ISSN No. 0976-5697

Volume 8, No. 5, May – June 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Hybrid Meta-heuristics based scheduling technique for Cloud Computing Environment

Shubhdeep Kaur Sandhu
Dept. Of Comp Engg. & Technology
Guru Nanak Dev University
Amritsar, India

Anil Kumar
Dept. Of Comp Engg. & Technology
Guru Nanak Dev Universiity
Amritsar, India

Abstract: Cloud computing provides an environment where the distinct resources are delivered as a service to the customers/tenants over the internet. Herein, the core idea is to map the tasks to appropriate resources in order to optimize one or more objectives. As the resource allocation is categorized to be NP-Hard problem, there are no such algorithms that may find the optimal solution within genuine polynomial time. Hence, it is preferable to utilize meta-heuristic algorithms to find sub-optimal solutions in short duration of time. This paper has designed a hybrid technique for parallel scheduling in a cloud computing environment. The proposed technique has utilized mutation and crossover operators to improve the hybridization of Simulated Annealing (SA) with Particle Swarm Optimization (PSO). Thus, proposed technique can efficiently reduce the schedule length and flow time. Experimental results indicate that the proposed algorithm is more efficient than existing techniques.

Keywords: Cloud Scheduling, Meta-heuristics, Simulated Annealing, Tabu Search, Particle Swarm Optimization, Genetic Algorithm

I. INTRODUCTION

A. Overview

Advancements in cloud computing research in the past few years have led to the substantial commercial interest in using cloud infrastructures to reinforce commercial applications as well as services. It has emerged as a powerful computing paradigm and has been extensively approved in IT industry and academia [7]. It is realized as a usage model for supplying resources and information technology as “at scale” and “on demand” service in a multi-tenant environment where the resources are reclaimed from the internet by the virtue of web-based devices [11]. It is also conceived as a vision of utility computing, where the users need to pay only for the services they are using, which exactly resembles the way, users pay for other utilities, like electricity and telephone [9]. It aims at supplying computation services in the form of scalable and virtualized resources to massive distant users in heterogeneous cloud framework [16]. The elevation in the computation technologies as well as ever increasing calls for computing resources has laid down the cloud computing as the chief computing paradigm for either large or small scale IT enterprises. In order to analyze the execution of cloud services, we ought to carry out experimentations utilizing multiple user requirement groupings, but it is impractical in actual cloud framework because of cost overheads involved in cloud services. To limit the cost overheads, it is always favorable to use simulation environment in which experimentations can be carried out.

B. Resource Provisioning

The recognition of cloud computing has been commenced by the matter of fact that many enterprises go through the inadequacy of resources and the excessive cost overheads while they are on arising scale [12]. The primary objective of resource provisioning is to fully exploit the infrastructuralresources and to integrate themfor acquiring tremendous throughput so that wide-scale computation complications can be solved [14]. Based on the requirements of optimal allocation of resources, utilizing relevant means and attaining QoS are the key factors that play a central role while assigning the jobs to appropriate resources [1]. The customers no longer need to bother about,where the resources reside. On the basis of retrieved resources, services of higher level applications can be executed by the users at their respective regions and less computation is performed locally [13]. Generally, the shared pool of resources and decisions regarding suitable operations and dynamic supervision of resources is controlled by the cloud broker [15]. A broker is meant to allot the jobs to different resources and to achieve speed up in the terms of execution of an application.

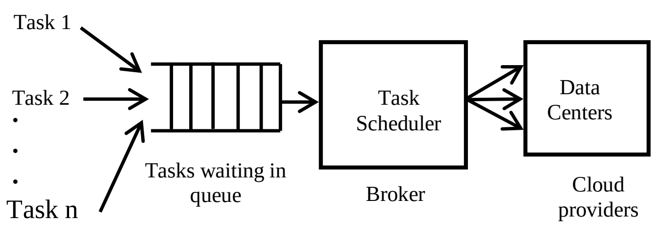

Figure 1. Cloud Scheduling Environment

C. Meta-heuristic Scheduling

Since the cloud computing has come forth as the most prominent distributed computing epitome amongst others in the current scenario, resource allocation has acquired sufficient consideration and large number of analysts have constructed comparable methods on the basis of virtual machines, heuristic approach, algorithms arising out of artificial intelligence [12]. The complexity of cloud scheduling has turned out to be a combinatorial optimization problem due to the presence of scalable and heterogeneous processing along with communication resources, which in turn, increases the complexity of scheduling [3][4][5][6]. Provided the seemingly immense computing resources of modern computing systems, unfortunately, there does not exist any polynomial time scheduling method in the contemplation of optimizing the resource allocation process as the majority of the scheduling issues are either NP-hard or NP-complete [2]. Meta-heuristic algorithms are also known as approximate algorithms. They make the use of iterative procedures to discover the optimal solutions in genuine time duration. Meta-heuristic algorithms provide a more adaptive approach to determine the optimal solution [2].The diversified meta-heuristic algorithms can be hybridizedto get better performing systems that integrate and exploit the complementary charactersof the underlyingpure optimization strategies. The hybridized algorithms are supposed to attain beneficial results from synergy. This is done to reinforce the numerous performance viewpoints such as computation time, quality of results or both. The selection of adequate consolidation of multiple algorithmic ideas is generally essential in accomplishing better performance in resolving complicated optimization problems [34]. For example, as most of the population based meta-heuristic algorithms are efficient for the supervision against the global minima, integrating more than two algorithms must have satisfactory logic. Else, it might back off meeting the adequate convergence speed without providing an improved result [2].

