a great impact on both cloud service providers and users. A properly scheduled service benefits both parties. Various scheduling strategies have been developed which include swarm-based optimization approaches as well. Due to the presence of multiple and conflicting requirements of users, multi-objective optimization techniques have become popular for workflow scheduling. This study deals with Cat-swarm based multi-objective optimization approach to schedule workflows in cloud computing environment. The objectives considered are minimization of cost, makespan and CPU idle time. Researchers have implemented this technique and compared the experimental results with existing Multi-Objective Particle Swarm Optimization (MOPSO) technique and have obtained improved performance.

Key words: Cloud computing, workflow scheduling, Multi-Objective Cat Swarm Optimization (MOCSO), cost minimization, makespan, CPU idle time

INTRODUCTION

In the last two decades, online computing services have become more popular than traditional offline computing services. Cloud computing has emerged as a dominant processing environment which has made it uncomplicated for demanding users to access their required services from among a wide variety of available and configurable computing resources. It shifts computation and information from client machines to over the networks where cloud service providers are connected (Buyya *et al*., 2009; Dikaiakos *et al*., 2009). Cloud environment provides services in the form of Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). Any type of user can opt for their required services from the available pool of configurable resources by simply connecting through the internet to the service vendor, e.g., Amazon Web Services, GOGRID, VERIO, MOSSO, etc. (Chaisiri *et al*., 2012). The benefits of using these services arise out of direct accessibility from any geographical location via a simple internet connection with normal speed as well as usage charges being levied in proportion to only the services used (Murugesan, 2011; Garg *et al*., 2013).

On the basis of types of users and their needs, cloud systems are classified into four categories; public cloud which provides open service access to every type of user, private cloud which provides service access within a particular organization, community cloud which is used by multiple organizations sharing similar needs and hybrid cloud which combines services provided by different cloud systems (Singh, 2012). Cloud computing environment overcomes the limitations due to the actual number of available physical machines by using the most important concept of virtualization that provides an abstraction between machines and users so that single machines can be shared among a number of users, thereby making the whole scenario transparent, flexible and robust (Li *et al*., 2012; Iosup *et al*., 2011; Sim, 2012). However, a major area of concern involves security and trust issues, since the users' sensitive data is processed and stored at physical locations unknown to users (Shaikh and Haider, 2011; Almorsy *et al*., 2011).

Scheduling of tasks on resources in a network-based computing environment such as cloud computing is always a challenging task. It is the process of allocating limited number of resources among a set of tasks that require the services of these resources. The main aim of

Corresponding Author: Saurabh Bilgaiyan, School of Computer Engineering, KIIT University, 751024 Bhubaneswar, India

scheduling is to minimize the cost and time of task completion while maximizing the Quality of Services (QoS). Task scheduling in cloud computing is termed as a NP-complete problem because of dynamic nature of cloud environment (Fard *et al*., 2013; Jangra and Saini, 2013).

Most real-life scheduling applications require scheduling which satisfies multiple objectives which are generally contradictory. Hence, the requirement arises to perform multi-objective scheduling that can satisfy such multiple objectives all together. For such problems, no single solution exists but generally a set of solutions that achieve tradeoff between the objectives can be found out. These set of solutions are known as non-dominated solutions and are represented graphically by Pareto front. Multi-objective workflow scheduling consists of a set of user-defined conflicting requirements (Fard *et al*., 2013; Wang *et al*., 2008).

Many optimization techniques exist in literature, like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Bee Colony Optimization (BCO), etc. Among them, PSO has proved to be one of the most efficient algorithms applicable to a wide variety of problems, due to ease of implementation and less adjustable parameters. It also proves advantageous in terms of swift convergence and simple computation even in case of multi-objective problems where there is a need to find multiple solutions that aim to simultaneously optimize different contrasting objectives. However, PSO exhibits limited accuracy during later stages and tendency to get trapped in local optimal points (Yin *et al*., 2006; Yang *et al*., 2007).

One of the newly emerging techniques is Cat Swarm Optimization (CSO) which represents the behaviour of cats in their search for food. They exhibit seeking and tracing modes which aim to minimize energy spent in searching. Experiments show that CSO achieves optimal solution in much less number of iterations than previous algorithms like PSO. CSO can also be applied to multi-objective problems (Kennedy and Eberhart, 1995; Chu *et al*., 2006).

This study aims to achieve optimization, considering the objectives as minimization of computation cost, makespan and CPU idle time. Researchers propose a multi-objective CSO technique to achieve scheduling. The proposed technique has been implemented and the results have been compared with multi-objective PSO technique. This method MOCSO achieved optimal results in lesser number of iterations than MOPSO. Good convergence is achieved as discussed in the results.

Literature review: Scientific workflows represent a set of computational tasks having dependencies between them. Different applications are modelled as workflows for computation (Gill *et al*., 2007). A major challenge is the allocation of these tasks in a manner to reduce the execution time and cost. The size of relocated data and the associated overhead both lead to a huge amount of data processing. So, various techniques have been discussed under this study for scheduling scientific workflows in cloud computing systems (Szabo *et al*., 2013).

