Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 15 Issue 3 Version 1.0 Year 2015
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Hybrid Genetic Swarm Scheduling for Cloud Computing

By Dr. M. Sridhar
R.V.R & J.C College of Engineering, India

Abstract- Cloud computing ensures access to shared resources and common infrastructure, offering services on demand over a network for operations to meet changing business needs. Scheduling is a prominent activity that is executed in a cloud computing environment. To increase cloud computing work load efficiency, tasks scheduling is performed to get maximum profit. In cloud, high communication cost prevents task schedulers from being applied in large scale distributed environments. Cloud environment system scheduling is NP-complete. To solve the NP complete and NP hard problems heuristic approaches are used. This study proposes a hybrid optimization based on Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) for scheduling in cloud environments.

Keywords: cloud computing, scheduling, particle swarm optimization (PSO), genetic algorithm (GA).

GJCST-B Classification : C.1.3 D.4.1

HYBRIDGENETICSWARMSCHEDULINGFORCLOUDCOMPUTING

Strictly as per the compliance and regulations of:

Hybrid Genetic Swarm Scheduling for Cloud Computing

Dr. M. Sridhar

Abstract- Cloud computing ensures access to shared resources and common infrastructure, offering services on demand over a network for operations to meet changing business needs. Scheduling is a prominent activity that is executed in a cloud computing environment. To increase cloud computing work load efficiency, tasks scheduling is performed to get maximum profit. In cloud, high communication cost prevents task schedulers from being applied in large scale distributed environments. Cloud environment system scheduling is NP-complete. To solve the NP complete and NP hard problems heuristic approaches are used. This study proposes a hybrid optimization based on Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) for scheduling in cloud environments.

Keywords: cloud computing, scheduling, particle swarm optimization (PSO), genetic algorithm (GA).

I. Introduction

Cloud computing attracts increasing applications to run in remote datacenters. Many complex applications need parallel processing capabilities some of which show a decrease in CPU resources use. Whenever there is a parallelism increase, when jobs are not scheduled correctly, it reduces computer performance. Scheduling allocates tasks to available resources based on tasks' qualities and need [1, 2]. The goal of scheduling is increased resource use without affecting cloud provided services.

Scheduling efficiency mechanism in cloud computing depends on how efficiently it manages the processes and increases server performance as well as resources. Scheduling problems involve jobs that should be scheduled on machines subject to constraints to optimize a given objective function [3]. The goal is computing a schedule that specifies when and on which machine a job is to be executed. The scheduler in online scheduling receives jobs that arrive over time, and schedules them without any knowledge of the future.

Cloud scheduling process is divided into 3 stages namely:

a) Resource discovering and filtering- Datacenter Broker discovers resources in a network system collecting status information on resources.
b) Resource selection - Target resource is selected based on task requirements and resource. The deciding stage.
c) Task allocation -Task is allocated to selected resource.

In Cloud computing, Task Scheduling algorithms aim to minimize tasks make span with minimum resources efficiently. Cloud computing, uses low-power hosts to achieve high usability. Cloud computing is a class of systems and applications that use distributed resources to perform a decentralized function [4].

Clouds computing, uses computing resources (service nodes) in networks to, ensure complicated tasks execution needing large-scale computation. Thus, node selection to execute a task in cloud computing is to be considered. Scheduling algorithms utilize better executing efficiency and maintain system load balancing. The cloud's efficiency depends on algorithms used for task scheduling.

The job scheduling algorithm's advantage is achieving high performance computing and best system throughput [5]. Traditional job scheduling algorithms cannot provide scheduling in cloud environments. According to simple classification, job scheduling algorithms in cloud computing are categorized into 2 groups; Batch mode heuristic scheduling algorithms (BMHA) and online mode heuristic algorithms.

Task scheduling algorithm maps jobs submitted to cloud environment to available resources so that in total response time, make span is minimized. A Max-Min algorithm feature is selecting the largest job and executing it on the fastest resource. The algorithm's drawback is its delaying execution of smaller jobs and indefinitely postponing smaller jobs execution because of the cloud's dynamic nature. The solution to this is improved Max-Min, which works well for a given set of jobs but, the dynamic cloud environment where jobs are submitted any time results in performance degradation.

