THINK INDIA JOURNAL
ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

Comparative Analysis Between Different Meta Heuristic Algorithms For Parallel Job Scheduling

Amandeep Kaur [1], Archana Chhabra[2]

School of Polytechnic Lovely Professional University. Phagwara, Punjab, India

Abstract:

As number of computing resources are increasing day-by-day in different firms, consequently, nominal job scheduling algorithms are necessary for the proficient use of resources. Job scheduling is considered as NP hard problem in parallel computing environment likeclouds ,cluster and grid . Metaheuristics such as GA, Ant Colony Optimization, Artificial Bee Colony, Cuckoo Search, Firefly Algorithm, Bat Algorithm etc. are used by many people to findoptimumresults for problem of job scheduling. These metaheuristic algorithms are utilized to schedule diversesort of jobs such Bag-of-Tasks and independent tasks.This paper provides details of popular metaheuristic techniques based on nature that are used to schedule separate kinds of jobs in order to attain better performance.

Keywords: Job scheduling, independent tasks, workflow, metaheuristic algorithm and bag-of-tasks.

I. INTRODUCTION

Different analysts have indicated a distinct fascination for planning techniques because of exponential increment in number of assets in various associations. Booking permits ideal asset distribution among different errands in a limited time bringing about quality assistance [1]. Booking and asset designation are considered as NP difficult issues. Populace based metaheuristics are seen as compelling for finding ideal or close to ideal arrangements. In this paper, different sorts'metaheuristic systems are examined and different target works that analysts have taken a shot at, utilizing these methods in various kinds of situations are talked about.

A. Scheduling

The technique by which indicated work is allocated to the asset by certain methods, to finish a specific undertaking is called booking. Schedulers keep the assets occupied by enabling different clients to share the frameworks assets in a successful way. It is because of booking that PCs can perform various tasks inside a solitary CPU. Disseminated registering has caught a lot of eye in the ongoing years as it is dependable, profoundly adaptable and has ease [1]. The intricacy of planning increments because of heterogeneity of the preparing and correspondence assets, which prompts NP-Difficult issues. For such issues, there aren't any calculations that may deliver ideal arrangements. This carries us to metaheuristic calculations that idea close to ideal arrangements inside sensible time. There are different sorts of metaheuristic calculations in particular Subterranean insect Settlement Enhancement (ACO), Firefly Calculation, Molecule Swarm Advancement (PSO), Hereditary Calculation (GA), Cuckoo Search, Reenacted Toughening and so on.

B. Metaheuristics

A metaheuristic is a method that finds or creates a calculation that may produce ideal arrangements. Metaheuristics are of different sorts dependent on search procedure, single arrangement based, nature motivated and so forth. Further are talked about different metaheuristic calculations:

1) **Genetic Algorithm:**It is a streamlining strategy dependent on populace and depends on Darwin's hypothesis of advancement [2]. In GA, every conceivable arrangement is spoken to by a chromosome. An underlying populace is utilized haphazardly and it is utilized at initial stage. A wellness work is

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

determined for each chromosome with the goal is known whether the chromosome is reasonable or not. Hybrid and change capacities are performed on the chose chromosmes and offsprings for new populace are made. This procedure is rehashed until enough offsprings are made [3]. The essential strides for hereditary calculation are :

- Initialization
- Selection
- Crossover
- Mutation

1. Pick an initial arbitrary population of individuals
2. Evaluate the fitness function of the individuals
3. **repeat**
4. The best individuals should be selected
5. Generate new individuals by applying crossover and mutation operators
6. The fitness function for new individuals should be examined
7. The worst individuals are replaced with the best ones
8. **until** a stopping criteria is met

Fig 1:Pseudocode for genetic Algorithm

2) **Ant Colony Optimization Algorithm:** In ACO, various fake ants help in building answer for improvement issues and by means of a correspondence plot, they trade the data. They locate the most limited ways as the moving ants lay pheromone on the ground, with the goal that when another insect experiences it, it can recognize it and choose to pursue the trail. Subsequently, the developed aggregate conduct means that in the event that various ants pick a specific way, at that point the likelihood of different ants following similar way increments [4]. The primary thought of ACO is to demonstrate an issue as the hunt of least cost way in a diagram [5].