II. TECHNOLOGIES USED

A. Simulated Annealing

The name and motivation of Simulated Annealing originated from the annealing process in metallurgy [8], an approach that comprises heating and controlled cooling of solid material to expand the size of its crystals [2]. The fundamental idea is to sporadically acknowledge the poor solutions with an aim to prevent the objective function from being stuck in local minima during the convergence procedure [2]. This causes the SA algorithm, to explore the different regions of solution space, so that the probability of locating an improved solution is increased. The various permutations of a schedule are considered as distinct states of search space [10]. The decision regarding the probability of transitions between two respective neighboring states depends upon the energy discrepancy between the two states and the temperature of an entire system [20]. The cooling schedule computes the value of temperature that figures out the curve with reference to drop in temperature of the system [20]. A transition towards a higher energy state that is considered as a poor solution has higher probability even when the temperature of the system is high. In this manner, SA is permitted to be rescued from local minima.

B. Tabu Search

Tabu search is a well-known meta-heuristic optimization method formalized for combinatorial problems [33]. It is also recognized as a neighborhood search procedure [31]. Tabu Search supervises the exploration of the solution space by taking into account, a set of diversified probable moves that are neighbors to the current state [33]. It categorizes the certain moves as “tabu” (also known as forbidden moves), and stores them in an array called “Tabu list”, thus reducing the neighborhood search [28]. The format of searching a neighborhood for a particular solution is different from one iteration to another [31]. This technique avoids the cycling i.e., prevents the execution of the same series of moves indefinite number of times and conducts the search in the unexplored regions [28]. The usage of adaptive memory not only monitors the local information, but also some information regarding the exploration procedure [28][31].

C. Particle Swarm Optimization

PSO is biologically self-adaptive algorithm [24], influenced by the procession of organisms living together and their interaction amongst themselves in large groups [21]. More specifically, it manipulates the socio-behavioral characteristics observed in swarms of bees, a flock of birds or the school of fish, out of which the paradigm of Swarm Intelligence has emerged [24]. Earlier, by the virtue of authentic configuration of PSO, it was used to yield satisfactory solutions for continuous problems only. But, for some past years, it has been used to generate and represent the results for discrete problems such as scheduling optimization [32]. PSO makes the use of particle as its elementary concept [29]. It is instantiated by a certain number of particles which are initialized as random solutions [23][29]. Every particle possesses two representatives, i.e., position and velocity. Through a fixed number of iterations, the particles tend to migrate in search for better solutions in the search space [29]. Each one of the particles has its own adaptive speed, which conducts the apparent motion and remembers the local best position detected so far [27]. On the basis of its local best position as well as global best position in the whole population, each particle adjusts its trajectory [24].

D. Genetic Algorithm

Cloud Computing is coordinated with bio-inspired computing for intelligent resource consignment [15]. Conventional genetic algorithms are extensively fragile procedures that do not necessarily execute the massive instances of NP-complete problems as they do not consume prior expertise about the current issue. Genetic Algorithm resides inthe category of evolutionary algorithms that, by employing the operations that imitate the natural evolution like selection, crossover, mutation and inheritance, yields solutions to the optimization problems [17]. Each possible solution could be designated by a chromosome that is initialized randomly at the beginning [21]. At each stage, the finest individuals are chosen so that crossover operation is applied to gain the new individuals which are appended to the population [27]. With an objective to bring in, the random slight modification, mutation operator is implemented on the collection of new individuals that are chosen with low probability [27]. Herein, the fitness function is utilized to assess the solution that is most suitable.

