Journal of Scientific & Industrial Research
Vol. 81, July 2022, pp. 776-784

Energy Aware Genetic Algorithm for Independent Task Scheduling in Heterogeneous Multi-Cloud Environment

Roshni Pradhan* & Suresh Chandra Satapathy
KIIT Deemed to be University, Bhubaneswar 751 024, India

Received 08 December 2021; revised 17 June 2022; accepted 18 June 2022

Cloud datacentres contain a vast number of processors. The rapid expansion of cloud computing is resulting in massive energy usage and carbon emissions which has reported a substantial increase day by day. Consequently, the cloud service providers are looking for eco-friendly solutions. The energy consumption can be evaluated with an energy model, which identifies that, server energy consumption scales linearly with resource (cloud) utilization. This research provides an alternate solution to task scheduling problem which designs an optimized task schedule to minimize the makespan and energy consumptions in cloud datacenters. The proposed method is based on the principle of Genetic Algorithm (GA). In the context of task-scheduling using GA, chromosomal representation is considered as a schedule of set of independent tasks mapped with available cloud or machine in the proposed methodology. A fitness function is taken to optimize the overall execution time or makespan. Energy consumption is evaluated based on minimum makespan value. The proposed technique also tested upon synthesized and benchmark dataset which outperforms the conventional cloud task scheduling algorithms like Min-Min, Max-Min, and suffrage heuristics in heterogeneous multi-cloud system.

Keywords: Cloud computing, Datacenters, Genetic algorithm, Makespan, NP-complete

Introduction

Cloud computing is a very popular client-server-based environment that is influenced by the concept of parallel and distributed computing. Various services are offered by Cloud Service Providers (CSP) out of which the most important are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).[1] In IaaS various computing resources are provided by CSP. It includes computing power, storage, processing unit, bandwidth, etc. Hence due to this IaaS service, the cloud offers infinite computing facilities. Besides these, it also offers on-demand service, elastic computing, pay-as-use facilities. These key features of the cloud make it more popular in the information technology industry.[2] A cloud environment may be homogeneous or heterogeneous. It may be of one cloud or constituents of more than one cloud which is specifically known as a multi-cloud environment.[3,4] In this environment, requests are accepted from consumers in the form of tasks. These tasks are accepted by CSP to provide end-service to consumers. The cloud service providers are implementing a number of client apps (incoming task) without much thought to energy usage in order to fully exploit the cloud resources (particularly data centres). Nowadays, Energy consumption has become a crucial consideration in cloud datacenters. As a result, efforts are being undertaken to minimise energy consumption and carbon emissions through effective resource management and usage. This leads to inclusion of an efficient task scheduling mechanism, which aims to minimize energy consumption in a heterogeneous cloud environment by optimizing the scheduling parameters. [5]

Cloud task scheduling problem is known to be NP-complete.[6] During task scheduling, parameters like makespan, deadline time, completion time, overall cloud or server utilization, and average energy utilization parameters are taken for optimization purposes. To evaluate the performance of distributed cloud computing system, cloud engineers are approaching nature-inspired optimized scheduling techniques. The ultimate aims of these scheduling methods are to minimize makespan and energy consumption at datacenters, simultaneously utilize the cloud resources efficiently. Evolutionary algorithms like Genetic Algorithm (GA), Differential Evolution, and Simulate annealing, etc which are most popular nature-inspired evolutionary algorithms

*Author for Correspondence
E-mail: roshni.pradhanfcs@kiit.ac.in

provides large solution space. Among them, GA is a type of heuristics that focuses on Darwinian evolution principles. In this work, GA is used with specific mathematical parameters like initialization, selection, crossover and mutation.[10–12] It is more flexible and can be easily applied for scheduling purposes in a multi-cloud environment. Inherent parallelism in GA helps to reduce the overall execution time. It operates on a set of solutions rather than a single solution. It begins with the initialization of the population followed by selection. Then new population is generated in the crossover phase. The diverse solution undergoes the mutation process. An objective function otherwise known as a fitness function is there to keep track of performance of each solution. The best solution is allowed to survive and ready to generate an efficient schedule. Best or efficient schedule is choosen based on the fitness function value (makespan). The corresponding energy consumption is also calculated for the generated schedule with the help of energy evaluation method used in one of the section in this paper.

The rest of this paper is organized as follows. Background related work is given in the next section, followed by different mathematical consideration used for the proposed methodology. After that, proposed algorithm is described within an illustration and the experimental result is tabulated followed by conclusion of the work.