A Look Ahead Genetic Algorithm (LAGA) has been proposed that implements RD (Reliability Driven) reputation and evaluates the resource reliability in distributed systems, hence, optimizing makespan and reliability for workflows.

ACO, BCO, GA and PSO have been applied to solve scheduling strategies in cloud systems that are market-oriented. ACO was found to show the best performance among all (Singh and Singh, 2013).

A multi-objective PSO Model has been designed to find an optimal solution minimize task execution cost, transfer time and task execution time. It finds best tradeoff and also effectively utilizes cloud resources and improves QoS (Ramezani *et al*., 2013).

A Heuristic algorithm for static workflow scheduling comprising energy consumption, makespan, reliability and economic cost is defined that performs better than Bi-Criteria Heuristic algorithms (Fard *et al*., 2012).

A GA-based Model is proposed which takes average VM, utilization, average simulation time, cost and no of tasks, average processing, response time as the optimization parameters. Every time when a scheduling request is arrived to the scheduler, it calls the GA function for each scheduling. This function makes a set of scheduling tasks to the available resources and assesses the QoS and according to satisfactory QoS it allocates the resources (Savitha and Reddy, 2013).

An immune GA is proposed for scheduling service workflows in cloud environment while considering QoS constraints as the optimization parameter. The scheduling heuristic considers multi-objective optimization. Here, cost, reliability, energy consumption, availability and makespan are considered as the optimization parameters. The proposed researches consist of objectives of conflicting types where researchers want to minimize some of them, i.e., cost, makespan, energy consumption, etc. while maximization of availability and reliability is required. So, as for different purposes the fitness function is divided into two categories where one is for minimizing and other is for maximizing the target values. The assessment and generating process has a two-stage process where in the first stage the GA generates the solution vectors and then it assigns each task to resource. Then, in the next step GA assesses the fitness values of the assigned solution vectors. The evaluation of solution

vector is done by using HEFT (Heterogeneous Earliest Finish Time) Model. The experimental result shows the better performance of proposed algorithm than that of other existing methods (Sellami *et al*., 2013).

A PSO algorithm is proposed for batch processing workflow scheduling. In this proposed technique GD (Generational Distance) and execution time is taken as a performance measures where GD represents the proximity between obtained solution and actual Pareto solution. In this proposed research, a substitute Deep Memory with Particle Swarm Optimization (DMPSO) is implemented using Dynamic Grouping and Scheduling Optimization (DGSO) and standard PSO. The experimental results show that DGSO gives better searching and average GD while the execution time of the algorithm increases with the number of iterations (Wen *et al*., 2012; Shi and Eberhart, 1998).

An Artificial Bee Colony Optimization Algorithm is proposed for workflow scheduling in cloud computing environment. The proposed algorithm optimizes the computation time and server utilization. The experiment is done using cloudsim tool and results are compared with existing GA. The proposed technique shows a better performance over the GA (Kumar and Anand, 2013).

A new technique named as Multiple Pheromone Algorithm (MPA) from the family of ant colony optimization is proposed. The aim of Proposed algorithm is to dynamically schedule the task in such a way so that the task completion time will be minimum while the resource utilization will be maximized. Makespan, cost and reliability are the QoS taken as optimization measures. Experimental results show better performance of MPA algorithm over ACO and GA (Gogulan *et al*., 2012).

An Ant Colony Optimization algorithm (ACO) is introduced for task scheduling in cloud computing environment. The basic ACO is enhanced for minimizing the execution time and makespan of the all scheduled tasks. Experiment is carried out using cloudsim tool kit and results show the better performance of extended ACO over existing simple ACO (Tawfeek *et al*., 2013).

MATERIALS AND METHODS

Workflow design: The increase in size and complexity of scientific and industrial applications in recent times has brought about the need to represent these problems in the form of workflows that can run on distributed settings. Real-time mapping of these tasks on available resources dynamically is a challenging task which requires efficient algorithmic solutions (Fard *et al*., 2013). So, to address this issue, in this study the researchers explore the usability of swarm-based optimization mechanism to achieve efficient scheduling of workflows while satisfying multiple objectives that are the major requirements of users in present scenario.

Basic workflow structure: Researchers have characterized a general workflow as a Directed Acyclic Graph (DAG). The DAG is denoted by G = (W, D). W represents a set of tasks in the workflow where W = {W_1, W_2, W_3, ..., W_n}. D represents the dependencies among these tasks. These tasks need to be mapped on a set of available resources S = {S_1, S_2, S_3, ..., S_n}, all geographically distributed throughout the world. Each resource has its own storage/memory, designated as M = {M_1, M_2, M_3, ..., M_n}.

The execution time and cost of every task on all resources is known as per a hypothetically assumed pricing policy within range defined by some policies of Gogrid and Amazon Web Services. The cost of transferring data between tasks is also known.

