ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 8(16), DOI: 10.17485/ijst/2015/v8i16/63248, July 2015* ISSN (Online) : 0974-5645

PBCOPSO: A Parallel Optimization Algorithm for Task Scheduling in Cloud Environment

R. Jemina Priyadarsini* **and L. Arockiam**

Department of Computer Science, St. Joseph's College, Trichirapalli - 620002, Tamil Nadu, India; jemititus@gmail.com, larockiam@yahoo.com

Abstract

Objectives: Cloud environment requires scheduling of independent tasks with the available resources to minimize the total execution time and to optimize the resource utilization in cloud environment. **Methods**: Evolutionary algorithms are widely used to find the suboptimal solution of a problem. This work adopts a parallel approach that considers Bee Colony Optimization (BCO) in parallel with Particle Swarm Optimization (PSO) for cloud task scheduling. The proposed approach is named as Parallel Bee Colony Optimization Particle Swarm Optimization (PBCOPSO). **Findings**: The results show that the proposed approach minimizes Makespan with optimized resource utilization. It is observed that the proposed method improved resource utilization by an average of 5.0383% when compared with Min-Min algorithm and by an average of 3.7243% when compared with Improved Bee Colony Optimization (IBCO). **Novelty of the Study:** The proposed hybrid PBCOPSO enables improved search in the solution space due to the parallel execution of BCO and PSO leading to better final solution quality and lower execution time. **Conclusion:** Thus two metrics namely makespan and resource utilization are evaluated and an optimal task to resource mapping is achieved with hybridization.

Keywords: Bee Colony Optimization, Optimization, Particle Swarm Optimization, Resource Utilization, Scheduling

1. Introduction

To increase cloud computing work load efficiency, scheduling is a task performed to get maximum profit[1]. Task scheduling is an important and critical problem in cloud computing and researchers tried to find an optimal solution for scheduling tasks on current resources in a cloud environment. Task scheduling is a NP-complete problem, generally[2].

Presently, heuristic optimization algorithms are used to solve NP-complete problems. An evolutionary algorithm finds a sub-optimal solution to a problem in computation time. To get a faster solution many heuristic based approaches are used. Also optimized solution near a main solution is found. Optimal task scheduling minimizes task running and communication costs using branch-and-bound technique and through simulation methods which help calculate this method's computational complexity. Conventional methods in optimization are deterministic, fast providing exact answers but often stuck in local optima.

Task scheduling is necessary to increase cloud computing working, to gain maximum profit, improve reliability, and system flexibility. The cloud environment's key technologies are resource management and task scheduling. Previously designed task scheduling algorithms consider QoS but do not consider how to optimize resources use to gain maximum profit[3].

To solve NP-complete problems, evolutionary and heuristic algorithms are used, and for distributed systems scheduling problems algorithms like Partial Swarm Optimization (PSO), Simulated Annealing, Tabu Search and Genetic Algorithm (GA) are used. For example, in heuristic Min-Min and Max-Min algorithms that are

* *Author for correspondence*

begun by a batch of unmapped tasks, a task's execution time on every processor[4] is computed. Ant Colony Optimization algorithm solves optimization problems to find good paths. The aim is to achieve minimum weighted flow time for jobs and minimizing makespan[5].

The challenge of optimization methods is finding a global optimal. In Bee swarm Algorithm, tasks are chosen randomly for first algorithm. Then, makespan is calculated for that set of tasks. Finally the algorithm checks for stopping criteria[6]. Artificial Bee Colony algorithm (ABC) solves multi-dimensional and multi-modal optimization problems. It is robust for multimodal problems, as it has multi agents working independently and in parallel. Bee Swarm Optimization (BSO) is a meta-heuristic as it represents a general algorithmic framework applicable to various optimization problems. It belongs to a class of population-based algorithms.

PSO is a population based search algorithm initialized with a population of random solutions, called particles[7]. Unlike evolutionary computation techniques, each PSO particle is associated with a velocity. This study uses BCO in parallel with PSO for scheduling. The rest of this study has the following sections: Section two reviews related work. Section three explains methods used in this work. Section four discusses experimental results and Section five concludes with the proposed work.