Backgrounds and Related Work

Energy estimation and cloud resources usage are profoundly coupled.[13,14] Low cloud resource usage is hollering an unsuitable measure of energy when they are completely used or adequately stacked. In the year 2005, a new survey showed that the energy consumption and carbon emissions of large data centres in the United States were at an all-time high. Datacenters in Europe have been estimated to use 1% of total carbon emissions, whereas in the United States it is 2.8% at the same time.[14–18] In distributed computing, the basic equipment framework is hidden from the end client. Despite the fact that application solicitation can be thought about for less utilization of energy and maximum usage of available cloud resources. Cloud resources ought not to be over-burden or under loaded by the undertakings rather ought to be utilized ideally.[9–19]

Many task scheduling problem statements have been proposed by the researchers who have optimized the scheduling parameter to achieve a near-optimal solution. Among those research works, one of the method describes that task scheduling can be done with the combination of GA and list scheduling.[20] In this method, they have followed two phase mechanism. In the first phase, GA technique is implemented and in the later part list scheduling is carried out. It was found to be efficient and overally good to some extent. In one of the research, it has proved a solution where non-pre-emptive tasks are allocated to the multiprocessor system.[21] In this method the algorithm overall execution time is considered and could have been more optimized. Another scheduling technique has been proposed by a study which was also found to be an efficient method and it has considered List scheduling techniques again for preparing an efficient schedule. Specifically, List Scheduling uses heuristics to choose the set of activities that should be scheduled in the upcoming cycle from among all of the jobs that are prepared for execution.[22] In another proposed work, a solution has been found which has combined four list heuristics that aim for minimum execution time.[23] Most of the task scheduling methods are focusing on minimization of overall completion time i.e Makespan. However it is a matter of concerned to provide a task scheduling method which can also minimize energy consumption in the datacenters.

In the year 2000–2005, a large number of data centres have been required in the cloud environment to fulfil user requests. About more than 16% of data centres have been added to the existing ones. It results from a high increment in power consumption and carbon emission (approximately 76%). Observing this change many laws and regulations were proposed by the government. The European code of ethics, the Energy Star programme, and the 80 PLUS24 projects were all introduced in the United States. There are two types of cloud computing system architecture: hardware execution and software application based. Hardware implementation based architecture is easy to handle and the circuits are modifiable to reduce energy consumption and carbon emission. However, it is a very difficult task to manage in a cloud environment. The study of energy consumption in a multi-cloud environment has become a popular field of research. In most of the research papers authors have addressed the same problem but using an evolutionary algorithm is rarely found. In Lee & Zomaya[25], two methods are proposed and energy consumption and cloud utilization parameters are evaluated. In cloud data centres, the resource and energy consumption using an efficient task scheduling method is applied as suggested by a research work.[19] But the performance evaluation is not conducted in this paper. Energy-aware task consolidation is proposed by some authors[26,27] which aims to restrict CPU utilization. In, one method[28], hierarchical scheduling methods is proposed to optimize the energy parameters.

Theoretical Considerations

Cloud Task Scheduling Model

Cloud task scheduling has two perspectives. One is user or client perspective and other is cloud service provider perspective. User requirement is QoS with minimal pricing of services where as service provider offers good QoS with profit. All these are managed by Service Level Agreement (SLA). It is decided by the cloud service provider and the user. To achieve each perspective, different approaches can be adapted which can result an optimized decision. Task manager is available to gather information regarding the available clouds, their services and the pricing. Based on SLA, client can access the services from private or public cloud. The service interface is located in a client broker side as an API. From service provider perspective, the scheduler allocates the upcoming request to the virtual machines.

A cloud environment is a set-up that consists of a group of machines or resources and a set of tasks. These resources are present in cloud-datacenters with high-end servers. To keep track of all the resources and tasks, a centralized task manager is embedded in a cloud environment. Data transfer cost is considered to be negligible between task managers and datacenters. Using GA an efficient schedule is prepared which keeps records about the overall completion time, cloud utilization, and energy consumption.

Application Model

In the scheduling model as described, a set of tasks are taken. The execution time for each of the tasks is previously estimated. This is represented with the help of the ETC matrix which is input to the computing environment. As this computation takes place in a static environment, all information about the task scheduling is known before. This kind of application model is specifically designed for IaaS cloud environments. It also assumed that there is no interference between the tasks and the cloud resources like storage, I/O, etc.

Energy Model

A simplified method for calculation of amount of energy consumption in cloud datacenter is considered.[25] It is easy, less complex for energy consumption calculation for multi cloud environment. Here it is assumed that cloud or servers are on-active power saving mode for idle slots. Task processing time and cloud utilization are two parameters considered for this purpose. It has been observed from some studies that Average cloud utilization is linearly related to energy consumption. Hence, cloud utilization needs to be optimized to reduce energy consumption. Cloud utilization is the percentage of time cloud is busy to serve the tasks. To estimate the overall energy utilization or consumption, the overall execution time of all the tasks on machine or cloud is required which can be taken from the ETC matrix. Average Cloud utilization U can be estimated with Eq. 1 for m number of clouds or machines.

$$U = (\sum_{i=1}^{m} CU(C_i))/m \qquad \text{... (1)}$$

Each cloud's energy consumption at any given time may be calculated as given in Eq. 2 .[25,28]

$$E_i = (P_{\max} - P_{\min}) \times U_i + P_{\min} \qquad \text{... (2)}$$

where,

C_i = ith cloud utilization

P_{max} is maximum power consumption value during high load (or 100% cloud utilization)

P_{min} is active mode power consumption (or as low as 1% utilization).