Min-Min algorithm first finds minimum tasks execution time and then chooses one with least execution time. The algorithm assigns the task to a resource with minimum completion time. The same is repeated by Min-Min, till all tasks are scheduled [6, 7]. The algorithm's limitation is that it chooses smaller tasks first using up resources with high computational power.

Author : R.V.R & J.C College of Engineering, India.
e-mail: mandapati_s@yahoo.com

Conventional scheduling is infeasible in a cloud environment due to its dynamic, distributed, and sharable properties. Tasks resource allocation is to meet performance targets. Many jobs need resources while operating simultaneously [8]. It is important to balance jobs on appropriate resources for optimal performance and cloud's efficient working. So, varied task parameters are considered for scheduling. Available resources should be effectively used without affecting service parameters.

Cloud environment system scheduling is NP-complete. As users increase, tasks to be scheduled also increase proportionately. So, better algorithms to schedule tasks on such systems are required. Scheduling algorithms are service-oriented and vary in environments. To solve NP complete and NP hard problems, heuristic approaches are used. Heuristic techniques used are local heuristics, meta-heuristics and hyper-heuristics. Hyper-heuristics operate at a higher abstraction levels. Meta-heuristic techniques are expensive techniques needing knowledge in problem domain and heuristic technique.

Evolutionary algorithms are based on species origin. Examples are Particle Swarm Optimization (PSO) and Genetic algorithm (GA). PSO is a parallel evolutionary computation technique and a heuristic search method inspired by biological populations swarming behavior [9]. Using PSO ensures a good performance. GA is a search heuristic that mimics natural evolution. It is routinely used to generate useful solutions for optimization and search problems. GAs belongs to a larger class of evolutionary algorithms, generating solutions to optimization problems with techniques from natural evolution like inheritance, selection, mutation, and crossover [10].

PSO algorithm has many advantages like easy realization, high flexibility, strong robustness, and scalability due to which it solves many combinational problems. But, its disadvantages are low convergence rate when solving large scale optimization problems and easily sinking into local optima due its randomicity [11]. PSO is good in an initial phase but when going through iterations convergence rate becomes low and particles lose variety.

There is need for an algorithm to offset these issues and so this study proposed a hybrid algorithm where PSO combines with GA i.e. GAPSO algorithm ensuring better results due to the properties of both. The remaining sections of this paper are organized as follows: Section 2 reviews related work, Section 3 explains the methodology. Section 4 discusses experimental results and Section 5 concludes the work.

II. Related Works

A hybrid task scheduling algorithm based on combining plus points of bio-inspired algorithms like Ant Colony Optimization (ACO) and Artificial Bee Algorithm (ABC) was proposed by Madivi and Kamath [12]. The strong points of both algorithms are incorporated to optimize task scheduling in a cloud algorithm. It is observed that the new algorithm ensured an improvement of about 19% compared to default FCFS scheduling strategy, 11% better than ABC algorithm and was 9% better than conventional ACO based task scheduling.

A Hybrid algorithm combining advantages of ACO and Cuckoo search was proposed by Raju et al., [13]. The makespan is lowered by the hybrid algorithm, as jobs were executed in a specified time interval by required resources allocation using the Hybrid algorithm. Results showed that Hybrid algorithm performed well when compared to ACO algorithm regarding performance of algorithm and make span.

The advantage of Multi-Agent Genetic Algorithm (MAGA) a hybrid of GA, whose performance is superior to traditional GA was proved by Zhu et al., [14]. MAGA solved load balancing in cloud computing by designing a load balancing model based on virtualization resource management. Experiment results comparing MAGA with Minimum strategy proved that MAGA achieved better load balancing performance.