1. Begin
2. Initialise pheromone trails;
3. Produce an underlying population of a solutions(ants);
4. For every ant f ϵ a: calculate fitness function (f);
5. Determine the best position for each ant;
6. Determine the best global ant;
7. Update the pheromone ;
8. Determine if termination= true;

Fig 2:Pseudocode for ACO

3) **Artificial Bee Colony Algorithm**:This is a prevalent methodology for streamlining that reenacts the shrewd scrounging nature of bumble bees. In this calculation, there are three types of honey bees. The initial ones being the utilized honey bees. They look for nourishment around the nourishment source and furthermore they share this data about the nourishment source with the spectator honey bees. They sift through the great nourishment sources among those found by the utilized honey bees. The top notch (wellness) nourishment source is bound to be chosen. The utilized honey bees which desert the nourishment source andthen find new ones are called scout honey bees [6].

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

```
1. Initialise a random population
2. Evaluate its fitness function
3. While (stopping criterion is not met)
4.     Pick sites for neighbourhood search
5.     The bees for picked sites should be examined and
       fitness function is calculated
6.     From each patch, the fittest bee should be selected.
7.     Remaining bees are assigned to search randomly and
       evaluate their fitness
8. End while
```

Fig3:Pseudocode for ABC

4) **Firefly Algorithm:**It is propelled from the blazing conduct of tropical fireflies. It has the accompanying significant highlights:

• The fireflies are pulled in to one another regardless of their sexual orientation.

• A firefly's engaging quality increments with its glimmer's splendour and appeal and brilliance decline with increment in separation.

• The brightening or brilliance of a firefly is harrowed by the scene of target function[7].

```
1.  Define objective function f(a), where a = (a1,..... ad)
2.  Produce an underlying population of fireflies
3.  Formulate the light intensity L
4.  Specify the absorption coefficient β
5.  While (t< Max_Gen)
6.  For i=1 to n (all n fireflies)
7.     For j=1 to n (all n fireflies)
8.           If ( Lj > Li ), move firefly i towards firefly j
9.        End if
10.          Examine new solutions and update light
          intensity;
11.    End for j
12.  End for i
13.  Rank the fireflies and find the current best
14.  End while
```

Fig4: Pseudocode for Firefly Algorithm.

5) **Cuckoo Search Algorithm:**It is propelled by brood parasitism of some cuckooo species. They lay eggs in the homes of their feathered creatures of various species. The following are the three essential standards for this calculation:

- At a given time one egg is laid and then it is dumped into another nest.
- Just the home having the greater eggs is moved to the people to come.
- The quantity of host nest is stabled and likelihood of finding the egg laid by the cuckoo by holder flying creature is dad. The host can dispose of the egg or it can surrender the home and construct a new one [7].

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

```
1.   begin
2.    Define objective function f(o)
3.    Formulate the initial population of h host nest
4.    Eggs should be ranked after assessing the fitness
5.    while (z>MaxGeneration) or stopping criteria is met
6.     z= z+1
7.    Get a cuckoo arbitrarily or produce new solution by
      Levy flights
8.     Assess fitness, F_i
9.     Choose a nest j, randomly
10.    if ( F_i > F_j )
11.       Replace j with the new solution
12.    end if
13.     Abandon the worst net with probabilty P_a and a new
        nest is then built
14.     Assess fitness, rank the solutions and find the
        current best
15.    end while
16:  end
```

Fig 5. Pseudo code for cuckoo search

6) Particle Swarm Optimization Algorithm: The possibility of PSO rose up out of the swarming conduct of group of flying creatures, swarm of honey bees, schools of fish and so forth. It is applied to take care of various capacity advancement issues. In PSO, the arrangements are named particles that movement in the issue space. They pursue the present ideal particles. The co-ordinates of every molecule in the issue space are followed by the molecule. They are related with the best arrangement accomplished up to now and the worth is known as pbest. Another best worth, lbest, is the best worth accomplished by any molecule in the area of the molecule. Worldwide best worth, gbest, is gotten when a molecule accepts all the populace as its neighbours [8].

```
1.    For each particle
2.       Initialise particle
3.    End for
4.    Do
5.       For each particle
6.          Evaluate the value of fitness
7.                if the fitness value is better than the
              best value
8.                set current value as pbest
9.          End
10.           select the particle with best fitness value as
            gbest
11.         For each particle
12.          evaluate particle velocity
13.         update particle position
14.      End
15.   while maximum iterations or minimum error
      condition is not achieved.
```