The overhead to turn off idle resources is insignificant in this case. Hence it is not considered. Average energy consumption is denoted as in Eq. 3.

$$E = (\sum_{i=0}^{m} E_i)/m \qquad \text{... (3)}$$

Scheduling Model using GA

Independent tasks set $T = \{T_1, T_2, T_3, \ldots\ldots T_n\}$ is taken in a heterogeneous cloud environment. Along with this set of cloud $C = \{C_1, C_2, C_3, \ldots.. C_m\}$ is appointed as resource. Each task takes some amount of time for execution on each cloud. It is represented by the Expected Time to Compute (ETC) matrix. Here $ETC_{i,j}$ represents the execution time of task T_i on cloud C_j where, $1 \le i \le n$ and $1 \le j \le m$.

Another important part of the scheduling model is to represent an effective schedule. In this proposed method, GA is used to generate an efficient schedule that can optimize the cloud parameters.

$$ETC_{i,j} = \begin{array}{cccccc} & C_1 & C_2 & . & C_m \\ T_1 & ETC_{1,1} & ETC_{1,2} & . & ETC_{1,m} \\ T_2 & ETC_{2,1} & ETC_{2,2} & . & ETC_{2,m} \\ .... & .... & .... & ... & .... \\ T_n & ETC_{n,1} & ETC_{n,2} & . & ETC_{n,m} \end{array}$$

The Proposed Genetic Algorithm based task Scheduling Algorithm

The method used in this paper started with an initial population of feasible solutions. Crossover and mutation operators are applied to the solution set to obtain the optimal solution. The best solution in terms of the fitness function is to find out with the help of makespan (Fig. 1). According to GA principles, a solution is initialized randomly. The solution set contains the information regarding the cloud identity to which tasks are assigned.

The main objective of the scheduling algorithm using GA is to get an optimized result with the help of a fitness function. Here fitness function is a minimization function. Makespan for each schedule is calculated and simultaneously average machine or cloud utilization and energy consumed during the process are estimated. The genetic Algorithm-based task Scheduling technique is given in the following table (Algorithm 1, Algorithm 2 and Algorithm 3).

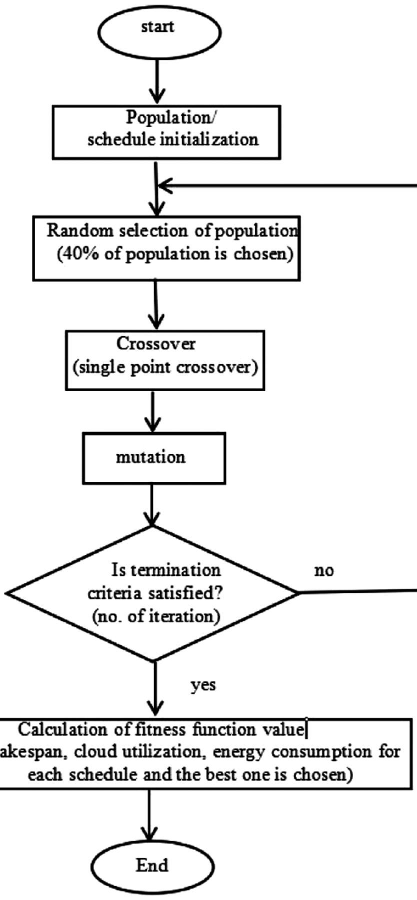

start
Population/ schedule initialization
Random selection of population (40% of population is chosen)
Crossover (single point crossover)
mutation
Is termination criteria satisfied? (no. of iteration)
no
yes
Calculation of fitness function value (makespan, cloud utilization, energy consumption for each schedule and the best one is chosen)
End

Fig. 1 — Flowchart for proposed algorithm

Algorithm 1: GA based task scheduling

```
Input:
  1. independent tasks set (n number of task)
  2. cloud or machine (m number of cloud)
  3. Expected Time to Compute (ETC) matrix
Output:
  1. Makespan
  2. Energy consumption
1. Population size= $m^n$ is randomly initialized and a set of population is randonmly chosen(40% of population size). iter= 40% of mn
2. Fitness value i.e makespan of the selected population is calculated.
3. for i=1 to iter where iter= 40% of mn
   {
     Choose two populations as parents
     Perform crossover (refer to Algorithm 2)
     Perform mutation (refer to Algorithm 3)
     Evaluate the fitness value of newly mutated offspring
     If child's f(x) < parent's f(x)
     Replace parents' schedule with child schedule
     Else no change required
   }
4. Repeat step 3 until some termination conditions are satisfied.
5. Update fitness function value, choose the best schedule, and calculate average energy consumption (Eq. 2)
```