The performance of Hadoop schedulers including FIFO and Fair sharing was analyzed by Rasooli and Down [15] comparing them with a Classification and Optimization based Scheduler for Heterogeneous Hadoop (COSHH) scheduler, developed by the authors. A hybrid solution was introduced, based on insights which selected appropriate scheduling algorithms for scalable and heterogeneous Hadoop systems regarding number of incoming jobs and available resources.

A new parallel hybrid evolutionary algorithm to solve issues of virtual machines subletting in cloud systems was presented by Iturriaga et al., [16]. It deals with allocation of a set of Virtual Machine (VM) requests from customers to available pre-booked resources from a cloud broker, to maximize broker profit. The new parallel algorithm used a distributed subpopulations model, and a Simulated Annealing operator. Evaluation analyzed profit and makespan results of the new methods over a set of problem instances accounting for realistic workloads and scenarios with real data from cloud providers. A comparison with greedy heuristics revealed that the new method computed solutions with up to 133.8% improvement in profit values, while ensuring accurate make span results.

A hybrid batch job scheduling method for grid environment combining GA and PSO techniques to reduce makespan and flow time was proposed by Dehghani Zahedani and Dastghaibyfard [17]. Results showed a reduced make span in 7 of 12 instances of Braun workload compared to Min-Min, Max-Min, and discrete PSO algorithms.

A hybrid scheduling method that computed 5 different schedules, based on a combination of two resource selection rules with 4 job selection rules using the best of the five was proposed by Ashraf and Erlebach [18]. Simulation of workflow scheduling in an advance reservation environment conducted with GridSim revealed that the new hybrid scheduling method achieved makespan improvement of up to 25.5% on benchmark workflows, compared to earlier methods.

A hybrid algorithm, ant colony system and GA to solve job scheduling issues was proposed by Alobaedy and Ku-Mahamud [19]. The high level hybridization algorithm ensured the identity of the algorithm performing scheduling tasks. The new study focused on static grid computing environment and metrics for optimization are makespan and flow time. Results showed that the new algorithm outperformed other stand-alone algorithms like ant system, GA and ant colony system for makespan. But for flow time, ant system and GA performed better.

A new updating mechanism for discrete PSO that directly used discrete solutions from personal and global best particles was proposed by Nguyen and Zhang [20]. A new local search heuristic was proposed to refine solutions found by PSO. Results showed that hybrid PSO is more effective than current PSO methods in literature when tested on 2 benchmark datasets. The efficient hybrid method suited handling large-scale problem instances.

Map Reduce HPSO-GA based on Map Reduce parallel programming model presented by Sadasivam and Selvaraj [21] yielded better results than normal PSO providing better load balancing and resource use in grid environment. It identified the node to which a task is assigned in a Hadoop cluster. So, the new approach could be used in Hadoop resource management system with Hadoop and system parameters to schedule jobs in a Hadoop cluster.

A hybrid job scheduling approach, which considered system load balancing and reduced total execution time and execution cost was presented by Javanmardi et al., [22]. The proposed work's goal was assigning jobs to resources considering VM MIPS and jobs length. The new algorithm assigned jobs to resources considering job length and resource capacity. Performance was evaluated with famed cloud scheduling models. Results showed the proposed approach's efficiency regarding execution time, execution cost and average degree of imbalance.

III. Methodology

In this work, a hybrid algorithm for cloud scheduling is proposed. It is based on PSO and GA. In the hybrid algorithm PSO combines with GA i.e. GAPSO algorithm ensuring better results.

a) Genetic Algorithm (GA)

GA is a meta-heuristic technique solving optimization problems by imitating natural selection; i.e., the adaptation to an environment performed by living beings [23]. GA is an appealing approach to solve a complex problem. GA determines not one solution but a whole 'population' of 'individuals,' which are candidate solutions to a problem. Each individual's distinctive features are coded into a 'chromosome' which is a string of genes, whose values are chosen from a set of symbols.