Fig 6: Pseudo code for PSO

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

7) Simulated Annealing:

This method is propelled from tempering in metallurgy. It includes warming and afterward moderate cooling of a material. This is utilized to expel/diminish the imperfections in a precious stone. To achieve this, the material is toughened for example it is warmed to a particular temperature and afterward cooled gradually until the material stops into the type of a decent precious stone [9]. The moderate cooling infers to slow diminish in probability of tolerating more terrible arrangements. On the off chance that the cooling is moderate, it implies that there is a higher possibility of finding ideal or close ideal arrangements [10].

```
1.  S0 = GenerateSolution()
2.  Temp = START_TEMP
3.  K = 0 // iteration count
4.  while
5.     S1 = Neighbour (S0)
6.     if (Fitness (S1) < Fitness (S0))
7.           S0 = S1
8.        else if (rand() < tempFunc (S0, S1, Temp,  K))
9.           S0 = S1
10.    end if
11.    AnnealingSchedule (S0, Temp, K)
12.    K = K+1
13. end while
```

Fig 7.:Pseudo code for Simulated Annealing

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

The following figure comprises of the examination of various nature-propelled metaheuristic calculations. They are looked a

Sr. No.	*Technique*	*Pros*	*Cons*
1.	Genetic Algorithm	Finds near optimal solution, Avoids being trapped in local optimal solutions.	No guarantee of finding global maxima, Convergence time is more.
2.	Ant Colony Opt. Algorithm	Inherent parallelism, Can be used in dynamic application.	Time to convergence is uncertain.
3.	Artificial Bee Colony Algorithm	High Performance, Fewer control parameters, Easily modified and hybridized with other metaheuristic algorithms.	Risk of losing relevant information. Population of solution increases the computational cost.
4.	Firefly Algorithm	All advantages of swarm based algorithms, Precise and robust.	Trapped in local minima, Slow convergence.
5.	Cuckoo Search	Global convergence, Global optimality.	Doesn't incorporate any type of local search to increase the convergence speed, Performance highly dependent on α i.e. step size.
6.	Particle Swarm Opt. Algorithm	Fast search speed, Calculation is simple.	Tendency of premature convergence, Suffers from partial optimism.
7.	Simulated Annealing	Can deal with arbitrary systems and cost functions, Gives a "good" solution.	Few local minima, Slow.

based on their pros and cons:

Fig 8:Comparison between different algorithm.

C. **Multi-criteria**

Multi-criteria or multi-target streamlining is a sort of basic leadership criteria. In this, at least two than two target capacities are improved all the while. It comes being used when ideal choices are required to be taken in nearness of exchange offs between at least two conflicting capacities. In numerical terms, it very well may be composed as:

$$min(g1(x), g2(x), \dots \dots ga(x))$$
$$s.t. x \epsilon X$$

where the whole number a ≥ 2 and an is the quantity of targets and X is the arrangement of plausible choice vectors. This estimation of a shows that base 2 destinations are required.

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

In this work, different kinds of occupations, parallel or non-parallel are examined and dependent on those, tables have been developed. Principle center is around the goals, kind of occupation and condition. Principally four kinds of occupations have been considered, to be specific: Bulk Synchronous Parallel (BSP), DAG and Workflow, Independent Jobs and Bag-of-Tasks.

II. PERFORMANCE METRICS:

Prior to going further, underneath are some presentation measurements that are wanted by various sorts of clients and suppliers:

A. Makespan: Makespan alludes to the all out length of the calendar for example the completing time of the last assignment. It is the most mainstream advancement rule and shows the profitability of a framework. Lesser the estimation of makespan, increasingly proficient is the scheduler [11].

Makespan=*Maxkϵtasks (Tk),*

B. Flowtime: The total of finish times of all undertakings is called flowtime.

Flowtime=$A\Sigma k\epsilon tasks\ (Tk)$,

C. Convergence Speed: The speed at which an iterative arrangement joins to a point where it arrives at an ideal or close to ideal arrangement is known as the speed of union.

D. Response time: Summation of waiting time and execution time is known as response time.

E. Throughput: The quantity of occupations that total their execution per unit time is named as throughput. At the end of the day, it is the measure of work done by a framework in given time.

F. Scalability: For the most part, speedup decays when number of processors increment and size of issue is fixed. Scalabilty alludes to the adjustment in execution of a parallel framework as the size of issue and PC size increment [11].