III. RELATED WORK

Kashani, Mostafa Haghi, Mohsen Jahanshahi et al. 2009 [22] described the approaches that tried to minimize the cost overheads in communication as well as makespan, while maximizing the CPU utilization. Since the drawback of many heuristic algorithms is that, so much time is consumed in scheduling and subsequently require exhaustive time limit. With an aim to handle this shortcoming, researchers used memetic algorithm. Alongside determined memetic algorithm, Simulated Annealing was applied as a local search method. The comprehensive experimental conclusions validated the efficiency of the proposed methodology. Gan, Guo-ning, Ting-lei Huang, Shuai Gao et al. 2010 [25] dissected an optimized technique named, Genetic Simulated Annealing for scheduling tasks in cloud environment. The QoS prerequisites of various kinds of tasks corresponding to the features of client tasks in cloud were analyzed. Since the dimensional aspects of parameters were distinct and even magnitudinal orders were extremely different, they were dealt without dimensions. The results revealed the efficiency of the proposed algorithm in accomplishing the search and allocation of resources. Pandey, Suraj et al. 2010 [30] introduced PSO based heuristic approach in order to schedule the workflow using available cloud resources. This technique not only considered the cost of computation, but alsothe cost of data transmission as well. The workflow application was analyzed by fluctuating the range of communication and computation cost. The cost savings were compared while working with "Best Resource Selection" algorithm and Particle Swarm Optimization. The outcomes indicated that PSO can gain three times more cost savings in comparison to BRS and also provided satisfactory distribution of workload onto computation resources. Pop, Florin et al. 2013 [26] devised a multi-objective approach, Reputation Guided Genetic Scheduling algorithm, that was used to schedule independent tasks in inter-cloud framework. The proposed technique involved the aggregation of unlike objectives into a weighted sum for reputation function. Being an evolutionary measure, the selection state of genetic algorithm was used in the exploration phase of reputation. The proposed solution is evaluated while considering the load balancing as a means to estimate the optimization impact for suppliers and as a metric unit for client performance. The evolution and operational factors were observed with diversified probability values of GA operators. Yi, Pan, Hui Ding, and Byrav Ramamurthy et al. 2013 [19] conducted a research that was aimed at minimizing the cost overheads induced while acquiring the resources, that clients had requested from cloud networks. The joint issues of task scheduling and resource allocation were resolved with the performance evaluation of Tabu Search based approach. The comparison of former technique was done with Best-Fit method. The scrutinized results revealed that both the methods, i.e. Best-Fit as well as Tabu Search can achieve approximate optimal solutions to the corresponding Mixed Integer Linear Programming (MILP) solutions. Moreover, in most of the cases, in contrast to Best-Fit method, Tabu Search lowered the traffic blocking rate by 4~30%. Verma, Amandeep, and Sakshi Kaushal et al. 2014 [18] suggested a technique, named Bi-criteria Priority based Particle Swarm Optimization (BPSO). It was intended for scheduling synthetic workflow tasks across the accessible resources which minimized the execution time and cost underneath the given budget and deadline constraints. The advised algorithm was evaluated by employing simulation with dissimilar real work synthetic workload applications. The comparison was done with standard PSO and Budget Constrained Heterogeneous Earliest Finish Time (BCHEFT) and simulation proved that BPSO minimized the execution cost of schedule substantially under similar budget, price and deadline constraints. K. Padmaveni, D. John Aravindhar et al. 2016 [16] analyzed that inspite of the presence of various workflow scheduling algorithms, these cannot be enforced in cloud frameworks as they fail to assimilate heterogeneity and flexibility in cloud. The issue of workflow scheduling was demonstrated while considering the deadline constraint and makespan as the two main objectives. The authors proposed the Particle Swarm Memetic Algorithm which was the hybridization of Memetic algorithm and PSO. This heuristic was examined on numerous recognized scientific workflows. The acquired outcomes showed that PSMA executed better than other contemporary algorithms. DAG based encoding was deployed to yield a solution for multi-objective scheduling problem in cloud. Marwah Hashim Eawna, Salma Hamdy Mohammed, El-Sayed, M.El-Horbaty et al. 2015 [35] examined that the resource allocation procedures were intended for single tier applications. A dynamic resource provisioning was introduced in multi-tier applications by employing Simulated Annealing, Particle Swarm Optimization and also their hybrid technique. The results of simulation proved that the hybrid mechanism of SA and PSO is faster than individual pure PSO and SA algorithms as it took much lesser mean execution time. Thiago A. L. Genez 2015 et al. [36] presented a PSO-based methodology to direct the client in slicing the total CPU capacity (sum of frequency) amongst fixed number of resources to minimize the makespan of workflow. The technique was assessed and compared against the traditional approach that tended to choose similar frequency configurations for resources. The simulation results demonstrated that when the comprehensive amount of provided CPU frequency was constant, the PSO became capable for reducing the makespan. It was done by choosing the different CPU frequencies for resources.

IV. GAPS

Along with the success of Simulated Annealing and Tabu Search, there are also some issues that are needed to be dealt with, which are as follows:

A. Simulated Annealing:

i. It does not determine whether it has found the optimal solution. So, there is another complementary method required for this purpose along with Simulated Annealing.
ii. It also involves intensive computation, that requires a large amount of time and it is hard for it to ascertain actual cooling schedule.
iii. It assures to find the global solution, but in order to get that, it requires exponentially long code of cooling schedule. Therefore, it is impractical.

B. Tabu search:

i. Depending upon the given context, there are various possibilities concerning specific information that is recorded. The complete solutions can be recorded, but it needs huge storage and makes it high-priced to determine whether a potential move is tabu or not.
ii. Itsanother limitation is the tendency to fall into local optimization, i.e., it may get into rut during the search of an optimal solution.
iii. It is comparatively slow, as the number of alternatives must be assessed before an optimal solution is chosen.

V. PROBLEM DEFINITION

In the present work, the performance of standard Simulated Annealing and Tabu Search is evaluated for efficient parallel scheduling. A novel concept is introduced for parallel scheduling using mutation and crossover operators in order to improve the hybridization of SA and PSO. By using some well-known quality metrics like, makespan, average schedule length, mean flowtime, efficiency and utilization,the results of conventional algorithms (i.e., SA and Tabu Search) are compared with the suggested technique.

VI. PROPOSED METHODOLOGY

A. This section contains the graphicalrepresentation of proposed technique, consisting of various steps which are required to successfully accomplish the suggested algorithm.