GAs are stochastic search methods managing a population of simultaneous search positions. A conventional GA has 3 essential elements:

- a coding of the optimization problem
- a mutation operator
- a set of information-exchange operators

GAs evaluate target function to be optimized at randomly selected points of a definition domain. Considering this information, a new set of points (a new population) is generated. Gradually the population approaches a function's local maxima and minima. The GA's pseudo code is given below [24].

1. Choose initial population(Random)
2. Repeat (until terminated)

2.1 Evaluate each individual Fitness

2.2 Prune population (Typically all; If not then the worst)

2.3 Select pairs to mate from best-ranked individuals

2.4 Replenish population (Selected pairs)

2.4.1 Apply Crossover operator

2.4.2 Apply mutation operator

3. Check for termination criteria

3.1 Loop if not terminating(Repeat from step 2)

b) Particle Swarm Optimization (PSO)

PSO, like other evolutionary computation techniques is a population based search algorithm initialized with a population of random solutions, called particles. Unlike other evolutionary computation techniques, a PSO particle is associated with velocity. Particles fly through search space with velocities, dynamically adjusted according to their and swarm's historical behaviors. So, particles fly towards better and better solutions in a search process. PSO algorithm is simple in concept, easy to implement, and efficient computationally. The PSO algorithm's updating rules are listed as [25].

$$V_i = W.V_i + C_1.rand_1(X_p - X_i) + C_2.rand_2(Xg - X_i) \quad X_i = X_i + V_i$$

The PSO algorithm consists of just three steps, which are repeated until some stopping condition is met [26]:

1. Evaluate the fitness of each particle
2. Update individual and global best fitness and positions
3. Update velocity and position of each particle

The first two steps are fairly trivial. Fitness evaluation is 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 needed. The PSO algorithm is summarized as follows [27]:

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

c) Hybrid Genetic Algorithm Particle Swarm Optimization (GAPSO)

GAs were applied successfully to varied problems. But, using them for large-scale optimization is expensive due to its requirement of many function evaluations for convergence, leading to high cost for function evaluations computation [28]. Considering the PSO's efficiency and GA and PSO's compensatory property combining searching abilities of both in one algorithm seems logical. Both GA and PSO share common elements [29]:

1. Both initialize a population similarly.
2. Both use an evaluation function to determine a potential solution's fitness.
3. Both are generational, repeating same processes for a predetermined time.

A hybrid algorithm has operator like enhancement, selection, crossover, and mutation.

Enhancement: In every generation, after fitness values of population individuals are calculated, the top-half best performers are marked and regarded as elite. Instead of reproducing the elite directly to next generation as elite GAs do, they are first enhanced.

Here, a dynamic decrease of ω value was suggested based on a fraction multiplier (kw). When no improvement was made for a predefined number of consecutive design iterations:

$$\omega^{t+1} = k_w \omega^t$$

It should be noted that elite in a generation can be from both groups of an earlier generation, i.e., enhanced elite or produced offspring. If elite i is an offspring produced by parents of a previous generation, then v_i^t is set to zero, and p_i^t is set to x_i^t , i.e., the newly generated individual itself. Otherwise, p_i^t records best solution of individual i evolved till then.

Selection: In GAPSO, GA operations are performed on the enhanced elite achieved by PSO. To select parents for crossover, a tournament selection scheme is used. Two enhanced elite are selected randomly, and their fitness values compared to select one with better fitness as a parent for placing it in a mating pool. This scheme is the selection operator in GA also.

Crossover: Parents are randomly selected from the mating pool in groups of two and two offspring are created through a crossover on parent solutions. This study uses a simulated binary crossover (SBX) [28]. SBX operator is suitable as the spread of children solutions around parent solutions is controlled using distribution index, ηc. With this operator, an arbitrary contiguous region can be searched, provided there is enough diversity among feasible parent solutions

Mutation: A final genetic operator, it creates new genetic material in the population to maintain the latter's diversity. Mutation operator used is a variable dependent random mutation where a solution is created near the parent solution with uniform probability distribution

$$x_i^{(1,t+1)} = x_i^{(1,t)} + (r_i - 0.5)\Delta_i$$

r_i is a random number in [0, 1]. Δ_i is user defined maximum perturbation allowed in ith decision variable (x_i). It should be checked if the above calculation takes $x_i^{(1,t+1)}$ outside a specified lower and upper limits.