III. BULK SYNCHRONOUS PARALLEL MODEL:

A (BSP) PC includes an assortment of processors. Every processor has its own memory for example it is an appropriated memory PC. It has predominantly three parts: processor/memory pair,communication system and synchronization obstruction [12]. This model is utilized for parallel employments. The occupations in BSP contain a fixed number of undertakings which have practically indistinguishable necessities and each assignment establishes different cycles.

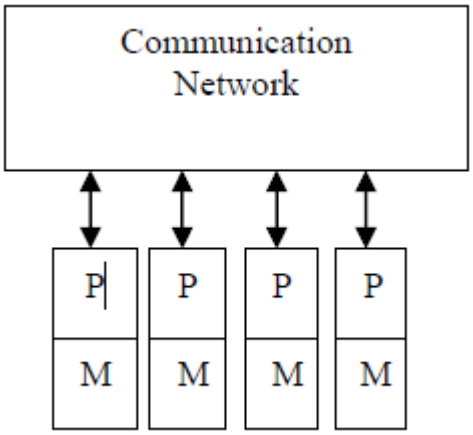

Fig 9:BSP Model

Weipeng and Danyu [12] exhibited another programming model named MaMR for cloud applications. It bolsters different guide and diminish capacities that run in parallel. It utilizes a mixture shared-memory BSP model. The recreation results depicts the exhibited model shows powerful improvement in execution contrasted with the past work. Rodrigo et al. [13] proposed a relocation model, MigPF, for BSP programs in heterogeneous conditions. A model was created with the AMPI library and was tried against other inherent AMPI rescheduling approaches. Exploratory outcomes indicated 41% execution gain and the overhead constrained to 5%.

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

Gabaldon et al. [14] introduced a multi-objective Genetic Algorithm that depends on a weighted boycott. The creators have dealt with decreasing makespan and furthermore towards vitality protection. An augmentation to the BSP model called MBSP, for multi-memory BSP, within the sight of various heirarchies and centers was proposed in [15]. Gabaldon [16] progressed in the direction of the planning in multi-bunch conditions utilizing a metaheuristic strategy for example Hereditary Algorithm. The multi-target work advances makespan just as the flowtime. Marcos et al. [17] proposed a basic BSP-based model for foreseeing execution times for CUDA applications on GPUs . The model expectations were 0.8 to 1.2 occasions the deliberate execution times which shows that the model is sufficient for summing up forecasts for various issue sizes and GPU arrangements. The populace assorted variety in Genetic Algorithm is improved in [18]. The primary spotlight is on movement. Three procedures for parallelisation were utilized which were : MapReduce, BSP and MPI. The proposed strategy was contrasted and existing strategies and results demonstrated improvement in arrangement precision.
Three methodologies for diagram handling to be specific MapReduce, BSP, and MR2 were thought about by Tomasz [19]. The examinations were led on four enormous chart datasets. Xiaodong and Weikin [20] proposed a reconciliation of disseminated memory and shared-memory BSP model named BSPCloud. The BSPCloud can make full usage of multi-center bunches. The presentation consistency and speedup were assessed. J. Steven and Ovidiu [21] displayed a parallel GPU model which helps the parallel GPU calculation architects to structure and execute ideal calculations. Results show that calculations planned dependent on the model's standards give critical increment in execution.

IV. WORKFLOW AND DAG BASED JOBS

Work process planning can be called as a general type of assignment booking. Here, the applications are displayed as (DAG). DAG is a famous portrayal of work process applications. Work process booking can extensively be characterized into two classifications:

A. Static Workflow Scheduling: Static planning can look from the arrangement space all around at work process level. Be that as it may, its restriction is that it expect that precise errand correspondence and execution time could be gotten before planning which can not really consistently be genuine [22].

B. Dynamic Workflow Scheduling:It is useful when booking data is inaccessible or when there is asset dispute with other framework's heap [23] library and was tried against other implicit AMPI rescheduling approaches. Test results demonstrated 41% execution gain and the overhead restricted to 5%. Gabaldon et al. [14] introduced a multi-objective GA that depends on a weighted boycott. The creators have dealt with diminishing makespan and furthermore towards vitality protection. An expansion to the BSP model called MBSP, for multi-memory BSP, within the sight of various heirarchies and centers was proposed in [15]. Gabaldon [16] progressed in the direction of the planning in multi-group situations utilizing a metaheuristic strategy for example Hereditary Algorithm.The multi-target work upgrades makespan just as the flowtime.