Algorithm 2: Crossover

```
1. Let parent1 and parent2 be the parent solution
2. Apply single point crossover and choose a random point for partition
3. Swap the cloud between two sequence
   Demo_part= parent1_part
   Parent1_part=parent2_part
   Parent2_part=parent1_part
4. child1 and child2 created
```

Algorithm 3: Mutation

```
child1 and child2 are offsprings (from algorithm 2)
1. Randomly cloud position is mutated
   $r_1$=rand(1,m), $r_2$= rand(1,m)
2. $r_1$th position cloud is replaced with $r_2$th cloud
3. Muted child1, child2 created
```

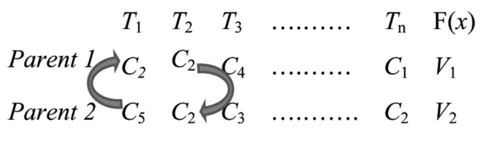
Illustration

In the selection phase, some percentage of the population is selected. In Fig. 2, $S_1, S_2, S_3.....S_n$ is the selected population. $T_1, T_2, T_3, ...T_n$ is a set of tasks and $C_1, C_2, C_3,..... C_m$ is a set of clouds or machines. The selected population in the next phase goes through crossover and a new set of the population is generated. In the crossover, randomly two parents are selected as shown in Fig. 3, and using a single-point crossover mechanism interchange takes place as depicted in Fig. 4. This results in two new children (two new sequences). Simultaneously the fitness function (F(x)) is also evaluated for the newly created child.

The procedure of creating the new chromosome (i.e. child or offspring) from two parents is known as crossover. It is GA's primary operator. Several crossover operators, such as single point crossover, partial-mapped crossover (PMX), order crossover (OX), cycle crossover (CX), position-based crossover, and so on, have been presented in recent years. Following crossing, the fitness function for the newly generated offspring (child 1, child 2) is determined, as illustrated in Fig. 5. A single point crossover occurs after the mutation. The mutation is employed to cause chromosomal disruptions in order to sustain population diversity. Inversion mutation and insertion mutation are the two primary types of mutation operators utilised in the literature. Inversion mutation is used to preserve a population's variety. Insertion mutation is utilised not just to create tiny perturbations, but also to conduct a thorough search for better progeny. Inversion mutation is used in this demonstration, and the results are presented in Fig. 6(a) and (b).

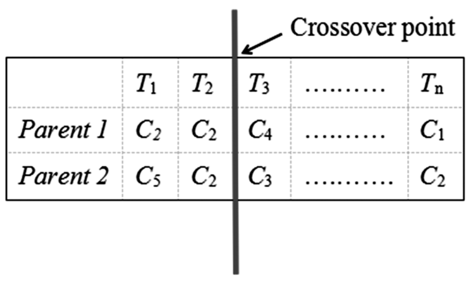
Solution/ tasks	T_1	T_2	T_3	....	T_n	F(x)
S_1	C_2	C_2	C_4	....	C_1	V_1
S_2	C_5	C_2	C_3	....	C_2	V_2
S_3	C_7	C_1	C_6	....	C_6	V_3
....				....		
S_n	C_1	C_7	C_3	....	C_4	V_n

Fig. 2 — Representation of a chromosome



	T_1	T_2	T_3	....	T_n	F(x)
Parent 1	C_2	C_2	C_4	....	C_1	V_1
Parent 2	C_5	C_2	C_3	....	C_2	V_2