Figure 1 shows a Sample Workflow Model having 14 tasks along with their inter-dependencies that the researchers have used for experimenting. The tasks are represented by W_1, W_2, ..., W_{14} and their dependencies are shown as $d_{i, j}$ which means a dependency on task W_j on W_i.

In Fig. 2, the available 4 resources are depicted in the form of S_1, ..., S_4 with the per unit data transfer cost betweenany two resources S_i and S_j is denoted by $tc_{Si, Sj}$. Size of transferred data is assumed to be constant.

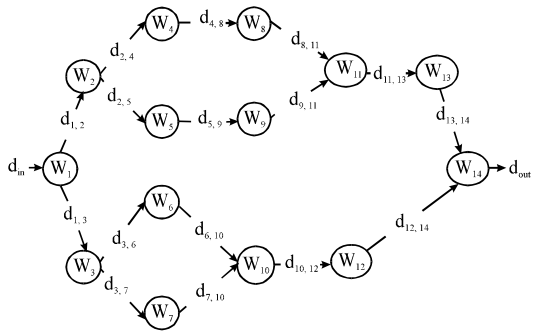

Fig. 1: A sample experimental workflow

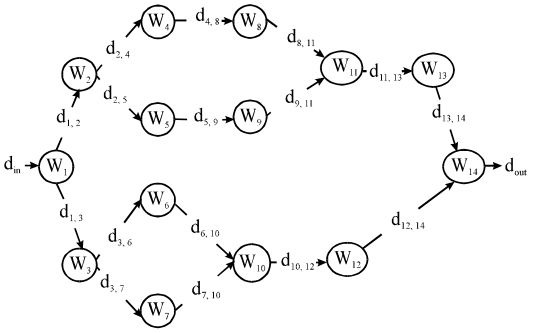

Fig. 2: A sample resource distribution architecture with local storages

Multi-Objective Scheduling Model: QoS is given high priority in providing services over the cloud. Hence, the scheduling needs to be done in a suitable manner to ensure good QoS. In this problem, the researchers incorporate the objectives of minimum cost, minimum makespan and minimum CPU idle time as factors comprising QoS. Thus, it becomes a multi-objective scheduling problem. Hence, this approach aims to find a Pareto solution set that represents multiple non-dominated solutions satisfying these specified objectives (Leong and Yen, 2008; Li *et al*., 2008). A general multi-objective minimization problem can be specified as:

$$\text{Minimize } h(x) = (h_1(x), h_2(x), \ldots, h_n(x)) \text{ such that } x \in X \quad (1)$$

Here:

$h_1, \ldots, h_n$ = The different objective functions
x = Any feasible solution in the solution space X

Any solution p is said to dominate q if p is at least as good as q in all objectives and better in one or more objectives. The aim here is to find the set of non-dominated solutions.

Mathematical representation of scheduling goals: Researchers represent the current multi-objective problem in the following mathematical format:

$$T\_cost(S_x) = \sum_y \left(ec_{yx} + \sum_z tC_{yz} \right), \quad \forall\, Sch(W_y) = S_x \text{ and } Sch(W_z) \neq S_x \quad (2)$$

Where:

$T\_cost(S_x)$ = The total cost of executing tasks assigned to resource S_x and transmitting data to other tasks dependent on those tasks
ec_{yx} = The cost of executing task W_y on S_x
Tc_{yz} = Transmission cost between tasks W_y and W_z which are on different resources
Sch = A specific scheduling map
$Sch(W_y)$ = The resource where, W_y is being executed

$$Max\_cost(Sch) = Max(T\_cost(S_x)), \forall S_x \quad (3)$$

where, Max_cost(Sch) signifies that the tasks are fairly distributed over resources:

$$T\_time(S_x) = \sum_y tt_{yx} \quad (4)$$

where, $T\_time(S_x)$ signifies total time of executing all tasks running on S_x. Here, tt_{yx} gives the total running time of task W_y on S_x:

$$Makespan(Sch) = max(T\_time(S_x)), \forall S_x \quad (5)$$

where, Makespan(Sch) indicates the total time between start and finish of total schedule:

$$T\_idle(Sch) = \sum_x (Makespan(Sch) - T\_time(S_x)) \quad (6)$$

where, T_idle(Sch) indicates the amount of time the resources remain idle, found out by summation of the idle time of each resource S_x till all tasks are completed:

$$\text{Minimize}(Max\_cost(Sch), Makespan(Sch), T\_idle(Sch)), \forall\, Sch \quad (7)$$

Equation 7 is the multi-objective fitness function for current scheduling problem that gives optimal solution which minimizes cost, makespan and CPU idle time.

General CSO algorithm: The common behaviour of cats in real-world has inspired the development of a new swarm-based optimization technique known as Cat Swarm Optimization (CSO) as proposed by Chu and Tsai (2007). The activity of cats in general includes spending maximum time resting but alert. While doing so, their movements are slow and calculated. This is referred to as seeking mode. On the other hand while chasing targets, they move with high velocity, converging towards the target. This is called as tracing mode. By switching between these two modes asrequired, cats reduce spending a lot of energy which achieves solutions faster and in a more intelligent manner (Shojaee *et al*., 2012; Sharafi *et al*., 2013; Chu and Tsai, 2007).