IV. Experimental Results

To verify the proposed algorithm Cloudsim, a simulation software provided by Gridbus project was used. The proposed algorithm is integrated in the cloudsim layer. The simulator models the datacenter components which are designed to handle the service requests. These requests are the tasks which need to be allocated to VMs for processing. The VM is allocated a share of processing power by the Datacenter. Each VM has a pre configured processing capability based on memory, storage and Millions of Instructions it can execute per second. In this work four VMs from two data centers are used. The available Band Width of 128Kbps is dynamically varied. Similarly the memory is also dynamically varied from 256 Mb to 1.5 Gb. In this work, number of tasks were varied from 200 to 1000 in increments of 200 for estimating average schedule length and ratio of successful execution.The results are shown in table 1.

Table 1 : Average Schedule Length

Number of tasks	Max-Min scheduling	Minimum Execution time	Hybrid GAPSO
200	679	664	641
400	1364	1337	1290
600	2072	2048	1956
800	2746	2690	2592
1000	3438	3373	3253

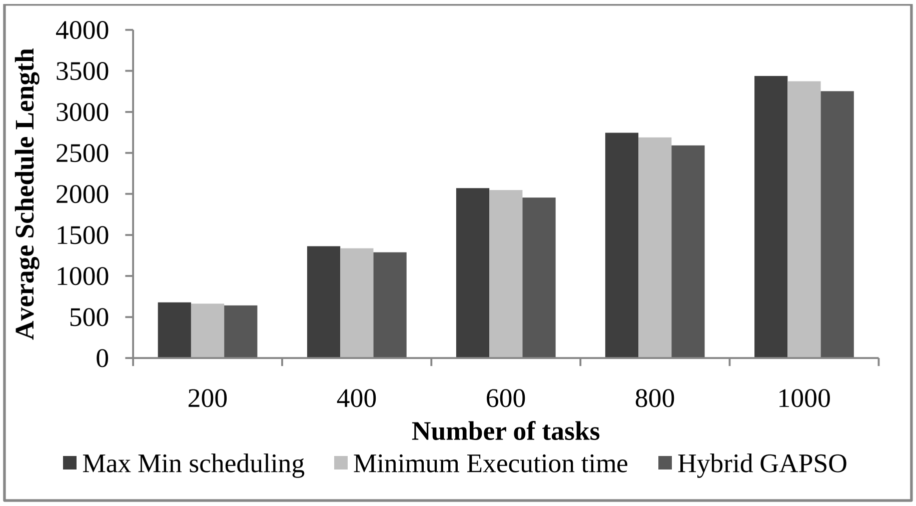

Figure 1 : Average Schedule Length

From table 1 and figure 1 it is observed that the proposed GAPSO performs better by reducing the schedule length when compared to Max-Min Scheduling and Minimum Execution time. The GAPSO reduced average schedule length by 5.76% than Max-Min Scheduling and by 4.6% than Minimum Execution Time with 600 numbers of tasks.

Table 2 : Ratio of successful execution

Number of tasks	Max Min scheduling	Minimum Execution time	Hybrid GAPSO
200	0.91	0.94	0.96
400	0.89	0.91	0.95
600	0.87	0.89	0.91
800	0.87	0.87	0.89
1000	0.85	0.86	0.88

