An Optimization of Makespan, Energy Consumption, and Load Balancing on The Task Scheduling in Cloud Computing using Particle Swarm Optimization (PSO)

Fajar Kusumaningayu, Antoni Wibowo

Abstract*: Cloud computing is widely used resource sharing computational technology to provide fast, reliable, and scalable computational process for organizations and companies without the need to build and maintain their own server. The research area about cloud computing is dynamic and versatile. One may have concern on the privacy, security, networking, optimization, etc. Due to huge demand for cloud computing, it creates several problems such as makespan, energy consumption, and load balancing. Task scheduling is one of the technologies that have been applied to solve those objectivities. However, task scheduling is one of the well-known NP-hard problems, and it is difficult to find the optimum solution. In order to solve this problem, previous studies have utilized meta-heuristic method to find the best solution based on the solution spaces. This study proposed Particle Swarm Optimization (PSO) to solve the multi-objective task scheduling to achieve the optimum solution. The effectiveness of the proposed algorithm will be compared with Genetic Algorithm (GA), Clonal Selection Algorithm (CSA), and Bat Algorithm (BA). This study converts three objectivities into single objectivity optimization with each objectivity act as variable assigned with weight that present its priority and has implemented those meta-heuristics. The simulation result from ten data set shows that PSO able to outperform GA, CSA, and BA especially for makespan and energy consumption without the cost of algorithm duration since PSO has fast convergence rate compare to the other three algorithms and making it a good choice for dynamic task scheduling in data center cloud computing where the algorithm duration is one of important factor

Keywords* : Cloud Computing, Multi-Objectivities, Particle Swarm Optimization, Task Scheduling.

Manuscript published on November 30, 2019.
* Correspondence Author

Fajar Kusumaningayu*, Binus Graduate Program - Master of Computer Science Bina Nusantara University , Anggrek Campus Jl. Kebon Jeruk Raya No. 27, Kebon Jeruk, West Jakarta 11480 Indonesia (phone: +62-21-53696969 ext 1803; e-mail:fajar.kusumaningayu@binus.ac.id[1] and anwibowo@binus.edu[2]).

Antoni Wibowo, Binus Graduate Program - Master of Computer Science Bina Nusantara University , Anggrek Campus Jl. Kebon Jeruk Raya No. 27, Kebon Jeruk, West Jakarta 11480 Indonesia (phone: +62-21-53696969 ext 1803; e-mail:fajar.kusumaningayu@binus.ac.id[1] and anwibowo@binus.edu[2]).

I. INTRODUCTION

The high demand of clients to the cloud required the utilization of capable data center to manage those workloads. It has been predicted that on 2021 94% of workloads will be handled by the cloud data center and the rest use traditional data center [1]. Cloud computing spend high energy consumption, in 2016 for 289 data centers in Europe they reached 3,735,735 MWh as total energy consumption [2]. Thus, it is inevitable for the data center to explode in power consumption and in terms of the number to meet high demand from users. This causes a rising concern on the environment, since 66.8% of electricity in the world in 2017 is powered by coal, gas, and oil [3], and encourages the community to embrace green cloud computing technology.

In order to develop a green cloud computing data center, several techniques have been proposed. First, the task scheduling algorithm with the objectivity to minimize the energy consumption using task queue [4]. Second, Virtual Machine (VM) migration to find the optimum VM placement and relocation [5]. Third, resource sharing and location where several hosts in the data center may share and access the same physical machines [6]. Forth, find the best location to serve the clients and produce the most energy-efficient [7]. Fifth, using renewable power supply where it uses another power resource apart from coal [8]. Last but not least, the applied software development process included green aspects in every step [9]. However, this study will cover only one of the method which is task scheduling optimization in the data center.

Users send several computational jobs or tasks to the data center to be executed. The data center will collect those tasks and create a scheduling process. Tasks scheduling will be assigned the task to a certain resource in the data center based on the characteristics and requirements of the tasks. Therefore, the tasks scheduling process holds an important role in order to give efficient services to users[10]. Even though energy consumption is an important aspect however one cannot ignore the makespan and the load balancing of each resource.

This study defines makespan as the time required to finish all the scheduled tasks, energy consumption is total energy used by the VM to execute all the tasks, and the load balancing will contain the variance of tasks assigned in one VM so that it can reach standard deviation near to zero.

This study will utilize fitness function where it will represent makespan, energy consumption, and load balancing variance. Several constraints of this study are the size of every task should be more than zero, execution time required to finish each task will more than zero, energy consumption in each VM will more than zero since no VM will be powered off, all the tasks must be scheduled, and one task only scheduled in one VM and executed only once. Each of those objectives will be fuss into a single objective function with constantan to represent the priority of each objective. This study will put equal important for three aspects.

Then, propose a Particle Swarm Algorithm (PSO) to reduce makespan, energy usage, and balanced load in the cloud computing data center as the fitness function to determine whether or not the proposed algorithm has reached an optimum solution. In order to evaluate the effectiveness of the proposed algorithm, this study will compare the proposed algorithm with the Genetic Algorithm (GA), Clonal Selection Algorithm (CSA), and Bat Algorithm (BA).

Several problems will be addressed in this study are mathematical model, implementation and evaluation of GA, PSO, CSA, and BA. In order to find the task scheduling mathematical model for a data center for reducing the energy consumption and makespan with the balanced load to each of the VM, find the algorithms to solve the NP-hard problem such as task scheduling, and simulate the task scheduling using GA, PSO, CSA, and BA, and compare their makespan, energy, and load balancing variance from the fitness function. The study is implemented in fitness simulation, and all the VMs are already built before the task scheduling. This study contribution is finding the best algorithm for three objectivities which are reducing the makespan, energy consumption, and load balancing variance in four algorithms (GA, PSO, CSA, and BA). To the best of authors' knowledge the four algorithms have not been implemented to solve those three objectivities.