The CSO optimization technique makes use of this behaviour of cats to search complex solution spaces for optimal solutions using an initial population of cats that are randomly divided into seeking and tracing modes as per a defined Mixture Ratio (MR) and each cat performs its actions as per the mode it is in. The cats move closer to solutions by updating best results in the memory. This process continues iteratively by redistributing cats into either mode each time, till all cats achieve best solution. A universal CSO algorithm can be described as follows (Pradhan and Panda, 2012; Santosa and Ningrum, 2009; Tsai *et al*., 2008):

CSO algorithm:

1. Generate N cats over required number of dimensions D
2. Allocate random velocities to all cats
3. Randomly distribute cats to seeking and tracing modes as per defined MR
4. Calculate fitness of all cats and memorize the non-dominated cats
5. For each cat, if cat is in seeking mode, perform seeking mode operations, else perform tracing mode operations on it and move it to its new position
6. If termination condition is not satisfied, goto step 3, else stop

Proposed research: In the proposed research, researchers discuss the MOCSO algorithm and its two working modes, namely seeking mode and tracing mode followed by the experimental environment setup details section that includes the performance benchmarks and the input data taken.

Algorithm: Researchers put forward an approach that utilizes this CSO technique to address the multi-objective scheduling problem as described in study. The different tasks of a workflow are represented by dimensions and each cat denotes a schedule between the set of tasks and available resources.

In each iteration these cats are divided into seeking and tracing modes randomly and the positions of these mappings are updated differently as per the current mode. This process continues till the best schedule is achieved or termination criteria is reached. Here, best schedule refers to a set of non-dominated optimal schedules that achieve balance between the required objectives, namely computation cost, makespan and CPU idle time. These various solutions form a graph which is known as Pareto front. Any among these solutions can be selected as the required mapping.

Seeking mode: Seeking mode represents the resting condition of cats wherein they remain alert and look around to their surroundings. This mode uses certain parameters that determine a cat's behavior. Those are.

Seeking Memory Pool (SMP): The number of replicas for each cat that are to be made. One among them will replace the original cat later.

Count of Dimension to Change (CDC): The number of allotments in each copy that will be modified. Each copy will be evaluated and one of the best will be selected for replacement. When a cat is in seeking mode, it performs the following steps:

Seeking mode steps:

- Generate replicas of the cat as per SMP
- Randomly change dimensions of each replica as per CDC
- Assess fitness values of all replicas
- Find the best non-dominated replicas
- Substitute the cat with a randomly selected non-dominated replica

Tracing mode: Tracing mode depicts movement of cats towards targets with high velocity while spending high amount of energy. In tracing mode a cat performs the following steps:

Tracing mode steps:

- Find the new velocity $V_{i\_d}^{t+1}$ for cat i by using the equation:

$$V_{i\_d}^{t+1} = w \times V_{i\_d}^{t} + c \times r \times (X_{best\_d} - X_{i\_d}^{t}) \quad (8)$$

Where:

$V_{i\_d}^{t}$ = The velocity at tth iteration
$X_{i\_d}^{t}$ = The position in tth iteration
X_{best_d} = The current global best position in dth dimension
c = A constant
r = A random number between 0 and 1

- Change cats to new position to the next best position $X_{i\_d}^{t+1}$ by adding new velocity as per:

$$X_{i\_d}^{t+1} = X_{i\_d}^{t} + V_{i\_d}^{t+1} \quad (9)$$

- Limit the updated position of cat within desired range
- Calculate fitness of all cats
- Update the set of solutions with non-dominated cats

Figure 3 represents the complete algorithm flow.

Setting up of experimental environment

Evaluation benchmarks: Following are the evaluation criteria taken up to judge the performance of multiobjective optimization:

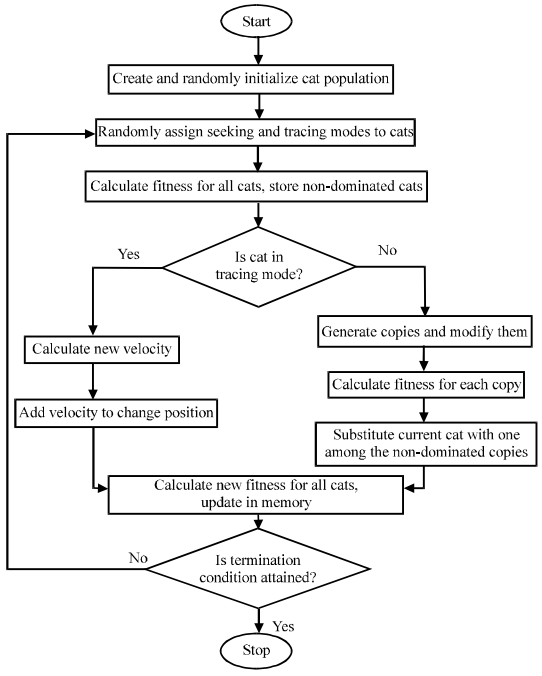

Fig. 3: Basic steps of MOCSO

- To find solutions closest to the Pareto front
- To find solutions widely distributed over the Pareto front
- To find solutions in minimum number of iterations