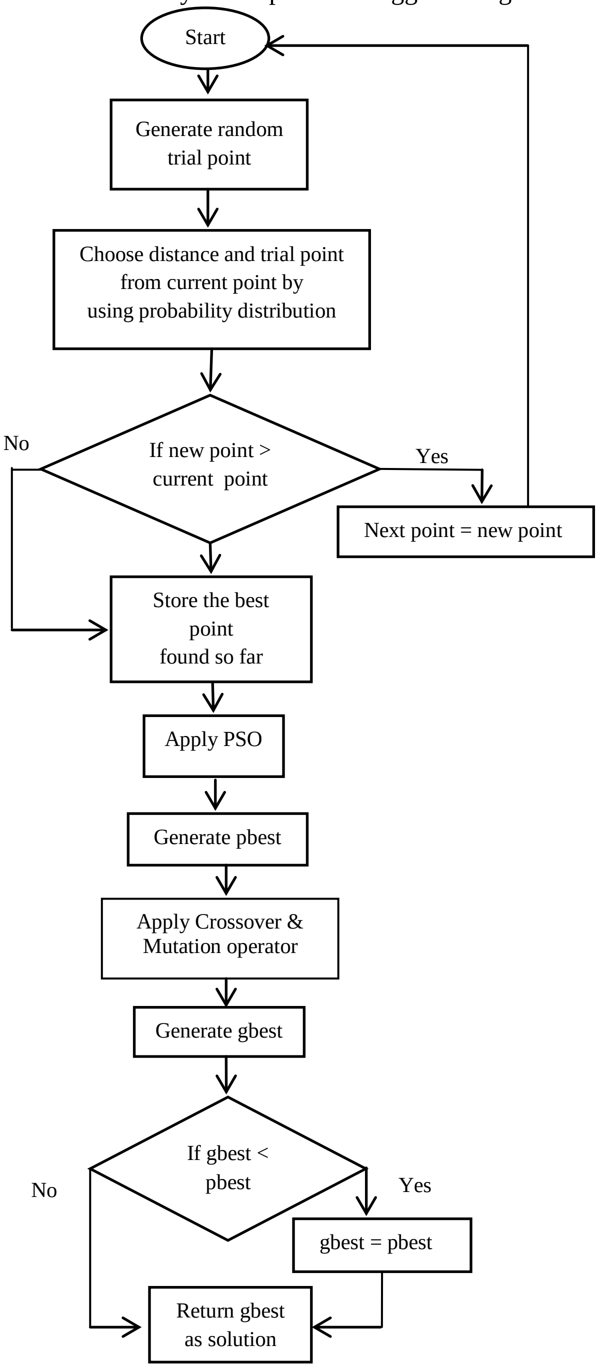

Figure 2. Flow chart of suggested method

B. The proposed technique is designed using MATLAB R2010a. For the evaluation of proposed algorithm in a controlled environment, a simulation is employed using Cloudsim simulator.

The subsequent section gives the details of the steps:

Step 1: Load initial workload (as jobs) for scheduling them in high end servers.

Step 2: Initialize cloud data centers (as ser) with high end servers along with their speed and the available resources.

Step 3:Develop batches of jobs.

```
max_res=max(jobs(:,3));
for j=1:max_res
k=1;
for i=1:size (jobs)
if jobs(i,3)==j
jbatch(j,k)=i;
k=k+1;
end
end
end
```

Step 4: Initialize random schedules for SA and assign suitable servers to them.

```
No_sol=50;
for j=1:No_sol
sol(:,j) = random_sol(100, 100);
end
```

Step 5: Assign jobs to servers by using the mapping of SA

```
for j=1:No_sol
sola(:,1) = sol(:,j);
sola(:,2) = serv;
for i=1:numel(sola(:,1))
for j=1:numel(sola(:,1))
if sola(i,1)==jobs(j,1)
sola(i,3)=jobs(j,2);
end
end
end
```

Step 6: Evaluate fitness function as makespan

```
for kk=1:numel(sola(:,1))
for i=1:Nser
if sola(kk,2)==(1+mod(i,Nser))
EX(1+mod(i, Nser))=EX(1+ mod(i, Nser)) +
ceil(sola(kk,3)/ser(i,3));
end
end
end
makespan(K)=max(EX);
```

Step 7: Evalulate best makespan so far

```
for j=1:No_sol
best=min(makespan);
end
A=sola(:,2);
B=sola(:,3)
C(:,1) = A(1:30);
C(:,2) = B(1:30);
x=sola(:,2);
y=sola(:,3);
```

Step 8: Apply PSO to improve the SA's best schedule further.

a. Evaluate velocity matrix

```
for  i=1:n
for j=1:n
d_ij(i,j)=sqrt((x(i)-x(j))^2 +(y(i)-y(j))^2 );
end
end
```

b. Evaluate and update velocity function and update pBest

```
for i=1:100
if lservc(i)<=l30
for k=1:n-1
v(job(i,k),job(i,k+1))=v(job(i,k),job(i,k+1))+10;
v(job(i,k+1),job(i,k))=v(job(i,k),job(i,k+1));
end
v(job(i,1),job(i,n))=v(job(i,1),job(i,n))+10;
v(job(i,n),job(i,1))=v(job(i,1),job(i,n));
i1=i1+1;
pcbest(i1,:)=job(i,:);
plbest(i1)=lservc(i);
end
end
[crossbest,j]=min(plbest);
mutebest=pcbest(j,:);
for nc=1:NC
tabu=ones(m,n);
tabu(:,1)=0;
schedule=ones(m,n);
for k=1:m
for step=1:n-1
```

Step 9: Evaluate and update schedule further using mutation and crossover operator

```
for i=1:m
lserv(i)=ca_serv(n,schedule(i,:),dij);
schedule1(i,:)=cross_serv_b
(schedule(i,:),mutebest,n);
schedule1(i,:)=cross_serv_b(schedule1(i,:),pcbest(i,:),
n);
schedule1(i,:)=mutation_b(schedule1(i,:),n);
lserv1(i)=ca_serv(n,schedule1(i,:),dij);
if lserv1(i)<lserv(i)
lserv(i)=lserv1(i);
schedule(i,:)=schedule1(i,:);
end
if lserv(i)<plbest(i)
plbest(i)=lserv(i);
pcbest(i,:)=schedule(i,:);
end
end
[crossbest,j]=min(plbest);
mutebest=pcbest(j,:);
lserv0=ca_serv(n,ts,dij);
if crossbest<lserv0
vs=mutebest;
lserv0=crossbest;
end
```