Marcos et al. [17] proposed a basic BSP-based model for foreseeing execution times for CUDA applications on GPUs . The model forecasts were 0.8 to 1.2 occasions the deliberate execution times which shows that the model is adequate for summing up expectations for various issue sizes and GPU arrangements. The populace decent variety in Genetic Algorithm is improved in [18]. The principle center is around relocation.Three strategies for parallelisation were utilized which were : MapReduce, BSP and MPI. The proposed technique was contrasted and existing strategies and results indicated improvement in arrangement precision. Three methodologies for diagram handling to be specific MapReduce, BSP and MR2 were analyzed by Tomasz [19]. The trials were led on four huge diagram datasets. Xiaodong and Weikin [20] proposed a mix of appropriated memory and shared-memory BSP model named BSPCloud. The BSPCloud can make full usage of multi-center bunches. The exhibition consistency and speedup were assessed. J. Steven and Ovidiu [21] displayed a parallel GPU model

ISSN: 0971-1260
Vol-22-NO-30 Special Issue- December-2019

which helps the parallel GPU calculation planners to structure and actualize ideal calculations. Results show that calculations planned dependent on the model's standards give huge increment in execution.

V. INDEPENDENT JOBS

Free occupations are those in which the errands don't legitimately speak with one another. All assignments perform same or comparable capacity and it isn't fundamental for them to run all the while.

Medhat et al. [34] analyzed a cloud task booking calculation dependent on Ant Colony Optimization (ACO) with FCFS and RR planning calculation. The principle thought process is to limit the makespan. The recreation in CloudSim demonstrated that the ACO based cloud task planning performs superior to FCFS and RR calculations. Rajni et al. [35] proposed a PSO-based hyper-heuristic asset planningcalculation for work planning for Grid situations. The PSOHH gave preferable outcomes over existing hyper-heuristic booking calculations to the extent cost, time and makespan are concerned. A mixture PSO and BAT calculation is introduced in [36]. The half and half consolidates the highlights of both PSO and BAT calculation. Results give quicker union speed, less parameters to tune and simple looking in huge spaces. An improved rendition of Ant Colony Optimization calculation is introduced in [37]. This work thinks about SLA infringement, vitality utilization and burden adjusting. Ch. Srinivasa [38] proposed a fluffy differential development for planning on computational frameworks. The exhibition of fluffy DE was contrasted and other existing transformative calculations. Trial results show that the proposed calculation created progressively ideal arrangements contrasted with different calculations. Rajni [39] proposed a novel bacterial searching based hyper heuristic planning calculation for booking assets in network condition. The reproduction results give better makespan and limited expense. Susmita et al. [40] present an asset intermediary engineering for GA in computational frameworks. There are various clients that present the occupations and different suppliers that are chosen by the intermediary. Subsequently, the all out finish time (TCT) is limited. Jinn-Tsong et al. [41] proposed an improved differential advancement calculation (IDEA) for task planning and asset designation in distributed computing condition. The proposed calculation incorporates the Taguchi strategy and Differential Evolution Algorithm. Results give littler makespan and cost. Dasgupta et al. [42] focussed on load adjusting utilizing hereditary calculation while booking assignments in cloud.

VI. BAG-OF-TASKS

Bag- of-Tasks establish of numerous free assignments that can be excuted in parallel. BoT discover their utilization in many field like picture handling information mining and so on. In these fields, applications comprise of a few stages rather than a solitary advance. These means could be successive, parallel or half and half of both. Each progression forms a BoT [45]. The errands of BoT applications are free of one another and may have diverse computational necessities. They likewise don't have to speak with one another during execution [46]. BoT execution for the most part requires exorbitant interests in foundation, the explanation being high utilization of assets and parallel nature of BoTs [49].

Zhicheng et al. [45] center around satisfying the work process cutoff time by utilizing a unique cloud asset provisioning and planning calculation. The VMs are leased powerfully and primary rationale is to limit the asset leasing cost. George and Helen [46] proposed control mindful planning arrangements by broadening the Min-Min and Max-Min strategies for homogeneous bunches. They proposed two sorts of approaches in particular voracious and preservationist. The ravenous approaches accomplished upto 20.6% vitality utilization which was more than preservationist strategies. The execution of BoTs and high-need assignments wasn't incredibly influenced. Ioannis [47] moved in the direction of booking of sack of-assignments applications in heterogeneous cloud with the utilization of metaheuristic procedures. Two calculations, Simulated Annealing and Tabu hunt have been applied . Recreation results for both the calculations show that there were benefits in both expense and execution. Javier [48] proposed a decentralized model for booking approach whose sole goal is reasonableness.