Input data for experiments: Researchers have assumed a hypothetical workflow having 14 tasks to be distributed over 4 resources that are located in different countries across the world as illustrated previously. The cost of execution and communication are preassumed on the basis of pricing policies followed by some well-known service providers, i.e., Amazon Web Services, Mosso, Gogrid, etc. The execution time for each task on each resource is preassumed as suitable for experimenting. The experiments have been performed and resultant graphs generated using MATLAB as a tool. Following are the input data tables where Table 1 represents execution time for tasks on resources, Table 2 gives communication costs between different resources and Table 3 denotes execution time of each task on each resource.

Table 1: Execution cost matrix (cents)

Values	S_1	S_2	S_3	S_4
T_1	1.56	1.59	1.59	1.72
T_2	1.83	2.01	2.24	2.14
T_3	1.65	1.68	1.71	1.82
T_4	2.21	2.00	2.33	2.48
T_5	2.11	2.03	1.98	1.87
T_6	1.79	2.19	1.63	1.50
T_7	1.50	1.57	1.88	1.72
T_8	2.50	2.03	2.16	2.42
T_9	1.57	1.93	1.82	1.75
T_{10}	2.19	2.03	2.25	2.36
T_{11}	1.50	1.83	1.76	2.23
T_{12}	2.31	2.98	2.50	2.27
T_{13}	1.52	1.93	1.61	1.74
T_{14}	2.39	2.14	2.04	1.96

Table 2: Communication cost matrix (cents/MB)

Cost	S_1	S_2	S_3	S_4
S_1	0.00	0.12	0.17	0.07
S_2	0.12	0.00	0.20	0.11
S_3	0.17	0.20	0.00	0.13
S_4	0.07	0.11	0.13	0.00

Table 3: Execution time matrix (h)

Time (h)	S_1	S_2	S_3	S_4
T_1	0.51	0.44	0.31	0.44
T_2	0.93	0.84	0.46	0.84
T_3	0.75	0.66	0.52	0.66
T_4	1.20	1.15	1.11	1.15
T_5	1.08	0.92	0.78	0.92
T_6	0.63	0.59	0.52	0.59
T_7	0.32	0.24	0.10	0.24
T_8	0.91	2.78	0.62	0.78
T_9	0.74	0.66	0.91	0.66
T_{10}	0.42	0.36	2.23	0.36
T_{11}	0.55	0.50	0.40	0.50
T_{12}	0.88	0.72	0.53	0.72
T_{13}	0.61	0.54	0.49	0.54
T_{14}	1.16	0.98	0.81	0.98

RESULTS AND DISCUSSION

Researchers have performed various experiments using the proposed algorithm on MATLAB tools and have compared the results with existing MOPSO algorithm for same problem. The population size was preset at 50 while the number of iterations has been varied from 100-300.

Analysis of the resultant graphs: Figure 4-8 show pairs of graphs where in each Fig. 4-8a represents Pareto front obtained by MOPSO and Fig. 4-8b represents Pareto front obtained by MOCSO for a particular number of iterations. By comparing the graph pairs, it can be observed that MOCSO approach generally provides more number of solutions than MOPSO for a fixed number of iterations which are also closer to and better distributed over the Pareto front. On analyzing the results for different iterations, it can be seen that MOCSO achieves good convergence and faster attainment of a convex Pareto front as compared to MOPSO. The rationale behind better performance of MOCSO can be attributed to the fact that it exhibits intelligent updating of positions rather than the random updating mechanism followed by MOPSO that saves energy and hence increases speed and efficiency of reaching at best solutions.