A Multi Queue Scheduling (MQS) algorithm to reduce the cost of reservation and on-demand plans using a global scheduler was proposed by Karthick et al[8]. Scheduling is an important complex part in cloud computing. The new methodology depicted the concept of jobs clustering based on burst time. The new MQS method gives more importance to select jobs dynamically to achieve optimum cloud scheduling and so used unused free space economically.

A Tri Queue Scheduling (TQS) algorithm for cloud environment was proposed by Karthick et al[9]. Resource sharing is a challenge in cloud computing. Resource sharing efficiency depends on the meta-scheduler. The methodology grouped small, medium and long jobs in a queue, based on the processor needed and time for resources to computers. A system which gives importance to all jobs using dynamic quantum time based Round Robin Fashion was proposed. To make efficient use of resources, the new scheduling algorithm achieved cloud scheduling problems optimization.

A scheduling algorithm that gives better throughput and viable communication costs compared to first-come-first-serve scheduling algorithm in a cloud computing environment was focused on by Leena et al[10]. The algorithm was implemented across clouds and had better response times.

An optimized scheduling algorithm to achieve optimization or sub-optimization for cloud scheduling problems was proposed earlier[11]. The possibility of placing Virtual Machines flexibly was investigated to improve speed of finding best allocation to ensure maximum resource use. The new scheduling policy achieved by Parallel GA was faster than traditional GA. Experiments showed that the new method improved speed of resources allocation and system resource use.

An optimized scheduling algorithm to achieve optimization or sub-optimization for cloud scheduling problems was proposed in[12]. An Improved GA was used for automated scheduling. IGA uses shortest genes and introduced the idea of Dividend Policy in Economics to select an optimal or suboptimal allocation for VMs requests. Simulation indicated that the suggested dynamic scheduling policy performed much better than Open Nebula, Eucalyptus, Nimbus IaaS cloud, etc. Tests illustrated that IGA's speed was almost twice that of traditional GA scheduling in Grid environment and the usage rate of resources was higher than open-source IaaS cloud systems.

A task scheduling algorithm in a heterogeneous multi-cloud environment proposed by Panda and Jana[13] was based on two algorithms namely, Min-Min and Max-Min. Experiments were performed on some benchmark/synthetic data sets and the results were compared with two existing multi-cloud scheduling algorithms. They showed that the new algorithm outperformed both algorithms in makespan and average cloud usage.

A hybrid task scheduling algorithm that combined the plus points of bio-inspired algorithms like ACO and ABA was proposed by Madivi and Kamath[14] where the strong points of both algorithms was used and included to optimize task scheduling in a cloud algorithm. It was seen that the new algorithm improved by about 19% compared to default FCFS scheduling strategy, 11% better than ABC algorithm and performed 9% better than conventional ACO based task scheduling.

An efficient task scheduling approach that combines Bee Colony Optimization with Hill Climbing local search was proposed by Jemina et al.[15] where the results shows

the effectiveness of hybridization and also the algorithm works in a sequential manner with BCO and Hill-Climbing approach one after the other.

2. Methodology

In this paper, Parallel Bee Colony Optimization Particle Swarm Optimization (PBCOPSO), with BCO in parallel with PSO for scheduling is proposed. The proposed hybrid PBCOPSO enables improved search in the solution space due to the parallel execution of BCO and PSO leading to better final solution quality and lower execution time. In this section, the BCO, PSO and the proposed hybrid methods are detailed.

2.1 Bee Colony Optimization (BCO)

BCO algorithms solve problems of various domains, like routing problems, Traveling Salesman Problem and NP hard problems. Some problems are solved with BCO concept and others with ABC algorithms[16,17]. There is a very thin distinction among variants of the Bee system as the agent in all algorithms is a bee. BCO was also framed comprising of initialization, forward pass and backward pass steps. Forward and backward passes are performed till a stopping criterion is met. Then the bees search for an optimal solution. The steps in BCO:

(1) *Initialization. Number of bees B, and the number of iterations I.*

(2) *Select the set of stages* $ST = \{st_1, st_2, ..., st_m\}$.
Find any feasible solution x of the problem. This solution is the initial best solution.