Fig. 3 — chromosome set for crossover



	T_1	T_2	T_3	....	T_n
Parent 1	C_2	C_2	C_4	....	C_1
Parent 2	C_5	C_2	C_3	....	C_2

Fig. 4 — single crossover

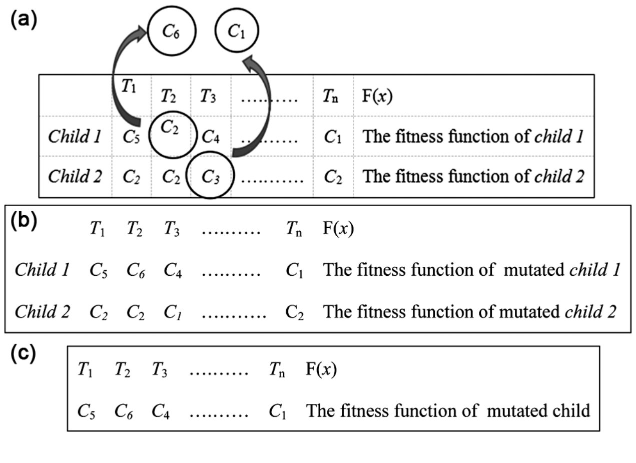
If mutated child fitness function value less than a parent, then mutated sequence is replaced with parent sequence. Otherwise, no change is required. This method is repeated for a particular number of iteration. According to the minimization function definition, a low fitness function value is selected and the schedule is noted for task allocation purposes. Let's assume that, the sequence given in Fig. 6(c) is satisfying the minimization function for fitness evaluation. In this instance, task -> cloud allocation will be T_1 ->C_5, T_2 ->C_6, T_3 ->C_4 and so on. Cloud parameters are evaluated for the scheduled task.

Experimental Evaluation and Results

To assess the criteria of performance of our proposed algorithm, synthesize dataset and benchmark dataset are taken. Overall completion time in terms of Makespan, cloud or machine utilization and amount of energy consumed for the scheduled set of tasks is evaluated. Proposed algorithm is implemented using an Intel processor (2.6 GH) using

	T_1	T_2	T_3	....	T_n	F(x)
Child 1	C_5	C_2	C_4	....	C_1	The fitness function of *child 1*
Child 2	C_2	C_2	C_3	....	C_2	The fitness function of *child 2*

Fig. 5 — child set after crossover



(a)

	T_1	T_2	T_3	....	T_n	F(x)
Child 1	C_5	C_2	C_4	....	C_1	The fitness function of *child 1*
Child 2	C_2	C_2	C_3	....	C_2	The fitness function of *child 2*

(b)

	T_1	T_2	T_3	....	T_n	F(x)
Child 1	C_5	C_6	C_4	....	C_1	The fitness function of mutated *child 1*
Child 2	C_2	C_2	C_1	....	C_2	The fitness function of mutated *child 2*

(c)

T_1	T_2	T_3	....	T_n	F(x)
C_5	C_6	C_4	....	C_1	The fitness function of mutated child

Fig. 6 — (a) Mutation on child, (b) Mutated child & (c) Generated schedule

C++ language. The dataset contains a heterogeneous task to machine execution time. The genetic algorithm concept is applied taking crossover point and mutation point and QoS parameters are evaluated. In traditional GA, multi-processing environment is one of the parameter which is considered, which is implemented through heterogeneous multi-cloud environment. These parameters are compared over 512 × 16 and 1024 × 32 datasets[29], where it represents *n* × *m* dataset for *n* number of task and *m* number of cloud or machine. Min-Min, Max-Min, and suffrage heuristics, cloud binning algorithm which are few states of methods, are compared with the proposed algorithm for each of the parameters. Simulation results obtained in terms of makespan and energy consumption value for different conventional cloud task scheduling algorithm for various benchmark dataset is given in Table 1 and Table 2. It is containing the instances like *U_C_Hi Hi, U_C_ HiLw* etc. and the makespan values for corresponding cloud task scheduling algorithms. *U, C, I* and *S* are the properties of dataset defining uniformity, consistence, inconsistence and semi-consistence respectively. *Hi* and *Lw* are task heterogeneity.[29]

Table 1 — Makespan for 1024 × 32 dataset

Instance	Min-min	Max-min	CNXM	cloud binning	GA
U_C_Hi Hi	22300000	32800000	22000000	22500000	21200000
U_C_ HiLw	2270000	3250000	2230000	2260000	2140000
U_C_Lw Hi	2200	3100	2180	2160	2030
U_C_LwLw	227	325	222	225.87	215
U_I_ Hi Hi	6290000	12100000	6990000	6370000	5910000
U_I_ HiLw	653000	1180000	647000	641000	554000
U_I_Lw Hi	685	1170	651	664.71	597
U_I_LwLw	60.5	118	59.9	63.72	58.9
U_S_ Hi Hi	14300000	22600000	14400000	14100000	13100000
U_S_ HiLw	1400000	2090000	1430000	1320000	1270000
U_S_Lw Hi	1440	2140	1390	1380	1290
U_S_LwLw	147	221	141	140.56	131

Experimental results for Makespan of 1024 × 32 datasets is shown in Fig. 7. It is showing that, GA given good result in all of the instances. It has been compared with the makespan of Min-min, Max-min, Cloud normalized scheduling and cloud binning algorithm.

Experimental results for Makespan of 512 × 16 datasets is shown in Fig. 8. It is describing that, GA performs good result in all of the instances. It has been compared with the makespan of Min-min, Max-min and Cloud k-measn scheduling technique.