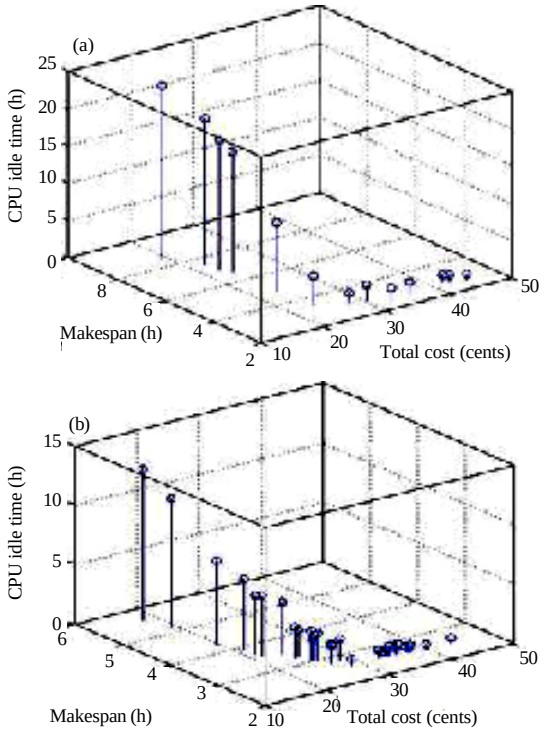

Fig. 4: Optimal results for 100 iterations; a) MOPSO and b) MOCSO

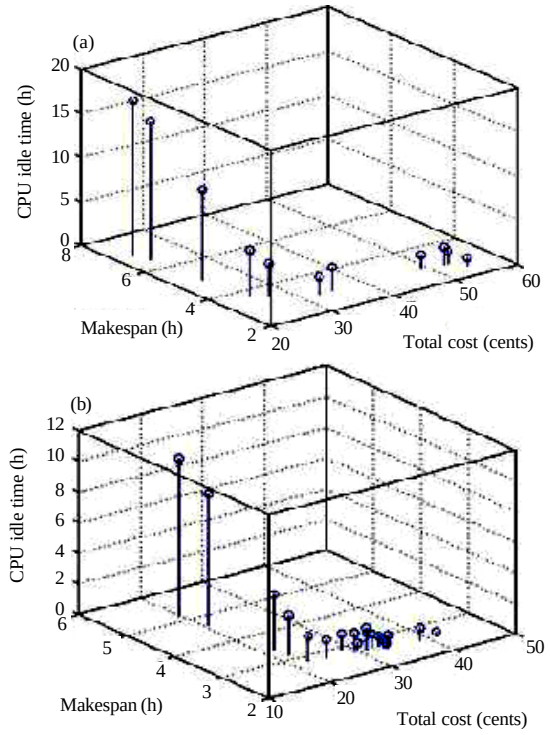

Fig. 5: Optimal results for 150 iterations; a) MOPSO and b) MOCSO

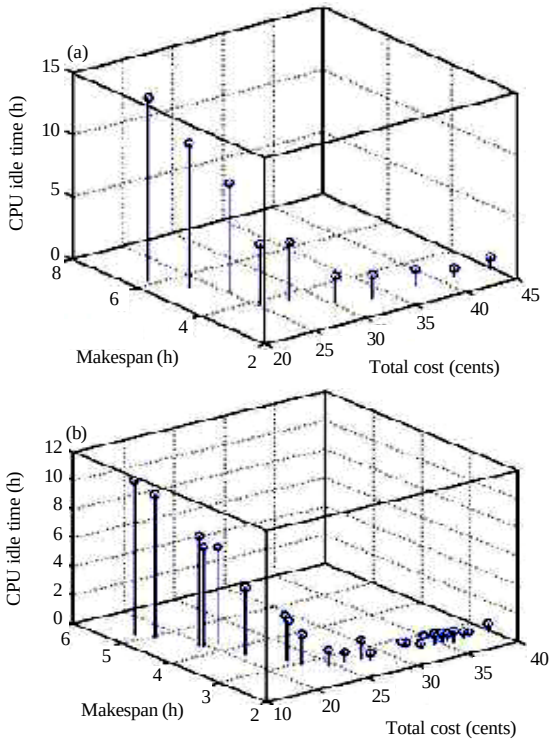

Fig. 7: Optimal results for 250 iterations; a) MOPSO and b) MOCSO

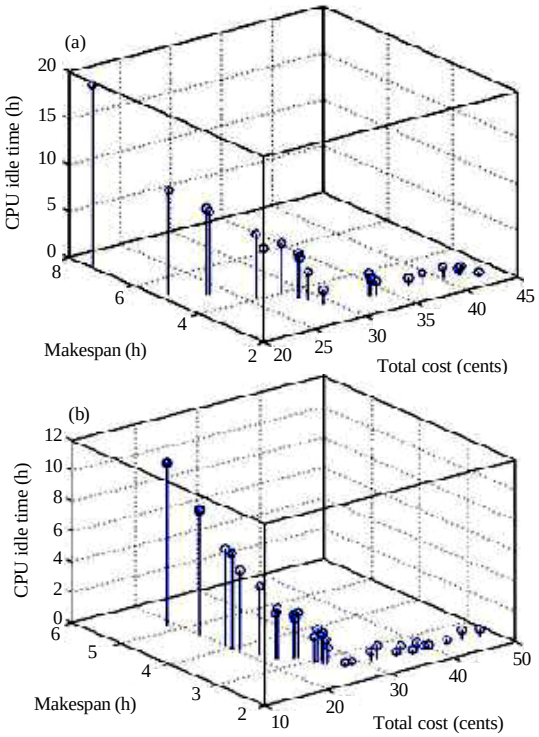

Fig. 6: Optimal results for 200 iterations; a) MOPSO and b) MOCSO

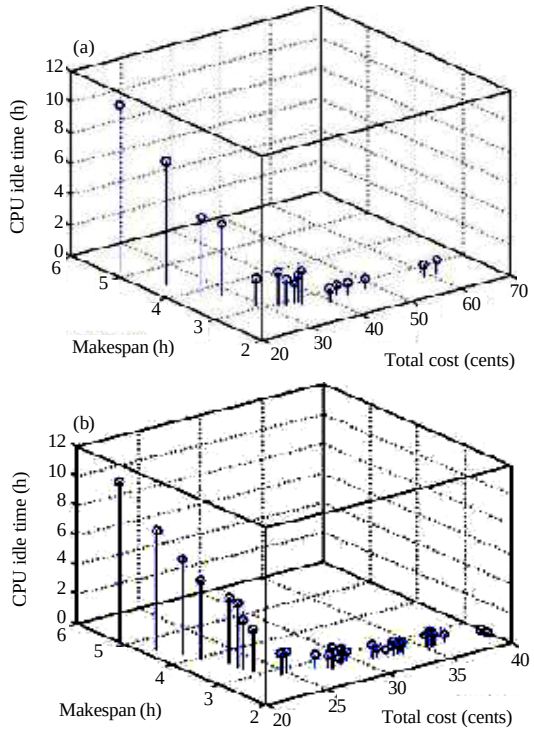

Fig. 8: Optimal results for 300 iterations; a) MOPSO and b) MOCSO

CONCLUSION

This study has presented a new and more efficient approach to use a swarm-based technique to solve the multi-objective scheduling problem in a cloud computing environment. This technique has proved to be faster and highly convergent over existing MOPSO technique which is already among the Dominant Optimization Methods. This is owing to the smart mechanism of position updating in MOCSO that reduces unnecessary energy expenditure and moves closer towards the solution in each iteration. Future scope in this field can consist of improving the running time of this algorithm and involving more number of real-time conflicting objectives which researchers believe can be effectively handled by this technique. Finding competent solutions to the workflow scheduling problem on cloud can increase the QoS and extend the reach of cloud computing over even more business and enterprise areas.

REFERENCES

Almorsy, M., J. Grundy and A.S. Ibrahim, 2011. Collaboration-based cloud computing security management framework. Proceedings of the 4th IEEE International Conference on Cloud Computing, July 4-9, 2011, Washington, DC., pp: 364-371.

Buyya, R., C.S. Yeo, S. Venugopal, J. Broberg and I. Brandic, 2009. Cloud computing and emerging IT platforms: Vision, hype and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst., 25: 599-616.

Chaisiri, S., B.S. Lee and D. Niyato, 2012. Optimization of resource provisioning cost in cloud computing. IEEE Trans. Servic. Comput., 5: 164-177.