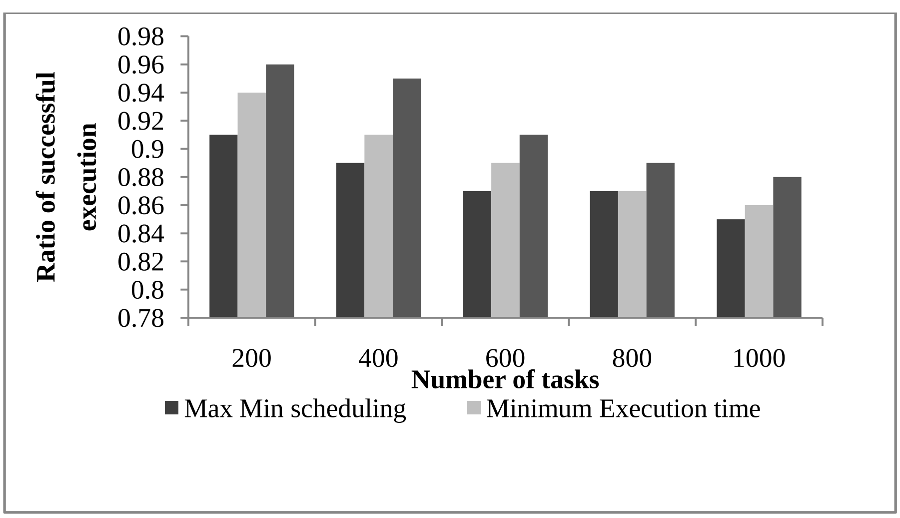

Figure 2 : Ratio of successful execution

From table 2 and figure 2 it is observed that the proposed GAPSO performs better by increasing the execution ratio when compared to Max-Min Scheduling and Minimum Execution time. The GAPSO improved Ratio of successful execution by 6.52% than Max-Min Scheduling and by 4.3% than Minimum Execution Time with 400 numbers of tasks.

V. Conclusion

Cloud computing is a provider of dynamic services using huge scalable and virtualized resources over the Internet. Due to its novelty, there is no standard task scheduling algorithm in a cloud environment. In this study, the GAPSO performs better when compared to Max-Min Scheduling and Minimum Execution time in case of schedule length and execution ratio. GAPSO reduced average schedule length by an average of 5.66% than Max-Min Scheduling and by an average of 3.83% than Minimum Execution Time. GAPSO improves execution ratio by an average of 4.45% than Max-Min Scheduling and by an average of 2.65% than Minimum Execution Time.