The rest of the study will construct as followed, the first background, the second section is related work, the third section is methodology, fourth mathematical model and simulation parameters, fifth is result and discussion, and last but not least is a conclusion and future works..

II. RELATED WORKS

A. Task Scheduling with Makespan Optimization

Dividing the task based on their level and consider the availability of the resources [10], treat the task scheduling and file assignment at the same time to reduce unnecessary steps while executing that procedure twice [11], able to reduce the makespan. Other strategies such as chopping the large task size into a smaller group of tasks[12], and using chaotic randomness help the best result from local optimum [13]. The constraint is checked during the fitness function, but it can be avoided by a direct implementation during solution production and it can help to reduce the makespan [14]. Makespan can be used as a constraint for other objectivities such as optimizing the budget cost for each of the customers [15]. The previous study uses the result of the greedy algorithm as initial solutions for PSO and able to reduce the scheduling time by ten seconds [16]. The same method applied by another study where the result from PSO will be used by Hill Climbing [17].

B. Task Scheduling with Energy Optimization

There are a lot of studies shows their concern on the energy optimization of the data center. Some of them using task scheduling algorithms to achieve that goal and proposed policy in the data center. The data center should apply one or several policies to optimize energy consumption [18].

Table- I: Summary of Related Works

No	Author	Approach	Input	Objective
1	[12]	PSO	Tasks	Makespan
2	[13]	Chaotic symbiotic organisms search	Tasks	Makespan and Cost
3	[10]	Intelligence Water Drop	integer: time and cost of the task	decrease the task execution time
4	[17]	Hybrid PSO and HC.	Directed Acyclic Graph (DAG)	decrease the makespan
5	[19]	GWO	Task and resource	decrease the makespan and energy optimization
6	[20]	A hybrid of GA and ILP	Resources, storage, tasks	Minimize energy usage
7	[11]	Hybrid Evolutionary Algorithm	execution time and shared resources	decrease the makespan
8	[21]	CSA map the resource and tasks	Task and resource	decrease the makespan and energy optimization
9	[22]	Stochastic-HC	Tasks and VM	decrease the energy usage
10	[23]	Multiple-Workflows-Slack-Time-Reclaiming	DAG	decrease the makespan and energy optimization
11	[24]	Non-DVFS and global DVFS	DAG	Energy optimization
12	[25]	GA	Tasks	Makespan and energy optimization
13	[16]	Hybrid of greedy and PSO	integer: the time required to execute the task	Reduce execution time, and resources optimization
14	[26]	The BA with a budget constraint	task execution time, task cost, VM reliability, budget	Optimization of cost, execution time, and reliability
15	[15]	ACO	CPU, task, and budget cost	Faster computation within budget cost
16	[27]	Greedy	Tasks	Makespan
17	[14]	GA	Tasks	Makespan
18	[28]	Greedy	Tasks and resources with dynamic voltage scaling	Makespan and energy consumption

The energy used in task caused by the frequency being applied thus utilize Dynamic voltage and frequency scaling (DVFS) policies to an assigned suitable frequency for each task will help to reduce the energy consumption[8]. Then, the migration of tasks to resources with minimum energy usage is able to decrease global energy consumption [24].

One of the methods by optimizing VM energy usage [22], without compromise several boundaries such as budget, cost, and execution time [26]. Greedy algorithm [27], and Hybrid algorithm which is integer linear programming (ILP) and GA [20] are proposed to solve the energy consumption optimization.

C. Task Scheduling with Makespan and Energy Optimization

By optimizing the makespan will affect energy consumption reduction [25]. The previous study using CSA to optimized the makespan and energy consumption during the execution of the task [21]. Resources do not only required energy during task execution but they also need the energy to maintain its idle status [23]. The greedy algorithm has been implemented to solve these objectivities with deadline constraints [28]. Another algorithm such as the grey wolf has become an alternative for solving task scheduling to achieve faster makespan and energy reduction. In order to have better performance than the original grey wolf, there is some modification on the encircling and hunting equation of the original algorithm [19].

D. Summary

Table 1 contains the list of summaries from the previous works. This study has highlight four promising algorithms to solve task scheduling problem which are GA, PSO, CSA, and BA, that has big potential to satisfy the task scheduling for the data center to optimize the makespan, energy consumption, and load balancing. To the best of authors' knowledge, the previous studies have not tried to find the best single meta-heuristic algorithm to solve an optimizing makespan, energy consumption, and load balancing.

Table- II: Experimental Settings[19]

Parameters	Value
Number of data center	5
Number of hosts	10
Number of VM	50
VM MIPS	[500-2500]MIPS
VM core	[1-5]
Number of tasks	[100-1000]
Task Instruction Length	[200-15000]MIPS

III. METHODOLOGY

This study workflow is presented in Fig. 1, it starts from the mathematical model, algorithm implementation, evaluation for each of parameters, and reporting.

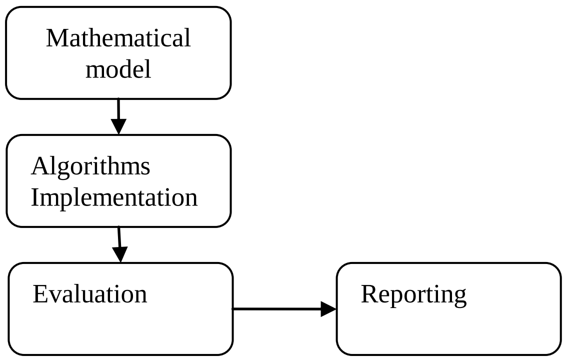

Fig 1. Research Workflow

A. Mathematical Model

Three objectivities of this study are makespan (MS), energy consumption (Etot), and load balancing (LB). The decision variable for MS is C_{ij}, where C_{ij} defined as the computational time required by ith VM to execute all tasks (jth) that assigned to ith VM. Etotal will be influenced by the Ec and Eidle where Ec is energy used by ith VM to execute jth tasks, and Eidle defined as the energy used by ith VM to maintain their idle condition. Then, for LB the decision will determine by the variance of tasks assigned across all the VMs.