Step 10: Return best selected solution

```
for mk=1:fix(Nser/2)
for j=1:No_sol
sola(:,1) = sol(:,j);
sola(:,2) = serv;
for i=1:numel(sola(:,1))
for j=1:numel(sola(:,1))
if sola(i,1)==jobs(j,1)
sola(i,3)=jobs(j,2);
end
end
end
jj=j;
for kk=1:numel(sola(:,1))
for i=1:Nser
if sola(kk,2)==(1+mod(i,5))
EX(1+mod(i,Nser))=
EX(1+mod(i,Nser))+sola(kk,3)+
ceil(sola(kk,3)/(max(ser(:,3))+ceil(lserv0/NC^2)));
end
end
end
```

Step 11: Update schedule if current solution has lesser makespan than actual one.

```
for j=1:No_sol
best1=min(makespan);
end
if best>best1
best=best1;
end
end
makespan(K)=max(EX);
```

Avg_{sl} = mean(EX)/(Nser+ceil(lserv0(NC^2));

```
serial_time=sum(EX);
end
```

VII. RESULTS AND DISCUSSION

A. Performance Analysis

We have used the initially loaded workload that contains 100 jobs that need to be scheduled. The burst time of all the jobs is assumed to be lying within the range of 50-51. We consider that each data centre has 8 servers which are allotted a maximum of 5 resources and 2-5 processors each. The speed of individual processor lies within 1-3 GHz. More is the speed of processors, lesser is the execution time of the jobs. The job batches are developed according to the resource requirement. The jobs having the same number of resource requirement would be in the same batch. The results are taken after the loop of 100 iterations, out of which, maximum execution time is marked as makespan of the first schedule. In the same way, all the parameters are calculated.

B. Results

The graph analyses the methodologies in accordance with the parameters being utilized for the assessment of efficiency of resource-aware scheduling. The mean flowtime, makespan, average schedule length, efficiency and utilization are utilized as metrics to compute the results of suggested algorithm. In the tables given below, we have written 15 values of each parameter and have compared the recorded values of two existing techniques, i.e. Simulated Annealing (denoted as “Existing 1”) and Tabu Search (denoted as “Existing 2”) against the proposed hybrid technique (denoted as “Proposed”). The graphical simulation results for all the metrics are depicted in the figures given below.

i. Flow Time: In general, it is defined as the sum of completion times of all tasks in the system. It determines the time interval between the tasks that arrive the first and the departure of the last completed task.

   Flowtime = $Flowtime = \frac{(\sum_{i=1}^{N} ET_i)}{N}$, where N represents total number of tasks

   While comparing the readings in Table 1, we can say that proposed method has been proved better than Simulated Annealing and Tabu Search. Figure 3 shows that the proposed algorithm optimizes the mean flowtime in an efficient way, in comparison to both the existing techniques.

Table 1. Mean flowtime values

Sr No	Existing 1	Existing 2	Proposed
1	211.0083	180.3900	152.7783
2	186.3200	196.0800	142.0317
3	196.6850	215.9500	154.5483
4	206.1017	206.1583	157.0467
5	236.1900	170.1650	169.4867
6	196.9483	167.7783	161.8383
7	177.1517	183.5300	160.5083
8	202.3233	236.6433	144.0383
9	188.3383	226.5100	176.3500
10	194.6517	191.3650	158.6367
11	186.4050	220.3067	170.1333
12	174.9300	189.6583	160.5900
13	196.4733	194.1133	151.1900
14	183.9983	160.4567	135.9817
15	156.2150	189.3517	165.1833

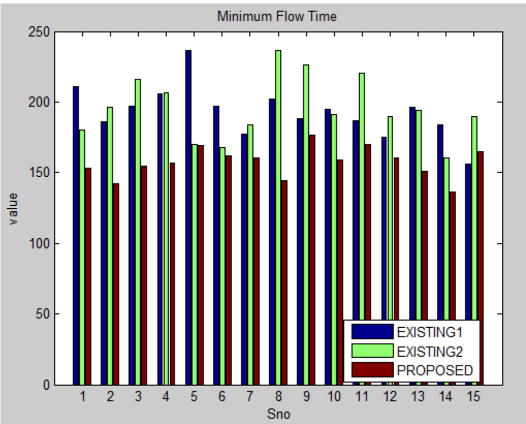

Figure 3. Mean flowtime comparison

ii. Makespan: It is defined as the amount of time, which elapses from beginning to the end in order to finish a sequence of jobs or tasks pertaining to a specific workload. In other words, it is considered as the maximum finishing time of task.

$Makespan = Max(SL_i)$, where i=1, 2….P and P represents the total number of processors.

Table 2 shows the results of comparison in the terms of makespan values. Figure 4 describes that the computed makespan while using the proposed method is lesser than the values evaluated using existing techniques.