(3) *Set* $i := 1$. *Until* $i = I$, *repeat the following steps:*

(4) *Set* $j = 1$. *Until* $j = m$, *repeat the following steps:*

(5) *Forward pass: Allow bees to fly from the hive*
and to choose B partial solutions from the set of partial solutions S_j *at stage* st_j.

(6) *Backward pass: Send all bees back to hive.*
Exchange information about partial solutions quality created
and to decide whether to abandon created partial solution
and become an uncommitted follower again,
continue to expand same partial solution without recruiting nestmates,
or dance and recruit nestmates before returning to created partial solution.

(7) *Set,* $j := j + 1$.

The BCO is simple, flexible and robust for finding optimal solution. It is easy to implement and has less control parameters when compared to other optimization methods[18].

2.2 Particle Swarm Optimization (PSO)

PSO is a multi-agent parallel search technique. Particles which are conceptual entities fly through multi-dimensional search space. At any instant, a particle has a position and velocity[19]. A particle's position vector regarding the origin of search space represents a search problem's trial solution.

PSO algorithm has three steps, repeated till a stopping condition is met :

- Evaluate the fitness of each particle.
- Update individual and global best fitness and positions.
- Update velocity and position of each particle.

The average time required for each and every task on all the resources is computed. It is generally observed that the time reduces as the cost of communication increases. All the tasks are mapped in the workflow. PSO finds global minima quickly and also attain balanced distribution of workload onto resources.

Fitness evaluation is conducted by supplying a candidate solution to an objective function. Individual and global best fitness and positions are updated by comparing newly evaluated fitness against earlier individual and global best fitness, and replacing best fitness and positions as necessary. Velocity and position update step is responsible for PSO algorithm's optimization ability. PSO algorithm is summarized as follows:

- Initialize the swarm X_i, the position of particles is randomly initialized within the feasible space.
- Evaluate the performance F of each particle, using its current position $X_i(t)$.
- Compare the performance of each individual to its best performance so far: if $F(X_i(t)) < F(P_{ibest})$:
  $F(P_{ibest}) = F(X_i(t))$
  $P_{ibest} = X_i(t)$
- Compare the performance of each particle to the global best particle:
  If $F(X_i(t)) < F(P_{gbest})$:
  $F(P_{gbest}) = F(X_i(t))$
  $P_{gbest} = X_i(t)$
- Change the velocity of the particle.
- Move each particle to a new position.
- Go to step 2, and repeat until convergence.

2.3 Proposed PBCOPSO: A Parallel Algorithm

The goal of parallelization is speeding up computations and to solve a specific problem by engaging many processors and dividing work between them. When meta-heuristics are considered, parallelization strategy and its performance is influenced by the final solution quality. Parallelization assures extension of search space that yields to improvement or degradation of final solution quality[20]. So, final solution quality should be considered as a parameter of parallelization strategy's performance. Consequently, combination of gains is expected: parallel execution enables efficient search of different regions in solution space yielding to improved final solution quality in smaller execution time. Initial solutions were formed and it is iterated through both BCO and PSO to achieve the optimal schedule. The proposed method flow chart is given in Figure 1.

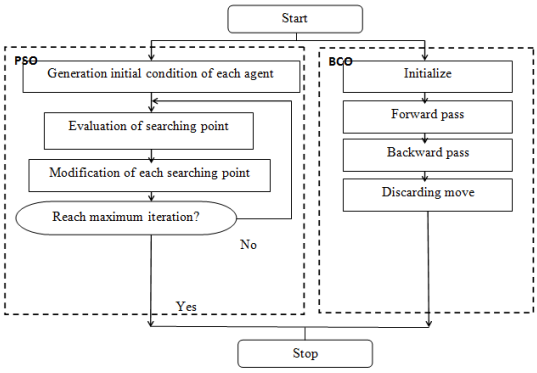

Figure 1. Flowchart for the Proposed Method.