Recreation results demonstrate this approach to be adaptable and powerful. The proposed strategy performed just 11% more awful than brought together usage.

Another cloud BoT execution instrument, CloudAgent is proposed in [49] which can deal with simultaneous BoT executions and bear low execution costs. Majed [50] proposed a structure for programmed tuning named APlug that gathers test execution times and constructs a model. Examinations were run on 16,384 CPUs, 480 centers on Linux bunch and 80 centers on Amazon EC2. The outcomes demonstrated that APlug is exact with insignificant overhead. Ioannis [51] assessed the utilization of reenacted strengthening and thermodynamic recreated tempering for booking in multi-cloud framework alongside virtual machines. The heuristics think about different targets while booking and attempt to advance both expense and execution. Calheiros and Buyya [52] focus on the issue of enery-effective execution of critical , CPU-serious Bag-of-Tasks applications. A cloud-mindful booking calculation was recommended that applies DVFS for empowering cutoff times and in this manner lessening vitality utilization. J. Octavio and Kwang [53] proposed a hereditary calculation for both spending plan and cutoff time compelled execution of BoT applications. Marco [54] in this work handle off base run-time evaluates by proposing an organized rescheduling calculation. Investigations were performed utilizing Grid'5000. The outcomes indicated 5% decrease accordingly time and 10% for stoppage measurements. Silva [55] examined the versatility of BoT applications running on multi-hub frameworks. 56]

VII. CONCLUSION

In multicluster environment, it is very difficult to do job allocation and scheduling. While doing job allocation, it is necessary to reduce makespan and flowtime. Another performance metric waiting time is also be reduced for better performance. It is solely feasible, if we select right algorithm for the job scheduling. This paper concludes that the metaheuristic can work much better if two or more algorithm will work in a combined way, only then optimal solution will be generated.

VIII. REFERENCES

[1]Kalra, Mala, and Sarbjeet Singh. "A review of metaheuristic scheduling techniques in cloud computing." Egyptian informatics journal 16.3 (2015): 275-295.

[2] Roberge, Vincent, Mohammed Tarbouchi, and Gilles Labonté. "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning."IEEE Transactions on Industrial Informatics 9.1 (2013): 132-141.

[3] Garg, Richa, and Saurabh Mittal. "Optimization by genetic algorithm."International Journal of Advanced Research in Computer Science and Software Engineering 4.4 (2014): 587-589.

[4] Yaseen, SaadGhaleb, and Nada MA Al-Slamy. "Ant colony optimization."IJCSNS 8.6 (2008): 351.

[5] Selvi, V., and Dr R. Umarani. "Comparative analysis of ant colony and particle swarm optimization techniques." International Journal of Computer Applications (0975–8887) 5.4 (2010).

[6] YunfengXu, Ping Fan, and Ling Yuan, "A Simple and Efficient Artificial Bee Colony Algorithm," Mathematical Problems in Engineering, vol. 2013, Article ID 526315, 9 pages, 2013. doi:10.1155/2013/526315

[7] Amandeep Kaur and AmitChhabra." A Review on Various Job Scheduling Algorithms" International Journal of Computational Intelligence Research, ISSN 0973-1873 Volume 13, Number 3 (2017), pp. 359-367(2017)

[8] Aote, Shailendra S., M. M. Raghuwanshi, and Latesh Malik. "A brief review on particle swarm optimization: limitations & future directions." International Journal of Computer Science Engineering (IJCSE) 14.1 (2013): 196-200..

[9] Rutenbar, Rob A. "Simulated annealing algorithms: An overview." IEEE Circuits and Devices Magazine 5.1 (1989): 19-26.

[10] Xinchao, Zhao. "Simulated annealing algorithm with adaptive neighborhood."Applied Soft Computing 11.2 (2011): 1827-1836.

[11] Xhafa, Fatos, and Ajith Abraham. "Computational models and heuristic methods for Grid scheduling problems."Future generation computer systems 26.4 (2010): 608-621.