B. Algorithms Implementation

The simulation has been conducted on Java Netbean 8.2 based on Fig. 1 for resource information. The detail of the experimental setting is listed in Table 2 and the detail of algorithms parameters are listed in Table 3.

C. Evaluation

This study will implement four algorithms which are PSO, GA, CSA, and BA. The testing process will be repeated for twenty times to determine each algorithm best, average, and worst result. The parameters that will be evaluated are makespan, energy consumption, load balancing, and algorithm execution time.

Table- III: Algorithm Settings

Parameters	Value
Number of testing	20
Number of iteration	50
GA parent	2 chromosomes
crossover	Half point
Mutation type	Swap mutation
CSA number of cloning and the number of multiplication	[3-10]
CSA constantan cloning and n best constantan	0.1
PSO weight	1
PSO p1, p2	0.8
BA frequency max	10
BA frequency min	0
BA Amplitude	1
Fitness α, β, γ	1/3

IV. PROPOSED ALGORITHM

PSO is one of a meta-heuristic algorithm that mimics the flock of bird's movement in search of food. The birds fly in any direction thus who found food will send a signal to others, the more food is being founded in a certain area the cry is growing louder to lead other birds to find the global or local optimum position. The bird's position and speed are changeable every iteration [29]. Fig.2 shows the flowchart of PSO.

The mathematical model for the next velocity equation and position are:

$$v_{n\_bird}[t+1] = wv_{n\_bird}[t] + \sum_{i=1}^{2} \rho_i r_i \left(X_{i_{n\_bird}}[t] - x_{n\_bird}[t] \right) \quad (1)$$

$$x[t+1] = x_{n\_bird}[t] + v_{n\_bird}[t+1] \quad (2)$$

Where $x_{n\_bird}[t]$ is the position of bird, $v_{n\_bird}[t]$ is the speed of bird, w,ρ_1,$\rho_2$are coefficient assigned weight, r_1, r_2 are random vectors, X_{1n_bird} [t] is local optimum solution and X_{2n_bird} [t] is the global optimum solution.

Another method by dividing the particles inside each of the swarm is divided into two types which are a local and global particle, this method called MSMOOA.
It uses the local swarm to run as the rule and decision of their own swarm but the global swarm will update their movement based on the global decision as part of their sharing information strategies[30].

PSO does not show satisfying optimization results for handling a large number of tasks. Split the tasks into several sub-parts, the PSO algorithm will be used to optimize those tasks within each task and after it finishes combine it into one and do resource adjustment based on VM loads. This method is able to reduce the makespan up to 50% for task scheduling cloud computing [12].

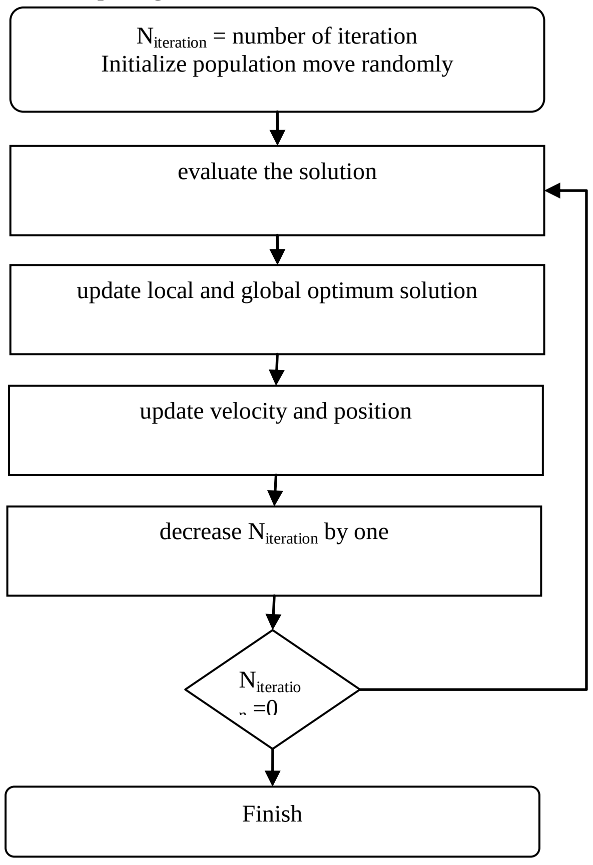
N_iteration = number of iteration
Initialize population move randomly
evaluate the solution
update local and global optimum solution
update velocity and position
decrease N_iteration by one
N_iteration =0
Finish

Fig. 2. PSO Flowchart

V. MATHEMATICAL MODEL

This session discusses the objective function of this study and the termination condition. Several constraints applied in this system are all the tasks register by the user should be scheduled does not matter which VM, for each of the task can only be executed once and only in one single VM. This condition is being applied during the solution generator. Therefore, there is no checking on a fitness function.

This study assumes that each task is independent of the other task and should be computed in a single VM. The breakdown of each task called subtasks can only be done inside the assigned VM and being distributed among the available core. Therefore, the finish tasks in one VM consider having full utilization of the core inside the assigned VM. Fig. 7 shows the structure of five data center with two hosts in each data center and register VM inside of hosts.

A. Makespan

This study defines makespan as the total execution time of all the tasks. The tasks are executed in VM for each of the finished tasks, the next task in the queue will be executed thus there is no delay time in the queue process and there is no waiting time to be calculated. Moreover, each task is independence from each other, therefore the task can be run at the same time without predecessor tasks and one VM only handles one task at the same time but more than one task can be scheduled to one VM. The decision variable for the makespan function is Cij as computation time Cij≥0, while makespan is the time required for all the tasks to be executed [13,19]. By calculating the maximum time required by the VM which runs at the same time, then it will represent the overall time the tasks will finish.

$$MS = \max\{C_{ij}, \forall i, \forall j\} \quad (3)$$

Where MS is makespan, Cij is computation time to solve jth as all tasks assigned to ith VM.