3. Experimental Results

We consider four resources to schedule 40 to 640 tasks. Simulations are carried out using CloudSim tool kit. Makespan and resource utilization are calculated. The results of the proposed method are given below in Table 1. Also the results are compared with existing Min-Min and IBCO also.

Table 1. Makespan (In Seconds)

Number of Tasks	Min-Min	IBCO	PBCOPSO
40	47	42.7	41.3
80	94.4	85.8	82.7
160	190.2	172.5	164.5
320	382.9	346.5	332.2
640	769.6	698	666.5

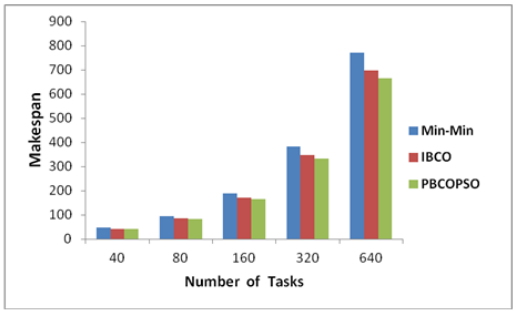

Figure 2. Number of Task vs Makespan (in Seconds).

The above Figure 2 shows the graphical representation of number of task verses the obtained Makespan. From the results it is observed that the proposed method improved the makespan by an average of 14.2099% when compared with Min-Min algorithm and by an average of 4.4289% when compared with Improved Bee Colony Optimization (IBCO).

Table 2. Resource Utilization (in %)

Number of Tasks	Min-Min	IBCO	PBCOPSO
40	79.5	81.9	85.8
80	79.9	81.3	84.7
160	81	82.5	84.8
320	80.4	79.6	82.7
640	79.7	80.5	83.2

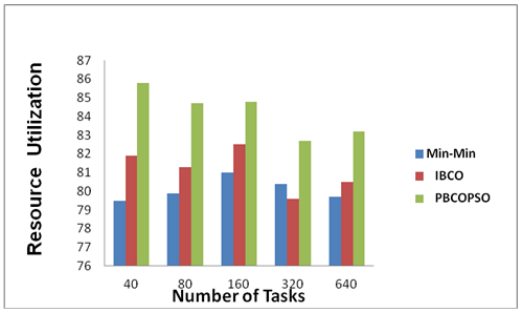

Figure 3. Resource Utilization (in%).

Similarly Table 2 and Figure 3 gives the result of resource utilization achieved for task size ranging from 40 to 640 respectively. Thus from the results it is seen that the proposed PBCOPSO method improved resource utilization by an average of 5.0383% when compared with Min-Min algorithm[21] and by an average of 3.7243% when compared with IBCO[22].

4. Conclusion and Future Work

Increasing applications from the public or enterprise users run in a Cloud, generating diverse sets of workloads regarding resource demands, performance requirements, and task execution, Job scheduling is a major activity in all computing environments. To increase the working of cloud computing environments efficiently, scheduling is performed to gain maximum profit. There are various scheduling algorithm in distributed computing. This study proposed a hybrid parallel algorithm PBCOPSO for task scheduling in cloud environment. Results proved that the proposed method gives an optimized makespan and resource utilization. In future some more parameters could be considered to increase the reliability and to provide better quality of service.