References Références Referencias

1. Tripathy, L., & Patra, R. R. Scheduling in cloud computing.
2. Kaur, R., & Kinger, S. Analysis of Job Scheduling Algorithms in Cloud Computing.
3. Azar, Y., Ben-Aroya, N., Devanur, N. R., & Jain, N. (2013, July). Cloud scheduling with setup cost. In *Proceedings of the 25th ACM symposium on Parallelism in algorithms and architectures* (pp. 298-304). ACM.
4. Vijayalakshmi A. Lepakshi, & Dr. Prashanth C S R (2013). A Study on Task Scheduling Algorithms in Cloud Computing. International Journal of Engineering and Innovative Technology (IJEIT), 2 (11).
5. Salot, P. (2013). A Survey Of Various Scheduling Algorithm In Cloud Computing Environment. *International Journal of Research in Engineering& Technology (IJRET)*, *2*(2), 131-135.
6. Santhosh B & Dr. Manjaiah D.H (2014). An Improved Task Scheduling Algorithm based on Max-min for Cloud Computing.
7. Patel, G., & Mehta, A. P. R. A survey on various task scheduling algorithm in cloud computing.
8. Kumar, V. S. (2014). Hybrid optimized list scheduling and trust based resource selection in cloud computing. *Journal of Theoretical & Applied Information Technology*, *69*(3).
9. Jacob, L., Jeyakrishanan, V., & Sengottuvelan, P. (2014). Resource Scheduling in Cloud using Bacterial Foraging Optimization Algorithm. *International Journal of Computer Applications*, *92*(1), 14-20.
10. Kaur, S., & Verma, A. (2012). An efficient approach to genetic algorithm for task scheduling in cloud computing environment. *International Journal of Information Technology and Computer Science (IJITCS)*, *4*(10), 74.
11. Kaur, G., & Sharma, E. S. Optimized Utilization of Resources Using Improved Particle Swarm Optimization Based Task Scheduling Algorithms in Cloud Computing.
12. Madivi, R., & Kamath, S. S. (2014, July). An hybrid bio-inspired task scheduling algorithm in cloud environment. In *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on* (pp. 1-7). IEEE.
13. Raju, R., Babukarthik, R. G., Chandramohan, D., Dhavachelvan, P., & Vengattaraman, T. (2013, February). Minimizing the makespan using Hybrid algorithm for cloud computing. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International* (pp. 957-962). IEEE.
14. Zhu, K., Song, H., Liu, L., Gao, J., & Cheng, G. (2011, December). Hybrid genetic algorithm for cloud computing applications. In *Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific* (pp. 182-187). IEEE.
15. Rasooli, A., & Down, D. G. (2012, November). A hybrid scheduling approach for scalable heterogeneous hadoop systems. In *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:* (pp. 1284-1291). IEEE.
16. Iturriaga, S., Nesmachnow, S., Dorronsoro, B., Talbi, E. G., &Bouvry, P. (2013, October). A parallel hybrid evolutionary algorithm for the optimization of broker virtual machines subletting in cloud systems. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on* (pp. 594-599). IEEE.
17. Dehghani Zahedani, S., & Dastghaibyfard, G. (2014, October). A hybrid batch job scheduling algorithm for grid environment. In *Computer and Knowledge Engineering (ICCKE), 2014 4th International e Conference on* (pp. 763-768). IEEE.
18. Ashraf, J., &Erlebach, T. (2011, July). A hybrid scheduling technique for grid workflows in advance reservation environments. In *High Performance Computing and Simulation (HPCS), 2011 International Conference on* (pp. 98-106). IEEE.
19. Alobaedy, M. M. & Ku-Mahamud, K. R., (2014).Scheduling jobs in computational grid using hybrid ACS and GA approach. Computing, Communications and IT Applications Conference (Com Com Ap), pp.223-228
20. Nguyen, S. B., & Zhang, M. (2014, July). A hybrid discrete particle swarm optimisation method for grid computation scheduling. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (pp. 483-490). IEEE.
21. Sadasivam, G. S., &Selvaraj, D. (2010, December). A novel parallel hybrid PSO-GA using Map Reduce to schedule jobs in Hadoop data grids. In *Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on*(pp. 377-382). IEEE.
22. Javanmardi, S., Shojafar, M., Amendola, D., Cordeschi, N., Liu, H., & Abraham, A. (2014). Hybrid Genetic Algorithm for Cloud Computing Applications. *arXiv preprint arXiv:1404.5528*.
23. Girgis, M. R., Mahmoud, T. M., Abdullatif, B. A., & Rabie, A. M. Solving the Wireless Mesh Network Design Problem using Genetic Algorithm and Tabu Search Optimization Methods.
24. Aravindh, S. (2012). Hybrid of Ant Colony Optimization and Genetic Algorithm for Shortest Path in Wireless Mesh Networks. *Journal of Global Research in Computer Science*, *3*(1), 31-34.
25. Liu, W., Liu, L., Cartes, D. A., &Venayagamoorthy, G. K. (2007). Binary Particle Swarm Optimization Based

Defensive Islanding Of Large Scale Power Systems. *IJCSA*, *4*(3), 69-83.
26. Blondin, J. (2009). Particle swarm optimization: A tutorial. *from site: http://cs. armstrong. edu/saad/csci8100/pso tutorial. pdf*.
27. Khanesar, M. A., Teshnehlab, M., & Shoorehdeli, M. A. (2007, June). A novel binary particle swarm optimization. In *Control & Automation, 2007. MED'07. Mediterranean Conference on* (pp. 1-6). IEEE.
28. Kaveh, A., &Malakouti Rad, S. (2010). Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design. *Iranian Journal of Science and Technology, Transaction B: Engineering*, *34*(B1), 15-34.
29. Settles, M. (2005). An Introduction to Particle Swarm Optimization. *Department of Computer Science, University of Idaho*.

This page is intentionally left blank