[12] Jing, Weipeng, et al. "MaMR: High-performance MapReduce programming model for material cloud applications." Computer Physics Communications 211 (2017): 79-87.

[13] da Rosa Righi, Rodrigo, et al. "MigPF: Towards on self-organizing process rescheduling of Bulk-Synchronous Parallel applications." Future Generation Computer Systems (2016).
[14] Gabaldon, Eloi, et al. "Blacklist muti-objective genetic algorithm for energy saving in heterogeneous environments." The Journal of Supercomputing (2016): 1-16.
[15] Gerbessiotis, Alexandros V. "Extending the BSP model for multi-core and out-of-core computing: MBSP." Parallel Computing 41 (2015): 90-102.
[16] Gabaldon, E., et al. "Multi-criteria genetic algorithm applied to scheduling in multi-cluster environments." Journal of Simulation 9.4 (2015): 287-295.
[17] Amaris, Marcos, et al. "A simple bsp-based model to predict execution time in gpu applications." High Performance Computing (HiPC), 2015 IEEE 22nd International Conference on. IEEE, 2015.
[18] Enomoto, Takuto, and Masaomi Kimura. "Improving Population Diversity in Parallelization of a Real-Coded Genetic Algorithm Using MapReduce."
[19] Kajdanowicz, Tomasz, PrzemyslawKazienko, and WojciechIndyk. "Parallel processing of large graphs."Future Generation Computer Systems 32 (2014): 324-337.
[20] Liu, Xiaodong, et al. "BSPCloud: A hybrid distributed-memory and shared-memory programming model." International Journal of Grid and Distributed Computing 6.1 (2013): 87-97.
[21] Kirtzic, J. Steven, OvidiuDaescu, and T. X. Richardson. "A parallel algorithm development model for the GPU architecture."Proc. of Int'l Conf. on Parallel and Distributed Processing Techniques and Applications. 2012.
[22] Wu, Fuhui, Qingbo Wu, and Yusong Tan. "Workflow scheduling in cloud: a survey." The Journal of Supercomputing 71.9 (2015): 3373-3418.
[23] Sonmez, Ozan, et al. "Performance analysis of dynamic workflow scheduling in multicluster grids." Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. ACM, 2010.
[24] Yang, Cuicui, et al. "Bacterial foraging optimization using novel chemotaxis and conjugation strategies." Information Sciences 363 (2016): 72-95.
[25] Kumar, Neetesh, and DeoPrakashVidyarthi. "A novel hybrid PSO–GA meta-heuristic for scheduling of DAG with communication on multiprocessor systems."Engineering with Computers 32.1 (2016): 35-47.
[26] Kliazovich, Dzmitry, et al. "CA-DAG: Modeling communication-aware applications for scheduling in cloud computing." Journal of Grid Computing 14.1 (2016): 23-39.
[27] Guo, Fengyu, et al. "A workflow task scheduling algorithm based on the resources' fuzzy clustering in cloud computing environment." International Journal of Communication Systems 28.6 (2015): 1053-1067.
[28] Navimipour, NimaJafari, and FarnazSharifiMilani. "Task scheduling in the cloud computing based on the cuckoo search algorithm."International Journal of Modeling and Optimization 5.1 (2015): 44.

[29] Xu, Yuming, et al. "A hybrid chemical reaction optimization scheme for task scheduling on heterogeneous computing systems." IEEE Transactions on parallel and distributed systems 26.12 (2015): 3208-3222.
[30] COMPUTING, FIREFLY ALGORITHM IN CLOUD. "A load balancing model using firefly algorithm in cloud computing." Journal of Computer Science 10.7 (2014): 1156-1165.
[31] Kang, Yan, He Lu, and Jing He. "A PSO-based Genetic Algorithm for Scheduling of Tasks in a Heterogeneous Distributed System." JSW 8.6 (2013): 1443-1450.
[32] Babukartik, R. G., and P. Dhavachelvan. "Hybrid Algorithm using the advantage of ACO and Cuckoo Search for Job Scheduling."International Journal of Information Technology Convergence and Services 2.4 (2012): 25.
[33] Mezmaz, Mohand, et al. "A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems." Journal of Parallel and Distributed Computing 71.11 (2011): 1497-1508.
[34] Tawfeek, Medhat A., et al. "Cloud task scheduling based on ant colony optimization." Int. Arab J. Inf. Technol. 12.2 (2015): 129-137.
[35] Aron, Rajni, InderveerChana, and Ajith Abraham. "A hyper-heuristic approach for resource provisioning-based scheduling in grid environment."The Journal of Supercomputing 71.4 (2015): 1427-1450.
[36] George, Salu. "Hybrid PSO-MOBA for Profit Maximization in Cloud Computing." INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS 6.2 (2015): 159-163.
[37] Khan, Shagufta, and Niresh Sharma. "Effective scheduling algorithm for load balancing (SALB) using ant colony optimization in cloud computing." International Journal of Advanced Research in Computer Science and Software Engineering 4 (2014).
[38] Rao, Ch, and B. RaveendraBabu. "A Fuzzy Differential Evolution Algorithm for Job Scheduling on Computational Grids." arXiv preprint arXiv:1407.6317(2014).
[39] Chana, Inderveer. "Bacterial foraging based hyper-heuristic for resource scheduling in grid computing." Future Generation Computer Systems 29.3 (2013): 751-762.