Chu, S.C. and P.W. Tsai, 2007. Computational intelligence based on the behavior of cats. Int. J. Innovat. Comput. Inform. Control, 3: 163-173.

Chu, S.C., P.W. Tsai and J.S. Pan, 2006. Cat swarm optimization. Proceedings of the 9th Pacific Rim international conference on Artificial intelligence, August 7-11, 2006, China, pp: 854-858.

Dikaiakos, M.D., D. Katsaros, P. Mehra and G. Pallis and A. Vakali, 2009. Cloud Computing: Distributed internet computing for it and scientific research. IEEE Comput. Soc., 13: 10-13.

Fard, H.M., R. Prodan and T. Fahringer, 2013. A truthful dynamic workflow scheduling mechanism for commercial multicloud environments. IEEE Trans. Parallel Distribution Syst., 24: 1203-1212.

Fard, H.M., R. Prodan, J.J.D. Barrionuevo and T. Fahringer, 2012. A multi-objective approach for workflow scheduling in heterogeneous environments. Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, May 13-16, 2012, Ottawa, ON, pp: 300-309.

Garg, S.K., S. Versteeg and R. Buyya, 2013. A framework for ranking of cloud computing services. Future Gener. Comput. Syst., 29: 1012-1023.

Gil, Y., E. Deelman, M. Ellisman, T. Fahringer and G. Fox *et al*., 2007. Examining the challenges of scientific workflows. IEEE Comput. Soc., 40: 24-32.

Gogulan, R., A. Kavitha and U.K. Kumar, 2012. An multiple pheromone algorithm for cloud scheduling with various QOS requirements. Int. J. Comput. Sci. Issu., 9: 232-238.

Iosup, A., S. Ostermann, M.N. Yigitbasi, R. Prodan, T. Fahringer and D.H. Epema, 2011. Performance analysis of cloud computing services for many-tasks scientific computing. IEEE Trans. Parallel Distrib. Syst., 22: 931-945.

Jangra, A. and T. Saini, 2013. Scheduling optimization in cloud computing. Int. J. Adv. Res. Comput. Sci. Software Eng., 3: 62-65.

Kennedy, J. and R. Eberhart, 1995. Particle swarm optimization. Proceedings of the International Conference on Neural Networks, Volume 4, November 27-December 1, 1995, Perth, WA., USA., pp: 1942-1948.

Kumar, P. and S. Anand, 2013. An approach to optimize workflow scheduling for cloud computing environment. J. Theoret. Applied Inform. Technol., 57: 617-623.