Table 2: Makespan values

Sr No	Existing 1	Existing 2	Proposed
1	708	505	380.5000
2	506	606	405
3	609	659	380
4	606	606	480.5000
5	558	456	478.5000
6	708	553	430.5000
7	658	661	404.5000
8	758	758	404
9	555	860	429
10	704	859	405.5000
11	607	704	529.5000
12	605	709	404
13	859	555	369.6667
14	660	606	379.5000
15	605	654	403.5000

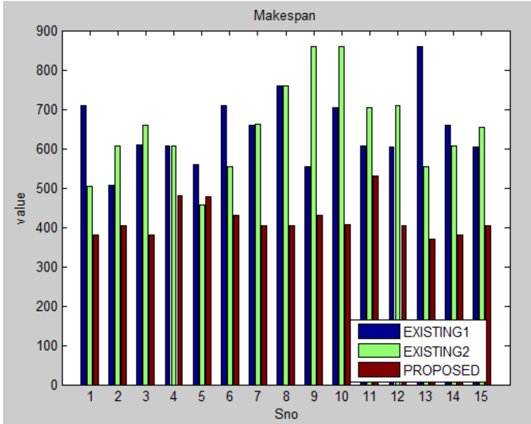

Figure 4. Makespan comparison

iii. Average Schedule Length: Minimization of schedule length is one of the major objectives in distributed cloud scheduling. It can be defined as the time taken by the schedule to finish all tasks in the system. Minimizing the schedule length leads to the best utilization of system resources.

$Asl = \sum \frac{schedule\ lengt\,h(i)}{P}$, where P represents the number of processors.

Based on the simulation results in Table 3 and the graphical representation in Figure 5, it is clear that the proposed algorithm adapts the best possible approach of task scheduling which tends to minimize the schedule length that leads to better performance.

Table 3: Average schedule length values

Sr No	Existing 1	Existing 2	Proposed
1	94.9600	73.7556	80.2500
2	98.2900	86.0889	89.9722
3	105.1400	86.4889	88.0417
4	114.9600	99.9556	83.8056
5	104.1600	98.4556	85.6111
6	90.4200	87.4333	85.6667
7	96.4900	85.9889	83.6667
8	96.7600	103.6556	81.9167
9	100.3200	106.1889	93.4722
10	98.6700	92.0333	94.4444
11	113.4000	86.5667	71.3889
12	97.9900	89.8444	89.9306
13	96.0800	89.5444	88.1528
14	96.5300	97.6222	86.5694
15	90.0900	85.7333	97.7222

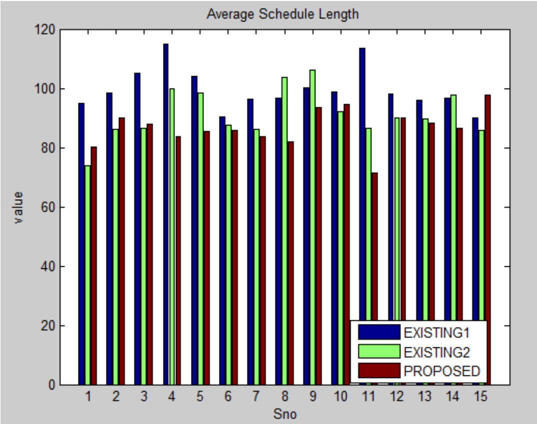

Figure 5. Average schedule length comparison

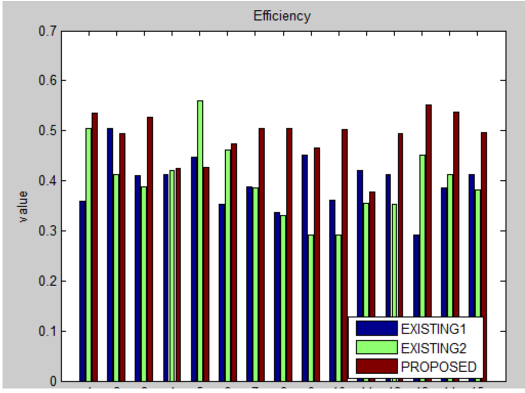

Figure 6. Efficiency comparison

iv. Efficiency: It is the evaluation of the degree to which the input data are considerably used for a specific task or an output function. It is quantitatively specified by the ratio of output to the total input. It is the comparative analysis method of what is actually acquired with what can be attained with same usage of resources.

$Efficiency = \frac{Speed\ up}{P}$, where P represents the total number of processors.

The experimental results of existing and proposed techniques are listed in Table 4. The graph in Figure 6 is showing comparison of efficiency between the traditional algorithm and proposed algorithm.As it can be seen that the proposed method is more efficient than other methods.

Table 4: Efficiency values

Sr No	Existing 1	Existing 2	Proposed
1	0.3602	0.5050	0.5361
2	0.5040	0.4125	0.4938
3	0.4105	0.3869	0.5263
4	0.4125	0.4208	0.4246
5	0.4480	0.5592	0.4263
6	0.3531	0.4611	0.4739
7	0.3875	0.3858	0.5043
8	0.3364	0.3298	0.5050
9	0.4505	0.2907	0.4662
10	0.3622	0.2910	0.5031
11	0.4201	0.3551	0.3777
12	0.4132	0.3526	0.4950
13	0.2910	0.4505	0.5518
14	0.3864	0.4125	0.5375
15	0.4132	0.3823	0.4957

v. Resource utilization: It refers to the total amount of resources consumed actually, compared against the resources aforethought for a particular task.

$Resource\ Utilization = \frac{Efficiency}{Ns}$, where Ns represents the number of servers.

Table 5 shows the results of resource utilization as obtained by implementing existing and proposed techniques. The Figure 7 depicts better performance of suggested method than the existing algorithms in terms of resource utilization.