B. Energy Consumption

Assume that the power used by the host to stay up will be equal to the VM register inside it. Furthermore, the power consumption information used during idle will be count as 50% of peak power, this assumption based on the claim of the previous study that during idle CPU still used power consumption [31].

The previous study stated that in computing compared to the energy used in another sector, most of the energy in the computer is used to power up the CPU [32]. Therefore, calculating the energy usage computing process can be represented by CPU energy usage. The previous study counts the energy consumption based on the energy used in task execution, therefore when the VM is idle, it is killed directly [21]. This study will count the energy used during the VM idle time [23]. The mathematical model for power idle can be eliminated depending on the policy applied in the data center, for the type of datacenter who turns off the VM and host when it no longer in services it can be removed.

The decision variables for energy consumption are Ec and Eidle.

$$E_c = C_{ij} \times P_i \quad (4)$$

$$E_{idle} = (MS - C_{ij}) \times (P_{i_{idle}}) \quad (5)$$

$$E_{total} = C_{ij} \times P_i + (MS - C_{ij}) \times (P_{i_{idle}}); \forall i, \forall j \quad (6)$$

$$E_{total} = E_c + E_{idle} \quad (7)$$

Where E total is the sum of energy consumption of energy required by VM to execute the tasks and energy used to maintain their idle, $P_{i_{idle}}$ is the energy used by VMi during idle, Pi is energy peak from VM core, MS is the makespan, Cij is the computation time in ith VM to execute jth tasks

C. Load Balancing

The goal of load balancing is to have every task distributed equally across the existing resources.

Assume that all tasks are equally distributed then using variance formula should equal zero, therefore lowering the variance result mean that the tasks are closer to be equally distributed. The variance function used in the study is.

$$LB = \frac{\sum_{i=1}^{m}\left(Task_{ij} - \overline{Task}\right)^2}{m} \quad \{1,2,3...Task_{ij}, i \in m\},\ j \in \{1,2,3...,n\} \quad (8)$$

Where LB is the variance of load balancing, jth is the total instruction length of all the tasks assigned to ith VM.

D. Fitness Function

Reduce the makespan, energy consumption, and load balancing using task scheduling approach is the main goal of this study. Therefore, function addressing these objectives need to be delivered. One may found the other to be more important than the other aspect. Therefore, value of α, β, and γ is used to determine the priority of said objective in the fitness function. The same strategies has been implemented for makespan and energy consumption [19].

$$\min F = \alpha \times F^1 + \beta \times F^2 + \gamma \times F^3 \text{ and } \alpha + \beta + \gamma = 1 \quad (9)$$

Where MS is makespan, Etotal is energy total, and LB is load balancing.

VI. RESULT AND DISCUSSION

The simulation has been conducted on Java Netbeans 8.2. The cloud environment condition refers to the structure design in Fig.7 with ten data consist of 100-1000 tasks that generated randomly. The algorithms being simulated are the proposed algorithm (PSO), Genetic Algorithm (GA), Clonal Search Algorithm (CSA), and Bat Algorithm (BA). There are four parameters being considered in this study such as energy consumption, makespan, load balancing violation, and the algorithm running time to achieve the solution. The detail of the result can be seen in Table 4.

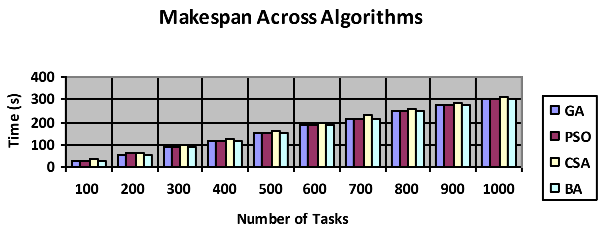

Fig. 3. Makespan Across Algorithms Group by Number of Tasks

A. Result

1. Makespan: Fig. 3 shows the comparison of makespan of each algorithm group by the number of tasks, with small differences among the algorithm but quite noticeable compared to energy consumption. In the makespan, PSO shows better performance than the other algorithms. The makespan and energy consumption shows a pair of the compatible result, both objectives shows identical behavior. It is understandable since energy consumption is highly influenced by the makespan, both objectivities give an almost similar result with a slight difference since the difference between makespan is much distinguish compare to energy consumption.

2. Energy Consumption: Fig. 4 shows the energy consumption of the solution offered by the algorithms group by the number of tasks being scheduled. From the graph, the differences between each algorithm are not noticeable. GA, PSO, and BA show competitive results among each other, in several occurrences such as 100 tasks and 700 tasks BA able to yield the best energy minimization, while GA wins for tasks 200. The rest of the data PSO shows a better result than the other algorithms.

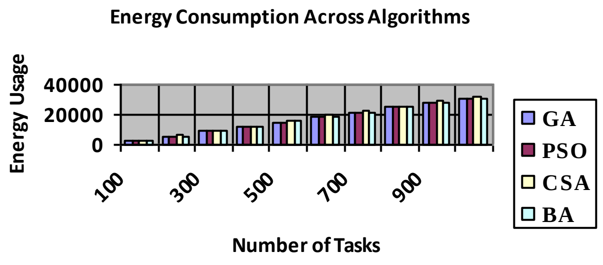

Fig. 4. Energy Consumption Across Algorithms Group by Number of Tasks

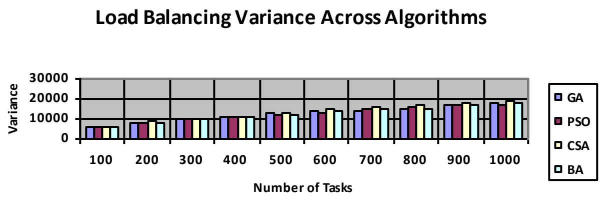

Fig. 5. Load Balancing Variance Across Algorithms Group by Number of Tasks

3. Load Balancing: The load balancing results are quire differences comparing to makespan and energy consumption. While one can see the pattern for makespan and energy consumption, but in case of load balancing violation, the best load balancing result is being spread out. Take a look at GA able to yield the best outcome on tasks with size 300, 400, 700, and 800. Then for BA able to achieve minimum load balancing for tasks size 100, 200, 500, and 900, and PSO gives its best shoot for tasks size 600 and 1000. The detail of load balancing can be seen in Fig. 5.