5. References

1. Abdi S, Motamedi SA, Sharifian S. Task scheduling using Modified PSO Algorithm in cloud computing environment. International Conference on Machine Learning, Electrical and Mechanical Engineering; Dubai, UAE. 2014 Jan 8–9.
2. Kaur G, Sharma ES. Optimized utilization of resources using improved Particle Swarm Optimization based task scheduling algorithms in cloud computing. International Journal of Emerging Technology in Advanced Engineering. 2014 June; 4(6):110–5.
3. Delavar AG, Aryan Y. A synthetic heuristic algorithm for independent task scheduling in cloud systems. International Journal of Computer Science Issues. 2011; 8(6):289–95.
4. Preethima RA, Margret J. Survey on optimization techniques for task scheduling in cloud environment. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 Dec; 3(12):413–5.
5. Umarani G, Uma V, Shanthi AP, Arul S. Task scheduling model. Indian Journal of Science and Technology. 2015 Apr; 8(S7):33–42.
6. Bitam S, Batouche M, Talbi EG. A survey on bee colony algorithms. IEEE International Symposium on Parallel and Distributed Processing, Workshops and Phd Forum (IPDPSW); IEEE; 2010 April; 1–8.
7. Liu W, Liu L. Cartes DA Venayagamoorthy GK. Binary Particle Swarm Optimization Based Defensive Islanding Of Large Scale Power Systems. International Journal of Computer Science and Applications. 2007; 4(3):69–83.
8. Karthick AV, Ramaraj E, Subramanian RG. An efficient multi queue job scheduling for cloud computing. IEEE World Congress on Computing and Communication Technologies; 2014 Feb. p. 164–6.
9. Karthick AV, Ramaraj E, Kannan R. An efficient tri queue job scheduling using dynamic quantum time for cloud environment. IEEE International Conference on Green Computing, Communication and Conservation of Energy; 2013 Dec. p. 871–6.
10. Leena VA, Beegom ASA, Rajasree MS. Inter-cloud scheduling technique using power of two choices. IEEE International Conference on Computational Intelligence and Computing Research. 2013. p. 1–4.
11. Zheng Z, Wang R, Zhong H. Zhang X. An approach for cloud resource scheduling based on Parallel Genetic Algorithm. International Conference on Computer Research and Development. 2011 Mar; 2:444–7.
12. Zhong H, Tao K, Zhang X. An approach to optimized resource scheduling algorithm for open-source cloud systems. IEEE China Grid Conference (ChinaGrid); 2010 Jul. p. 124–9.
13. Panda SK, Jana PK. An efficient task scheduling algorithm for heterogeneous multi-cloud environment. IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI); 2014 Sep. p. 1204–9.
14. Madivi R, Kamath SS. An hybrid bio-inspired task scheduling algorithm in cloud environment. IEEE International Conference on Computing, Communication and Networking Technologies; 2014 Jul. p. 1–7.
15. Jemina PR, Arockiam L. SBCOHC: An optimized task scheduling approach for public cloud environment. International Journal of Applied Engineering and Research. 2015 Mar.
16. Fazel MD, Jamali S, Bekravi M. Survey on job scheduling algorithms in cloud computing. International Journal of Emerging Trends and Technology in Computer Science. 2014 Apr; 151–4.
17. Kaur A, Goyal S. A survey on the applications of bee colony optimization techniques. International Journal on Computer Science and Engineering. 2011May; 3(8):3037–46.
18. Davidovic T, Teodorovic D, Selmic M. Bee colony optimization Part I: The algorithm overview. Yugoslav Journal of Operations Research. 2014; 2334–6043.
19. Das S, Abraham A, Konar A. Particle Swarm Optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives. In Advances of Computational Intelligence in Industrial Systems. Berlin, Heidelberg: Springer; 2008; 1–38.
20. Khanesar MA, Teshnehlab M, and Shoorehdeli MA. A novel binary particle swarm optimization. IEEE Mediterranean Conference on Control and Automation; 2007 Jun. p. 1–6.
21. Jemina PR, Arockiam L. Performance evaluation of min-min and max-min algorithms for job scheduling in federated cloud. International Journal of Computer Applications. 2014 Aug; 99(18):47–54.
22. Jemina PR, Arockiam L. Improved Bee Colony Optimization for efficient task scheduling in cloud environment. International Journal of Applied Engineering and Research. 2015 Nov 3; 10(3):6731–43.