Experimental results for Energy Consumption of 1024 × 32 datasets are shown in Fig. 9. It is indicating that, GA has performed good result in maximum of the instances. It has been compared with the makespan of Min-min and Max-min heuristic.

Experimental results for Energy Consumption of 512 × 16 datasets are shown in Fig. 10 which prompts that GA has performed well in maximum of the

Table 2 — Makespan for 512 × 16 dataset

Instance	Max-Min	Min-Min	CNXM	CKMS	GA
U_C_HiHi	12400000	8460000	8660000	8630000	8260000
U_C_ HiLw	204000	162000	163000	162000	158000
U_C_Lw Hi	393000	276000	284000	279000	264000
U_C_LwLw	6950	5440	5420	5400	5380
U_I_ Hi Hi	8020000	3510000	3540000	3460000	3340000
U_I_ HiLw	152000	80800	81100	80500	78100
U_I_Lw Hi	252000	121000	121000	118000	115000
U_I_LwLw	5180	2790	2790	2760	2730
U_S_ Hi Hi	9210000	5160000	5690000	5200000	4740000
U_S_ HiLw	173000	104000	105000	103000	102000
U_S_Lw Hi	282000	140000	148000	142000	135000
U_S_LwLw	6230	3810	3820	3750	3700

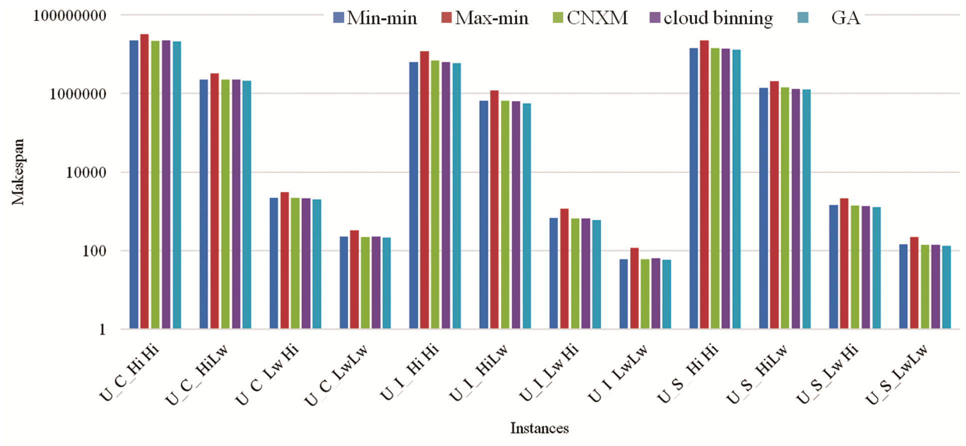


Fig. 7 — Graphical representation of Makespan (1024 × 32 dataset)

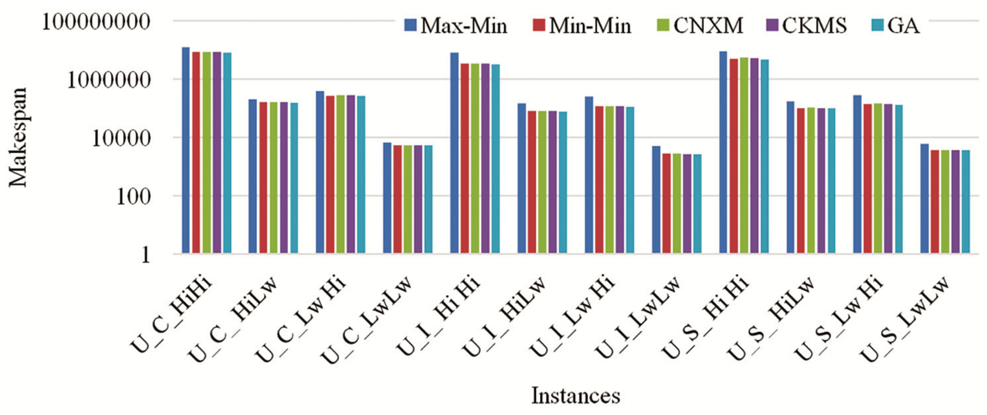


Fig. 8 — Graphical representation of Makespan (512 × 16 dataset)