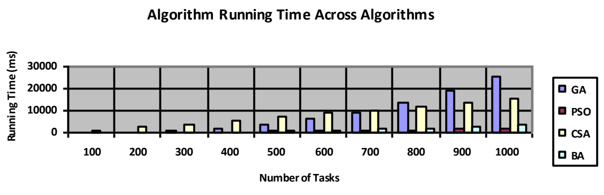

Fig. 6. Algorithm Running Time Across Algorithms Group by Number of Task

4. Running Time: Fig. 6 shows the time required to derive the best solution of tasks sequenced for each of the algorithms.

The result shows that GA required the largest amount of time to solve the task scheduling problem, followed by CSA. While surprisingly even though BA is well known for have a fast convergent rate, however for tasks size larger than 300, PSO shows faster computation time.

B. Discussion

From the result on previous section, some information can be derived. In the PSO the number of velocity and position is being used to determine the number of swap operation need to be done for certain solution space to achieve the best result.

In this simulation, PSO shows outstanding performance compare to the other algorithms in almost all of energy consumption and makespan, PSO able to achieve the best result with quite small running time especially in the larger number of task scheduled the increasing running time is relatively small considering the increasing length of data.

In the Genetic Algorithm (GA) the study used half point cross over with swap operation as the mutation procedure. GA does give fast running time in their early testing for small number of tasks however when dealing with quite large number of tasks to be scheduled, they required longer time to finish the algorithm. In the couple of their based performance GA capable to shows best result for couple of times in term of energy, makespan, and load balancing violation.

During the simulation, CSA required a big amount of memory space since it applied the cloning process, therefore, this study limit cloning from 3 to 10 to avoid the big size of memory required to perform the CSA scheduling process. Even though, CSA does not give the close range to best result. However, in term of running time parameter, it is important to note that CSA required largest computation time compared to other algorithms in smaller tasks size, however, due to the limitation of cloning multiplication, the computation cost in larger tasks size is quite stable supported by the fact that CSA is able to beat GA.

Bat Algorithm (BA) does a wonderful job in terms of energy consumption and makespan in a couple of tasks where the PSO does not yield the best result. BA is well known to have a fast convergence rate compare to the other algorithms. Thus it comes as a surprise that it can only keep up the reputation for smaller tasks size, for larger tasks size PSO able to yield the minimum computation time.

VII. CONCLUSION

The metaheuristic algorithms have been implemented to solve optimization in NP-hard problems such as task scheduling whether it is for computing, manufacture, job, and employee scheduling. The review from the previous studies highlights four promising algorithms such as GA, PSO, CSA, and BA. Furthermore, recent studies show interest in task scheduling for multi-objectivities.

This study aims to minimize the makespan, energy consumption, and load balancing for task scheduling data center in cloud computing. The mathematical model to measure the quality of solution spaces has been proposed by converting three objectivities into one single objective. This study has simulated the GA, PSO, CSA, and BA with the result derived from fitness function, and PSO shows better performance compare to GA, CSA, and BA.

Overall, PSO displays good performance for optimizing the makespan, energy consumption, and load balancing. In addition, PSO and BA are suitable to solve dynamic task scheduling since both of the algorithms show a fast running time. Therefore, it will decrease the delaying time for the tasks to be scheduled. BA will perform faster in smaller tasks size while in larger tasks size PSO is more suitable. Moreover, in case where duration is one of important parameter then PSO and BA are promising algorithms for a hybrid selection.

REFERENCES

1. Cisco, "Cisco Global Cloud Index: Forecast and Methodology, 2016–2021 White Paper," 2018. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html.
2. M. Avgerinou, P. Bertoldi, and L. Castellazzi, "Trends in Data Centre Energy Consumption under the Energy Efficiency," Energies, vol. 10, no. 1470, pp. 1–18, 2017.
3. IAE, "Electricity Statistics," 2017. [Online]. Available: https://www.iea.org/statistics/electricity/. [Accessed: 03-Oct-2019].
4. M. Guzek, D. Kliazovich, and S. Member, "Minimum Dependencies Energy-Efficient Scheduling in Data Centers," IEEE Trans. Parallel Distrib. Syst., vol. 27, no. 12, pp. 3561–3574, 2016.
5. V. R. Reguri, S. Kogatam, and M. Moh, "Energy Efficient Traffic-Aware Virtual Machine Migration in Green Cloud Data Centers," in 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016, pp. 268–273.
6. M. Karuppasamy, S. Suprakash, and S. P. Balakannan, "Energy-Aware Resource Allocation for an Unceasing Green Cloud Environment," in 2017 International Conference on Intelligent Computing and Control (I2C2), 2017.
7. F. Larumbe and B. Sanso, "Green Cloud Broker: On-line Dynamic Virtual Machine Placement Across Multiple Cloud Providers," in Proceedings - 2016 5th IEEE International Conference on Cloud Networking, CloudNet 2016, 2016, pp. 119–125.
8. C. Wang, M. Zink, and D. Irwin, "Optimizing parallel HPC applications for green energy sources," in 2015 6th International Green and Sustainable Computing Conference, 2016, pp. 1–8.
9. S. K. Sharma, P. K. Gupta, and R. Malekian, "Energy efficient software development life cycle - An approach towards smart computing," in 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), 2015, pp. 1–5.
10. S. Elsherbiny, E. Eldaydamony, M. Alrahmawy, and A. E. Reyad, "An extended Intelligent Water Drops algorithm for workflow scheduling in cloud computing environment," Egypt. Informatics J., vol. 19, pp. 33–55, 2018.
11. L. Teylo, U. de Paula, Y. Frota, D. de Oliveira, and L. M. M. A. Drummond, "A hybrid evolutionary algorithm for task scheduling and data assignment of data-intensive scientific workflows on clouds," Futur. Gener. Comput. Syst., vol. 76, pp. 1–17, 2017.
12. H. Saleh, H. Nashaat, W. Saber, and H. M. Harb, "IPSO Task Scheduling Algorithm for Large Scale Data in Cloud Computing Environment," IEEE Access, vol. 7, pp. 5412–5420, 2019.
13. M. Abdullahi, M. A. Ngadi, S. I. Dishing, S. M. Abdulhamid, and B. I. Ahmad, "An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment," J. Netw. Comput. Appl., vol. 133, no. 1 May 2019, pp. 60–74, 2019.
14. Y. Xu, K. Li, J. Hu, and K. Li, "A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues," Inf. Sci. (Ny)., vol. 270, pp. 255–287, 2014.
15. L. Zuo, L. Shu, S. Dong, C. Zhu, and T. Hara, "A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing," IEEE Access, vol. 3, pp. 2687–2699, 2015.
16. Z. Zhong, K. Chen, X. Zhai, and S. Zhou, "Virtual machine-based task scheduling algorithm in a cloud computing environment," Tsinghua Sci. Technol., vol. 21, no. 6, pp. 660–667, 2016.