[40] Singh, Susmita, et al. "Genetic algorithm based resource broker for computational Grid." Procedia Technology 10 (2013): 572-580.
[41] Tsai, Jinn-Tsong, Jia-Cen Fang, and Jyh-Horng Chou. "Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm." Computers & Operations Research 40.12 (2013): 3045-3055.
[42] Dasgupta, Kousik, et al. "A genetic algorithm (ga) based load balancing strategy for cloud computing." Procedia Technology 10 (2013): 340-347.
[43] Kim, Sung-Soo, et al. "Optimal job scheduling in grid computing using efficient binary artificial bee colony optimization." soft computing 17.5 (2013): 867-882.
[44] Kardani-Moghaddam, Sara, et al. "A hybrid genetic algorithm and variable neighborhood search for task scheduling problem in grid environment." Procedia Engineering 29 (2012): 3808-3814.
[45] Cai, Zhicheng, et al. "A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times in clouds." Future Generation Computer Systems (2017).
[46] Terzopoulos, George, and Helen D. Karatza. "Power-aware Bag-of-Tasks scheduling on heterogeneous platforms."Cluster Computing 19.2 (2016): 615-631.
[47] Moschakis, Ioannis A., and Helen D. Karatza. "A meta-heuristic optimization approach to the scheduling of Bag-of-Tasks applications on heterogeneous Clouds with multi-level arrivals and critical jobs."Simulation Modelling Practice and Theory 57 (2015): 1-25.
[48] Celaya, Javier, and UnaiArronategui. "Fair scheduling of bag-of-tasks applications on large-scale platforms."Future Generation Computer Systems 49 (2015): 28-44.
[49] Gutierrez-Garcia, J. Octavio, and Kwang Mong Sim. "Agent-based Cloud bag-of-tasks execution."Journal of Systems and Software 104 (2015): 17-31.
[50] Sahli, Majed, et al. "Automatic tuning of bag-of-tasks applications." Data Engineering (ICDE), 2015 IEEE 31st International Conference on.IEEE, 2015.
[51] Moschakis, Ioannis A., and Helen D. Karatza. "Multi-criteria scheduling of Bag-of-Tasks applications on heterogeneous interlinked clouds with simulated annealing." Journal of Systems and Software 101 (2015): 1-14.
[52] Calheiros, Rodrigo N., and RajkumarBuyya. "Energy-efficient scheduling of urgent bag-of-tasks applications in clouds through DVFS."Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on.IEEE, 2014.
[53] Gutierrez-Garcia, J. Octavio, and Kwang Mong Sim. "GA-based cloud resource estimation for agent-based execution of bag-of-tasks applications."Information Systems Frontiers 14.4 (2012): 925-951.
[54] Netto, Marco AS, and RajkumarBuyya. "Coordinated rescheduling of BagofTasks for executions on multiple resource providers."Concurrency and Computation: Practice and Experience 24.12 (2012): 1362-1376.
[55] da Silva, Fabricio AB, and Hermes Senger. "Scalability limits of Bag-of-Tasks applications running on hierarchical platforms." Journal of Parallel and Distributed Computing 71.6 (2011): 788-801.
[56] Kaur, Amandeep"hybrid bat and cuckoo based meta-heuristic algorithm for parallel job scheduling in multi-cluster environment.",International Journal of Advanced Research in Computer Science Vol. 9 Issue 1, p520-526. 7p.(2018)