Leong, W.F. and G.G. Yen, 2008. PSO-based multiobjective optimization with dynamic population size and adaptive local archives. IEEE Trans. Syst. Man Cybernetics, 38: 1270-1293.

Li, B.B., L. Wang and B. Liu, 2008. An effective PSO-based hybrid algorithm for multiobjective permutation flow shop scheduling. IEEE Trans. Syst. Man Cybernetics, 38: 818-831.

Li, J., M. Qiu, Z. Ming, G. Quan, X. Qin and Z. Gu, 2012. Online optimization for scheduling preemptable tasks on IaaS cloud systems. J. Parallel Distri. Comput., 72: 666-677.

Murugesan, S., 2011. Cloud computing gives emerging markets a lift. IT Professional, 13: 60-62.

Pradhan, P.M. and G. Panda, 2012. Solving multiobjective problems using cat swarm optimization. Expert Syst. Applic., 39: 2956-2964.

Ramezani, F., J. Lu and F. Hussain, 2013. Task scheduling optimization in cloud computing applying multi-objective particle swarm optimization. Proceedings of the 11th International Conference, Volume 8274, December 2-5, 2013, Berlin, Germany, pp: 237-251.

Santosa, B. and M.K. Ningrum, 2009. Cat swarm optimization for clustering. Cat swarm optimization for clustering. Proceedings of the IEEE International Conference on Soft Computing and Pattern Recognition, December 4-7, 2009, Malacca, pp: 54-59.

Savitha, P. and J.G. Reddy, 2013. A review work on task scheduling in cloud computing using genetic algorithm. Int. J. Sci. Technol. Res., 2: 241-245.

Sellami, K., M. Ahmed-Nacer, P.F. Tiako and R. Chelouah, 2013. Immune genetic algorithm for scheduling service workflows with Qos constraints in cloud computing. S. Afr. J. Ind. Eng., 24: 68-82.

Shaikh, F.B. and S. Haider, 2011. Security threats in cloud computing. Proceedings of the International Conference on Internet Technology and Secured Transactions, December 11-14, 2011, Abu Dhabi, pp: 214-219.

Sharafi, Y., M.A. Khanesar and M. Teshnehlab, 2013. Discrete binary cat swarm optimization algorithm. Proceedings of the 3rd IEEE International Conference on Computer, Control and Communication, September 25-26, 2013, Karachi, pp: 1-6.

Shi, Y. and R. Eberhart, 1998. A modified particle swarm optimizer. Proceedings of the World Congress on Computational Intelligence and IEEE International Conference on Evolutionary Computation, May 4-9, 1998, Anchorage, AK., pp: 69-73.

Shojaee, R., H.R. Faragardi, S. Alaee and N. Yazdani, 2012. A new cat swarm optimization based algorithm for reliability-oriented task allocation in distributed systems. Proceedings of the 6th IEEE International Symposium on Telecommunications, November 6-8, 2012 Tehran, pp: 861-866.

Sim, K.M., 2012. Agent-based cloud computing. IEEE Trans. Services Comput., 5: 564-577.

Singh, H., 2012. Current trends in cloud computing a survey of cloud computing systems. Int. J. Electron. Comput. Sci. Eng., 1: 1214-1219.

Singh, L. and S. Singh, 2013. A survey of workflow scheduling algorithms and research issues. Int. J. Comput. Applic., 74: 21-28.

Szabo, C., Q.Z. Sheng, T. Kroeger, Y. Zhang and J. Yu, 2013. Science in the cloud: Allocation and execution of data-intensive scientific workflows. J. Grid Comput. 10.1007%2Fs10723-013-9282-3.

Tawfeek, M.A., A. El-Sisi, A.E. Keshk and F.A. Torkey, 2013. An ant algorithm for cloud task scheduling. Proceedings of the International Workshop on Cloud Computing and Information Security, November 9-11, 2013, Shanghai, China, pp: 169-172.

Tsai, P.W., J.S. Pan, S.M. Chen, B.Y. Liao and S.P. Hao, 2008. Parallel cat swarm optimization. Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, July 12-15, 2008, Kunming, pp: 3328-3333.

Wang, X.J., C.Y. Zhang, L. Gao and P.G. Li, 2008. A survey and future trend of study on multi-objective scheduling. Proceedings of the 4th IEEE International Conference ON Natural Computation, October 18-20, 2008, Jinan, pp: 382-391.

Wen, Y., Z. Chen, T. Chen, J. Liu and G. Kang, 2012. A particle swarm optimization algorithm for batch processing workflow scheduling. Proceedings of the 2nd IEEE International Conference on Cloud and Green Computing, November 1-3, 2012, Xiangtan, pp: 645-649.

Yang, X., J. Yuan, J. Yuan and H. Mao, 2007. A modified particle swarm optimizer with dynamic adaptation. Applied Math. Comput., 189: 1205-1213.

Yin, P.Y., S.S. Yu, P.P. Wang and Y.T. Wang, 2006. A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems. Comput. Standards Interfaces, 28: 441-450.