17. N. Dordaie and N. J. Navimipour, “A hybrid particle swarm optimization and hill climbing algorithm for task scheduling in the cloud environments,” ICT Express, vol. 4, no. 4, pp. 199–202, 2018.
18. M. Bala and Devanand, “Performance Evaluation of Cloud Datacenters Using Various Green Computing Tactics,” in 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015, pp. 956–961.
19. G. Natesan and A. Chokkalingam, “Task scheduling in heterogeneous cloud environment using mean grey wolf optimization algorithm,” ICT Express, vol. 5, 2018.
20. H. Ibrahim, R. O. Aburukba, and K. El-Fakih, “An Integer Linear Programming model and Adaptive Genetic Algorithm approach to minimize energy consumption of Cloud computing data centers,” Comput. Electr. Eng., vol. 67, pp. 551–565, 2018.
21. R. K. Jena, “Energy Efficient Task Scheduling in Cloud Environment,” Energy Procedia, vol. 141, pp. 222–227, 2017.
22. S. Rashmi and A. Basu, “Resource optimised workflow scheduling in Hadoop using stochastic hill climbing technique,” IET Softw., vol. 11, no. 5, pp. 239–244, 2017.
23. J. Jiang, Y. Lin, G. Xie, and L. Fu, “Time and Energy Optimization Algorithms for the Static Scheduling of Multiple Workflows in Heterogeneous Computing System,” J Grid Comput., vol. 15, no. 4, pp. 435–456, 2017.
24. G. Xie, G. Zeng, X. Xiao, R. Li, and S. Member, “Energy-efficient Scheduling Algorithms for Real-time Parallel Applications on Heterogeneous Distributed Embedded Systems,” IEEE Trans. Parallel Distrib. Syst., vol. 28, no. 12, pp. 3426–3442, 2017.
25. Y. Shen, Z. Bao, X. Qin, and J. Shen, “Adaptive task scheduling strategy in cloud : when energy consumption meets performance guarantee,” World Wide Web, vol. 20, no. 2, pp. 155–173, 2017.
26. N. Kaur and S. Singh, “A Budget-constrained Time and Reliability Optimization BAT Algorithm for Scheduling Workflow Applications in Clouds,” Procedia Comput. Sci., vol. 98, pp. 199–204, 2016.
27. Z. Dong, N. Liu, and R. Rojas-cessa, “Greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers,” J. Cloud Comput., vol. 8, no. 8, pp. 1–14, 2015.
28. P. Lindberg, J. Leingang, D. Lysaker, S. U. Khan, and J. Li, “Comparison and analysis of eight scheduling heuristics for the optimization of energy consumption and makespan in large-scale distributed systems,” J Supercomput, vol. 59, no. 1, pp. 323–360, 2012.
29. M. Couceiro and P. Ghamisi, Fractional Order Darwinian Particle Swarm Optimization: Applications and Evaluation of an Evolutionary Algorithm. Springer, 2016.
30. Y. A. O. Guang-shun, D. Yong-sheng, and H. A. O. Kuang-rong, “Multi-objective workflow scheduling in cloud system based on cooperative multi-swarm optimization algorithm,” J. Cent. South Univ, vol. 24, no. 5, pp. 1050–1062, 2017.
31. C. Yang, K. Wang, H. Cheng, C. Kuo, and W. C. C. Chu, “Green Power Management with Dynamic Resource Allocation for Cloud Virtual Machines,” in IEEE International Conference on High Performance Computing and Communications Green, 2011, pp. 726–733.
32. A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing,” Futur. Gener. Comput. Syst., vol. 28, no. 5, pp. 755–768, 2012.

AUTHORS PROFILE

Fajar Kusumaningayu was born in Madiun, Indonesia on March 11th, 1995. She has received his first degree from Sampoerna University on 2013 with research topic Human Computer Interaction (HCI) with orange technologies for people with different abilities. She is currently pursuing a master degree in Computer Science from Bina Nusantara University since 2018. Her current research interests are task scheduling and optimization algorithm. Fajar is currently working as a Programmer in a hybrid framework and excited to work as individual and in team. She loves to learn a new topic and explore the knowledge, as well as embrace social activities.