Table 5: Resource utilization values

Sr No	Existing 1	Existing 2	Proposed
1	0.1139	0.1597	0.1896
2	0.1594	0.1305	0.1746
3	0.1298	0.1224	0.1861
4	0.1305	0.1331	0.1501
5	0.1417	0.1768	0.1507
6	0.1117	0.1458	0.1675
7	0.1226	0.1220	0.1783
8	0.1064	0.1043	0.1785
9	0.1424	0.0919	0.1648
10	0.1145	0.0920	0.1779
11	0.1328	0.1123	0.1335
12	0.1307	0.1115	0.1750
13	0.0920	0.1424	0.1951
14	0.1222	0.1305	0.1901
15	0.1307	0.1209	0.1752

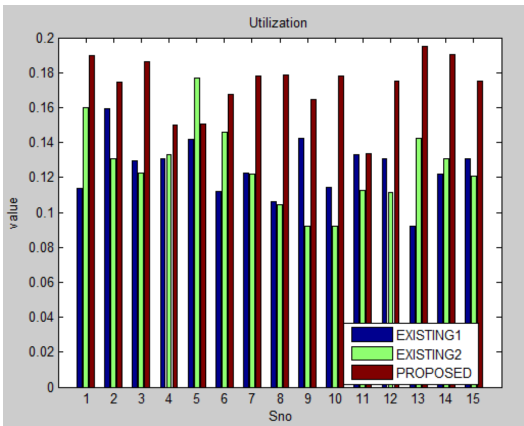

Figure 7. Resource utilization comparison

vi. Mean

It refers to the evaluation of central tendency either of probability distribution or of a random variable which is characterized by that distribution.

$Mean(\bar{x}) = \frac{\sum x}{n}$, where

x represents the sum of all the elements in a data set and

n represents the total number of elements in a data set.

vii. Sample Standard Deviation

It yields the standard deviation of population on the basis of a random sample. It evaluates the dispersion of data around the sample mean.

$Sample\ Standard\ Deviation = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})}$,

where $\bar{x}$ represents the mean of sample elements,

n represents the number of elements in a sample and

x_i represents each of the values of a sample

Table 6: Comparison of mean and sample standard deviation values

Para-meters	Existing 1		Existing 2		Proposed	
	$\bar{x}$	σ	$\bar{x}$	σ	$\bar{x}$	σ
MFt	192.91	18.15	195.23	22.20	157.35	11.05
Mp	647.06	89.74	650.06	116.17	418.94	44.75
Asl	99.61	7.16	91.29	8.45	86.70	6.38
Eff	0.39	0.05	0.39	0.07	0.48	0.04
Ut	0.12	0.01	0.12	0.02	0.17	0.01

The abbreviations used in Table 6 are as follows:
MFt = Mean Flowtime,
Mp = Makespan,
ASL = Average Schedule Length,
Eff = Efficiency and
Ut = Utilization

The mean flowtime, makespan and average schedule length are showing less mean and lesser standard deviation. The lesser standard deviation for proposed technique depicts that proposed technique is more consistent with respect to these three parameters.

But in case of efficiency and utilization, the mean of proposed technique is more and standard deviation is less which exhibits that proposed method provides better efficiency with good consistency as compared to earlier methodologies.

VIII. CONCLUSION AND FUTURE WORK

Much attention has been paid to optimistic grid scheduling synthesis and optimization by using meta-heuristic approaches. In general, Simulated Annealing (SA) is able to provide good solutions, but with large computational efforts. In this novel study, the hybrid technique for parallel cloud scheduling using mutation and crossover operators based on SA and PSO is introduced. SA is used for topology optimization, while job allocations are supervised PSO and the sub-optimal solutions are enhanced by using mutation and crossover operators. The experimental results demonstrate that the suggested method outperforms the available techniques with respect to different quality metrics. The further enhancement of this study involves the evaluation of algorithm by simulating it actual scientific workload in a real-time cloud computing environment. In this work, we have neglected the effect of failures in cloud data centres. Therefore, in the near future, we will propose fault-tolerance based technique to enhance the results.

REFERENCES

[1] Alla, Hicham Ben, Said Ben Alla, and Abdellah Ezzati. "A novel architecture for task scheduling based on Dynamic Queues and Particle Swarm Optimization in cloud computing." *Cloud Computing Technologies and Applications (CloudTech), 2016 2nd International Conference on*. IEEE, 2016.

[2] Tsai, Chun-Wei, and Joel JPC Rodrigues. "Metaheuristic scheduling for cloud: A survey." *IEEE Systems Journal* 8.1 (2014): 279-291.

[3] Javadi, Bahman, Mohammad K. Akbari, and Jemal H. Abawajy. "A performance model for analysis of heterogeneous multi-cluster systems." *Parallel computing* 32.11 (2006): 831-851.

[4] Schuurman, Petra, and Gerhard J. Woeginger. "Polynomial time approximation algorithms for machine scheduling: Ten open problems." *Journal of Scheduling* 2.5 (1999): 203-213.

[5] Jones, William M., et al. "Characterization of bandwidth-aware meta-schedulers for co-allocating jobs across multiple clusters." *The Journal of Supercomputing* 34.2 (2005): 135-163.

[6] Li, Kai, and Shan-lin Yang. "Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms." *Applied mathematical modelling* 33.4 (2009): 2145-2158.

[7] Koutsandria, Georgia, et al. "Can everybody be happy in the cloud? Delay, profit and energy-efficient scheduling for cloud services." *Journal of Parallel and Distributed Computing* 96 (2016): 202-217.

[8] Al-Olimat, Hussein S., et al. "Cloudlet scheduling with particle swarm optimization." *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on*. IEEE, 2015.

[9] Buyya, Rajkumar, et al. "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility." *Future Generation computer systems* 25.6 (2009): 599-616.

[10] Dong, Lingbo, et al. "A comparison of a neighborhood search technique for forest spatial harvest scheduling problems: A case study of the simulated annealing algorithm." *Forest Ecology and Management* 356 (2015): 124-135.