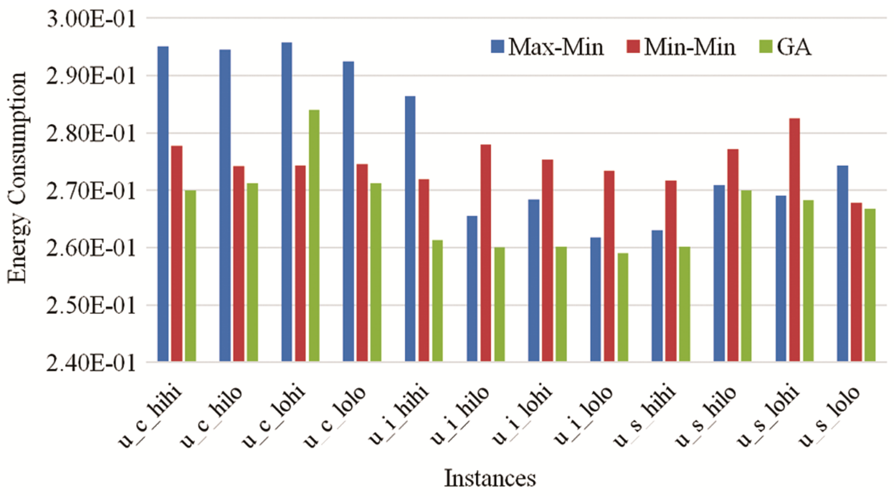


Fig. 9 — Graphical representation of Energy Consumption(1024 × 32 dataset)

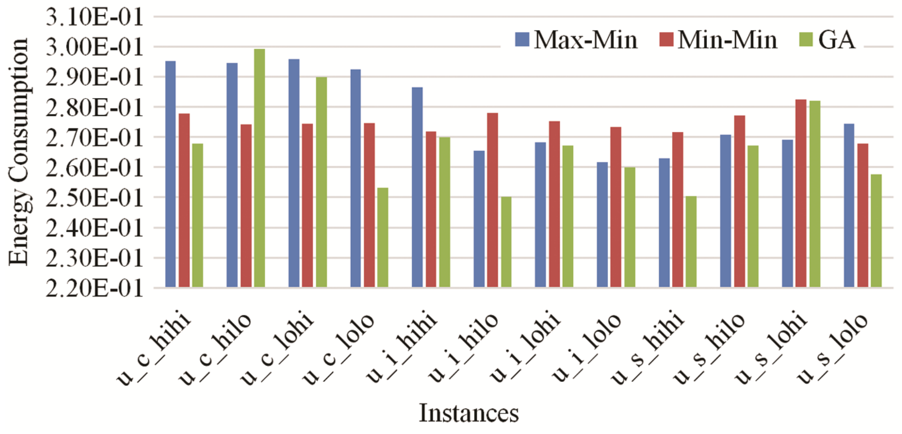


Fig. 10 — Graphical representation of Energy Consumption (512 × 16 dataset)

instances. It has been compared with the makespan of Min-min and Max-min heuristic.

A synthesized dataset has been considered by taking ETC matrix value and makespan, cloud utilization, and energy consumption for evaluation. Graphical comparison for the data input is given for the algorithms Min-min, Max-min , Suffrage heuristic and GA in Fig. 11.

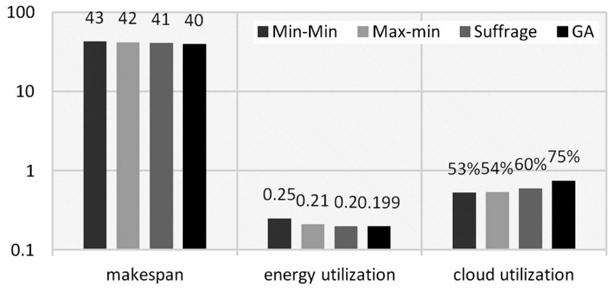


Fig. 11 — Cloud parameters comparison for synthesized dataset

Conclusions

In the modern cloud computing paradigm, task scheduling and energy usage are two major concerns. This study has presented a task sheduling algorithm for heterogeneous multi-cloud system. The algorithm has been presented based on the principle of GA, which is one of the popular meta-heuristics techniques. Here makespan is taken as a parameter to find the optimized schedule and corresponding cloud utilization for energy consumption evaluation. Experimental analysis shows that it has outperformed the existing algorithms like Min-Min, Max-Min, sufferage, CNXM and cloud binning methods in different instances. Both synthesized and benchmark datasets are taken for experimental analysis purposes. Heterogeneous tasks and a multi-cloud environment provide a good solution with a genetic algorithm that follows the evolutionary techniques. This algorithm is based on single objective optimization parameter. However, multi objective optimization techniques can also be implemented to find out optimal result.

References

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

2 Li J, Qiu M, Ming Z, Quan G, Qin X & Gu, Z, Online optimization for scheduling preemptable tasks on IaaS cloud systems, *J Parallel Distrib Comput*, **72** (2012) 666–677.

3 Panda S K, Pradhan R, Neha B & Sathua S K, Fairness-Aware Task Allocation for Heterogeneous Multi-Cloud Systems in *Advanced Research on Cloud Computing Design and Applications,* edited by Shadi, (Hershey, PA: IGI Global), 2015, 147–170.