Antoni Wibowo has received the first degree of Applied Mathematics in 1995 and a master degree of Computer Science in 2000. In 2003, He awarded a Japanese Government Scholarship (Monbukagakusho) to attend Master and PhD programs at Systems and Information Engineering in University of Tsukuba-Japan. He completed the second master degree in 2006 and PhD degree in 2009, respectively. His PhD research focused on machine learning, operations research, multivariate statistical analysis and mathematical programming, especially in developing nonlinear robust regressions using statistical learning theory. He has worked from 1997 to 2010 as a researcher in the Agency for the Assessment and Application of Technology – Indonesia. From April 2010 – September 2014, he worked as a senior lecturer in the Department of Computer Science - Faculty of Computing, and a researcher in the Operation Business Intelligence (OBI) Research Group, Universiti Teknologi Malaysia (UTM) – Malaysia. From October 2014 – October 2016, he was an Associate Professor at Department of Decision Sciences, School of Quantitative Sciences in Universiti Utara Malaysia (UUM). Dr. Eng. Wibowo is currently working at Binus Graduate Program (Master in Computer Science) in Bina Nusantara University-Indonesia as a Specialist Lecturer and continues his research activities in machine learning, optimization, operations research, multivariate data analysis, data mining, computational intelligence and artificial intelligence.

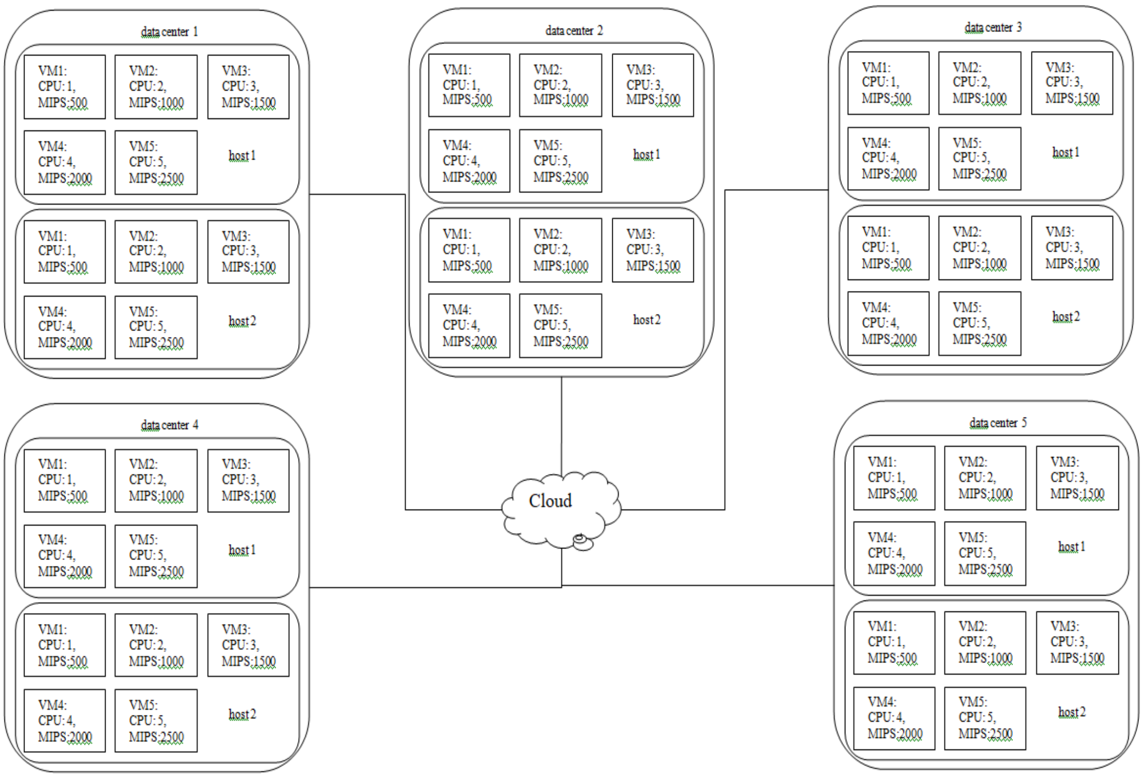

Fig 7. Cloud Computing Data Center Visualization

Table- IV: Comparison Among the Algorithms

Tasks	Aggregate	GA				PSO			
		Energy	MS(s)	LB	Execution (ms)	Energy	MS(s)	LB	Execution (ms)
100	Min	**2597.989**	**24.7**	5206.92005	50	2666.014	25.607	**5019.07749**	69
	Avg	2887.249	28.5568	5947.54558	72.6	2868.21775	28.30305	6023.43058	78.6
	Max	3052.339	30.758	6859.24471	212	3136.339	31.878	7346.00835	151
200	Min	**5413.269**	**51.856**	**6781.30284**	228	5596.119	54.294	6953.6788	143
	Avg	**5805.174**	**57.0814**	8367.66027	243.65	5857.4715	57.7787	8388.96133	152.2
	Max	6094.119	60.934	9544.30819	296	**6089.469**	**60.872**	10058.1523	177
300	Min	**8482.213**	**82.074**	7553.37192	623	8559.463	83.104	8610.66245	238
	Avg	8991.7555	88.8679	**9761.21511**	637.9	**8873.6905**	**87.2937**	10425.3388	247.85
	Max	9458.113	95.086	**11654.0305**	667	**9231.163**	**92.06**	11749.9649	256
400	Min	**11341.771**	**110.142**	**7986.36277**	1327	11559.271	113.042	9130.10265	333
	Avg	12012.0085	119.0785	**10858.0301**	1766.75	**11938.786**	**118.1022**	10994.8369	**351.45**
	Max	12522.421	125.884	13300.4192	2049	**12309.871**	**123.05**	13614.8367	**364**
500	Min	14750.851	143.924	9867.54139	2963	**14652.451**	**142.612**	9761.86643	**454**
	Avg	15302.326	151.277	12827.0446	3531.65	**15293.761**	**151.1628**	12221.5609	**482.15**
	Max	15745.051	157.18	16306.7423	3645	**15697.951**	**156.552**	14474.1347	**505**
600	Min	18074.629	176.744	11458.3464	6018	18108.979	177.202	**10737.0524**	**605**