[11] Abdullah, Monir, and Mohamed Othman. "Simulated annealing approach to cost-based multi-quality of service job scheduling in cloud computing environment." *American Journal of Applied Sciences* 11.6 (2014): 872-877.

[12] Feng, Mingyue, et al. "Multi-objective particle swarm optimization for resource allocation in cloud computing." *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*. Vol. 3. IEEE, 2012.

[13] Sebastio, Stefano, Giorgio Gnecco, and Alberto Bemporad. "Optimal distributed task scheduling in volunteer clouds." *Computers & Operations Research* 81 (2017): 231-246.

[14] Vinothina, V., and R. Sridaran. "A survey on resource allocation strategies in cloud computing." *International Journal of Advanced Computer Science & Applications* 1.3 (2012): 97-104.

[15] Xavier, T. Aleena, and R. Rejimoan. "Survey on various resource allocation strategies in cloud." *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*. IEEE, 2016.

[16] Padmaveni, K., and D. John Aravindhar. "Hybrid memetic and particle swarm optimization for Multi objective scientific workflows in cloud." *Cloud Computing in Emerging Markets (CCEM), 2016 IEEE International Conference on*. IEEE, 2016.

[17] Kacem, Imed. "Genetic algorithm for the flexible job-shop scheduling problem." *Systems, Man and Cybernetics, 2003. IEEE International Conference on*. Vol. 4. IEEE, 2003.

[18] Verma, Amandeep, and Sakshi Kaushal. "Bi-criteria priority based particle swarm optimization workflow scheduling algorithm for

cloud." *Engineering and Computational Sciences (RAECS), 2014 Recent Advances in*. IEEE, 2014.

[19] Yi, Pan, Hui Ding, and Byrav Ramamurthy. "A tabu search based heuristic for optimized joint resource allocation and task scheduling in grid/clouds." *Advanced Networks and Telecommuncations Systems (ANTS), 2013 IEEE International Conference on*. IEEE, 2013.

[20] 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.

[21] Effatparvar, Mehdi, et al. "Swarm Intelligence Algorithm for Job Scheduling in Computational Grid." *Intelligent Systems, Modelling and Simulation (ISMS), 2016 7th International Conference on*. IEEE, 2016.

[22] Kashani, Mostafa Haghi, and Mohsen Jahanshahi. "Using simulated annealing for task scheduling in distributed systems." *Computational Intelligence, Modelling and Simulation, 2009. CSSim'09. International Conference On*. IEEE, 2009.

[23] Elhady, Gamal F., and Medhat A. Tawfeek. "A comparative study into swarm intelligence algorithms for dynamic tasks scheduling in cloud computing." *Intelligent Computing and Information Systems (ICICIS), 2015 IEEE Seventh International Conference on*. IEEE, 2015.

[24] Gabaldon, Eloi, et al. "Particle Swarm Optimization Scheduling for Energy Saving in Cluster Computing Heterogeneous Environments." *Future Internet of Things and Cloud Workshops (FiCloudW), IEEE International Conference on*. IEEE, 2016.

[25] Gan, Guo-ning, Ting-lei Huang, and Shuai Gao. "Genetic simulated annealing algorithm for task scheduling based on cloud computing environment." *Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on*. IEEE, 2010.

[26] Pop, Florin, et al. "Reputation guided genetic scheduling algorithm for independent tasks in inter-clouds environments." *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, 2013.

[27] Toader, Florentina Alina. "Evolutionary algorithms for job shop scheduling." *Electronics, Computers and Artificial Intelligence (ECAI), 2016 8th International Conference on*. IEEE, 2016.

[28] Armentano, Vinı´ cius A., and Denise S. Yamashita. "Tabu search for scheduling on identical parallel machines to minimize mean tardiness." *Journal of intelligent manufacturing* 11.5 (2000): 453-460.

[29] Wang, Xiaotong, et al. "Scheduling Budget Constrained Cloud Workflows with Particle Swarm Optimization." *Collaboration and Internet Computing (CIC), 2015 IEEE Conference on*. IEEE, 2015.

[30] Pandey, Suraj, et al. "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments." *Advanced information networking and applications (AINA), 2010 24th IEEE international conference on*. IEEE, 2010.

[31] Boicea, Valentin A. "Distribution grid reconfiguration through Simulated Annealing and Tabu Search." *Advanced Topics in Electrical Engineering (ATEE), 2017 10th International Symposium on*. IEEE, 2017.

[32] Chenyang, Gao, Gao Yuelin, and Lv Shanshan. "Improved simulated annealing algorithm for flexible job shop scheduling problems." *Control and Decision Conference (CCDC), 2016 Chinese*. IEEE, 2016.

[33] 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 ModellinPractice and Theory* 57 (2015): 1-25.

[34] Blum, Christian, et al. "A brief survey on hybrid metaheuristics." *Proceedings of BIOMA* (2010): 3-18.

[35] Eawna, Marwah Hashim, Salma Hamdy Mohammed, and El-Sayed M. El-Horbaty. "Hybrid Algorithm for Resource Provisioning of Multi-tier Cloud Computing." *Procedia Computer Science* 65 (2015): 682-690.

[36] Genez, Thiago AL, et al. "A Particle Swarm Optimization Approach for Workflow Scheduling on Cloud Resources Priced by CPU Frequency." *Utility and Cloud Computing (UCC), 2015 IEEE/ACM 8th International Conference on*. IEEE, 2015.