4 Pradhan R & Dash A K , A novel task scheduling algorithm in heterogeneous cloud environment using equi-depth binning method, in *Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing* edited by Management Association, (IGI Global) 2021, 1303–1316.

5 Ibarra O H and Kim C E, Heuristic algorithms for scheduling independent tasks on nonidentical processors, *J ACM*, **24**, (1977), 280–289, https://doi.org/10.1145/322003.322011.

6 Pradhan R, Panda S K, & Sathua S K, K-means Min-Min scheduling algorithm for heterogeneous grids or clouds, *Int J Inf Process*, **9** (2015) 89–99.

7 Panda S K & Jana P, An efficient task scheduling algorithm for heterogeneous multicloud environment, in *3rd IEEE Int Conf Adv Comput*, *Commun Informat.*

8 Pradhan R & Satapathy S C, Task scheduling in heterogeneous cloud environment—A, *ICICC 2019*, *Advances in Intelligent Systems and Computing* (Springer, Singapore) 2020, 1034.

9 Mahmood A, A hybrid genetic algorithm for task scheduling in multiprocessor real-time systems, 2000, http://www.ici.ro/ici/revista/sic2000-3/art05.html

10 Zhao C, Zhang S, Liu Q, Xie J & Hu J, Independent tasks scheduling based on genetic algorithm in cloud computing, *5th Int Conf Wire Commun, Netw Mobile Comput*, 2009, 1–4.

11 Burke E K, Elliman D G & Weare R F , A genetic algorithm for university timetabling, *AISB Workshop Evolut Comput*, *Leeds,* 1994 .

12 Lee Y & Zomaya A, Energy efficient utilization of resources in cloud computing systems, *J Supercomput*, **60** (2010) 268–280.

13 Kaabouch N & Wen C H, Energy-aware systems and networking for sustainable initiatives in *Hershey*, (IGI Global), 2012.

14 Koomey J G , Estimating total power consumption by servers in the U.S. and the world, *Lawrence Berkeley National Laboratory*, *Stanford University,* (2007)

15 Barroso L & Holzle U, The case for energy-proportional computing, *Computer*, **40** (2007) 33–37.

16 Bohrer P, Elnozahy E N, Keller T, Kistler M, Lefurgy C & McDowell C, The case for power management in web servers, *Power Aware Comput*, 2002, 261-289, DOI: 10.1007/978-1-4757-6217-4_14.

17 Fan X, Weber W & Barroso L, Power provisioning for a warehouse-sized computer, *ACM SIGARCH Computer Architecture News*, **35** (2007) 13–23.

18 Koomey J, Worldwide electricity used in data centers, *Environ Res Lett*, **3**(2008).

19 Meisner D, Gold B & Wenisch T, Power Nap, *ACM SIGARCH Computer Architecture News*,**37** (2009) 205–216.

20 Grajcar M, Genetic list scheduling algorithm for scheduling and allocation on a loosely coupled heterogeneous multiprocessor system, *Proc Design Automat Conf IEEE*, 1999, 280–285.

21 Esquivel S C, Gatica C R & Gallard R H, A genetic approach using direct representation of solutions for the parallel task scheduling problem, *Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology),* 2000.

22 Nossal R, An evolutionary approach to multiprocessor scheduling of dependent tasks, *Future Gener Comput Syst*, **14** (1998) 383–92.

23 Auyeung A, Gondra I, Dai H K, Multi-heuristic list scheduling genetic algorithm for task scheduling, *Proc 2003 ACM Symp Appl Comput*, 721–724.

24 www.plugloadsolutions.com/80PlusPowerSupplies.aspx (01 March 2019)

25 Lee Y & Zomaya A, Energy efficient utilization of resources in cloud computing systems, *J Supercomput*, **60** (2010) 268–280.

26 Hsu C, Chen S, Lee C, Chang H, Lai K, Li K & Rong C, Energy-aware task consolidation technique for cloud computing, *2011 IEEE 3rd Int Conf Cloud Comput Technol Sci*, (2011), 115–121.

27 Kessaci Y, Mezmaz M, Melab N, Talbi E G & Tuyttens D, Parallel evolutionary algorithms for energy aware scheduling, in *Intelligent Decision Systems in Large-Scale Distributed Environments* edited by P Bouvry, H González-Vélez & J Kołodziej (Springer, Berlin, Heidelberg) 2011, 75–100.

28 Luo L , Wu W J & Zhang F, Energy modeling based on cloud data center, *J Softw*, **25** (2014) 1371–1387

29 Braun T D, Siegel H J, Beck N, Boloni L L, Maheswaran M, Reuther A I, Robertson J P, Theys D M, Yao B, Hensgen D & Freund F R, A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems, *J Parallel Distrib Comput*, **61** (2001) 810–837.