	Avg	18708.499	185.1956	13969.482	6072.6	**18693.619**	**184.9972**	**13429.068**	**631.95**
	Max	19330.879	193.494	16264.4662	6136	**19049.029**	**189.736**	**16201.0302**	**827**
700	Min	**21186.92**	**207.594**	12745.0448	9158	21382.82	210.206	**10953.5171**	**749**
	Avg	21858.8975	216.5537	**14481.7199**	9247.05	21848.8325	216.4195	14552.333	**1108.35**
	Max	22504.67	225.164	**17018.8628**	9548	22440.77	224.312	17112.7693	**1163**
800	Min	**24260.885**	**238.318**	**12481.2341**	13463	24291.035	238.72	14087.2128	**919**
	Avg	24952.55	247.5402	**15126.3365**	13561.15	**24836.9675**	**245.9991**	16085.4983	**1133.6**
	Max	25328.135	252.548	**18138.8424**	14227	**25305.635**	**252.248**	19071.4311	**1428**
900	Min	26967.934	264.782	13979.2002	18691	**26775.484**	**262.216**	14691.5544	**1115**
	Avg	27591.679	273.0986	17120.3793	18901.35	**27495.3115**	**271.8137**	17245.4416	**1560.6**
	Max	28387.084	283.704	**19110.9281**	19646	**28048.984**	**279.196**	21294.1541	**1747**
1000	Min	29661.879	290.858	14088.1821	25190	**29617.779**	**290.27**	14841.6562	**1304**
	Avg	30604.929	303.432	17941.3837	25506.45	**30510.999**	**302.1796**	**17491.9811**	**1827.6**
	Max	31315.029	312.9	25052.4726	26419	**31129.029**	**310.42**	**21250.3113**	**2270**
Average	Min	**16273.834**	**159.0992**	10214.7507	7771.1	16320.9415	159.7273	10478.6381	**592.9**
	Avg	16871.5068	167.06817	12640.0797	7954.115	**16821.7656**	**166.404955**	12685.845	**757.435**
	Max	17373.784	173.7652	15325.0317	8284.5	**17243.824**	**172.0324**	**15217.2793**	**888.8**

Table- IV: Continue Comparison Among the Algorithms

Tasks	Aggregate	CSA				BA			
		Energy	MS(s)	LB	Execution (ms)	Energy	MS(s)	LB	Execution (ms)
100	Min	3028.639	30.442	5726.2527	1310	2647.189	25.356	5112.24416	41
	Avg	3237.4465	33.2261	6297.59048	1345.75	**2868.0865**	**28.3013**	**5852.81045**	**56.45**
	Max	3428.989	35.78	6828.97765	1577	**3048.739**	**30.71**	**6840.99755**	**137**
200	Min	6008.319	59.79	7150.57605	2148	5483.319	52.79	7285.02701	**105**
	Avg	6426.2265	65.3621	8619.94848	2296.55	5851.6815	57.7015	**8185.66308**	**126.85**
	Max	6727.719	69.382	10256.6401	3138	6181.269	62.096	**9441.28096**	**157**
300	Min	8851.963	87.004	8418.06511	3012	8494.513	82.238	**7471.50151**	**186**
	Avg	9561.0505	96.4585	10387.5678	3979.3	8980.828	88.7222	9892.12733	**227.55**
	Max	10029.163	102.7	12192.7454	4711	9504.613	95.706	12664.7293	**290**
400	Min	11981.371	118.67	9579.8989	3999	11473.021	111.892	9034.30513	**303**
	Avg	12563.986	126.4382	11269.5691	5525.55	11959.186	118.3742	11276.5521	358.95
	Max	12812.821	129.756	**12621.6932**	6077	12289.021	122.772	13357.7865	421
500	Min	15503.851	153.964	11109.4724	5806	14773.651	144.228	**8818.41923**	446
	Avg	16150.9285	162.5917	13401.3659	7098.65	15387.2935	152.4099	**11974.3455**	561.65
	Max	16698.001	169.886	16012.1062	7790	15815.401	158.118	**14222.1234**	645

600	Min	18808.429	186.528	12275.7377	7388	**17953.729**	**175.132**	10772.7969	674
	Avg	19502.1715	195.7779	14968.9938	8751.5	18771.7015	186.0383	13659.4165	1071.65
	Max	20035.429	202.888	16940.3759	9248	19267.579	192.65	18020.5832	1568
700	Min	21731.57	214.856	13205.7673	8936	21327.92	209.474	12181.7373	918
	Avg	22659.245	227.225	15644.767	10442.5	**21825.29**	**216.1056**	15116.6135	1490
	Max	23318.72	236.018	17929.7769	10939	**22280.42**	**222.174**	17910.3135	1870
800	Min	24600.185	242.842	12661.0256	11416	24295.985	238.786	13634.2512	1729
	Avg	25656.755	256.9296	16801.9926	12205.5	24872.8475	246.4775	15475.0762	2226.35
	Max	26437.385	267.338	19799.5559	12713	25393.835	253.424	19316.6589	2538
900	Min	28100.884	279.888	15075.5946	12929	27139.984	267.076	**13804.5456**	2309
	Avg	28729.639	288.2714	18078.5157	13867.25	27751.5115	275.2297	**16890.9699**	2627.75
	Max	29420.134	297.478	20294.1257	14757	28302.334	282.574	19962.0419	3020
1000	Min	30553.029	302.74	13474.7481	15420	30264.279	298.89	**12484.6602**	2632
	Avg	31487.2065	315.1957	19254.9677	15616.45	30739.869	305.2312	17881.3205	3206.15
	Max	32120.979	323.646	22475.927	16291	31551.429	316.052	21688.0645	3712
Average	Min	16916.824	167.6724	10867.7138	7236.4	16385.359	160.5862	**10059.9488**	934.3
	Avg	17597.4655	176.74762	13472.5278	8112.9	16900.8295	167.45914	**12620.4895**	1195.335
	Max	18102.934	183.4872	15535.1924	8724.1	17363.464	173.6276	15342.458	1435.8