ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse207942020.pdf
https://doi.org/10.30534/ijatcse/2020/207942020

# An Optimization on Task Scheduling for Makespan, Energy Consumption, and Load Balancing in Cloud Computing Using Meta-Heuristic

**Fajar Kusumaningayu[1], Antoni Wibowo[2]**
[1]Binus Graduate Program - Master of Computer Science Bina Nusantara University Jakarta, Indonesia, fajarkusumaningayu@binus.ac.id
[2]Binus Graduate Program - Master of Computer Science Bina Nusantara University Jakarta, Indonesia, anwibowo@binus.edu

**ABSTRACT**

The 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. To solve this problem, previous studies have utilized a meta-heuristic method to find the best solution based on the solution spaces. This study aims to compare four meta-heuristic such as the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Clonal Selection Algorithm (CSA), and Bat Algorithm (BA) to solve the multi-objective task scheduling to achieve the optimum solution. This study converts three objectivities into single objectivity optimization with each objectivity act as variable assigned with the weight that presents its priority and has implemented those meta-heuristics. The simulation result from sixteen datasets that have been grouped into three for a small dataset, medium dataset, and large dataset. In small and medium dataset BA able to outperforms others while in large dataset PSO shows better performance.

**Key words :** Meta-heuristic, Multi-Objectivities, Optimization, Task Scheduling

## 1. INTRODUCTION

Cloud computing spend high energy consumption, in 2016 for 289 data centers in Europe they reached 3,735,735 MWh as total energy consumption [1]. 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 [2], and encourages the community to embrace green cloud computing technology.

In Task scheduling, 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 to give efficient services to users[3]. 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 the 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 standard deviation. 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 equally important for three aspects. From the previous studies, task scheduling problems tend to be solved using a meta-heuristic algorithm. The study will aim to compare and the best algorithm to solve the optimization of makespan, energy consumption, and load balancing using four meta-heuristic such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Clonal Selection Algorithm (CSA), and Bat Algorithm (BA).

## 2. LITERATURE REVIEW

This section discuss about the related works regarding task scheduling in cloud computing and their approach, and the proposed algorithm used for the simulation.

### 2.1 Related Works

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. Based on 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 1:** Summary of Related Works

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

### 2.2 Proposed Algorithms

This section will discuss four meta-heuristic such as the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Clonal Selection Algorithm (CSA), and Bat Algorithm (BA) which has been used to solve task scheduling optimization in the previous studies.

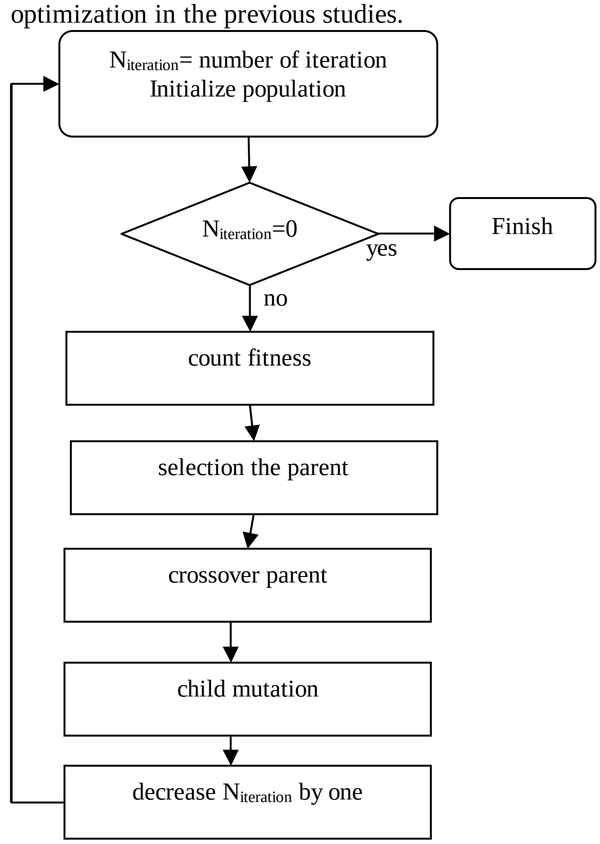


**Figure 1:** GA flowchart

Genetic algorithm is one of the metaheuristic algorithms inspired by Genom. The starting solution is generated randomly, then count the fitness of the solution from the fitness result the algorithm will determine the parent of the solution, from the parent the crossover function will be executed to generate the child solution, then the child will undergo some mutation to be considered as the next solution space and the process will be repeated until the condition is satisfied [25]. Figure 1 shows the flowchart of Genetic Algorithm.

### *B. Particle Swarm Optimization (PSO)*

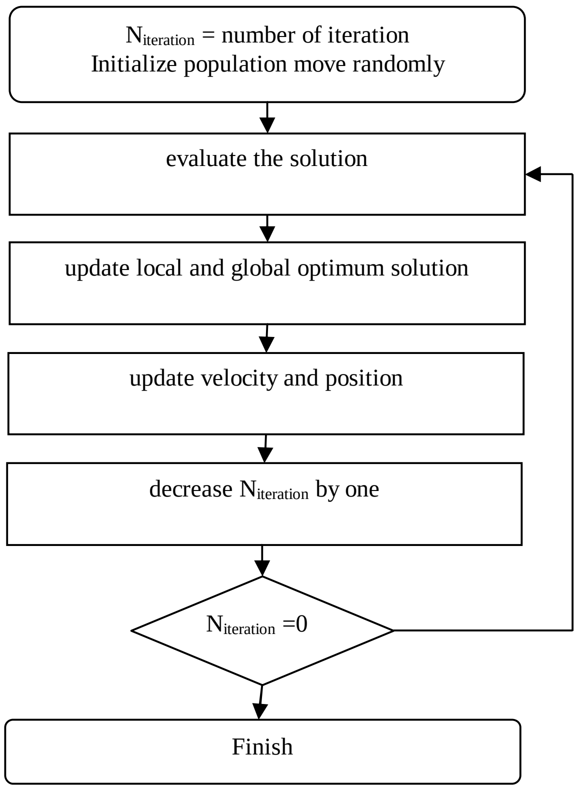


**Figure 2:** PSO Flowcharts

PSO is one of the widely known algorithm to solve task scheduling. PSO observe the movement of bird to find the food source, the using their local information as well as listening to global information produced by other population to determine the best path they need to take to arrive at their destination. The bird of continuously to update their velocity and position to be closer to the food source [26]. The detail of algorithm is presented in Figure 2. 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 the bird, $w, \rho_1, \rho_2$ are coefficient assigned weight, $r_1$, $r_2$ are random vectors, $X_{1n\_bird}$ [t] is the local optimum solution and $X_{2n\_bird}$ [t] is the global optimum solution.

### *C. Clonal Selection Algorithm (CSA)*

CSA is a meta-heuristic algorithm inspired by antibodies system, using cell B and cell T for its cloning, selection, and memory set. At the beginning of the clonal selection algorithm (CLONALG) is used as machine learning and pattern recognition proposed, where it empathizes on its ability to store several solutions not all solutions that provided the best outcome for other information rather than starting from the start again. However, for its potential, the CLONALG is implemented for optimization with three adjustments which are first no explicit antigen thus antibody population does not need to make separate memory. Second select *n* antibodies rather than select the best individual and third, assume all antibodies selected for cloning (*N*) will be cloned in the same number. The number of cloned antibodies will be counted with the equation [27][28]. Figure 3 shows the flow of CSA.

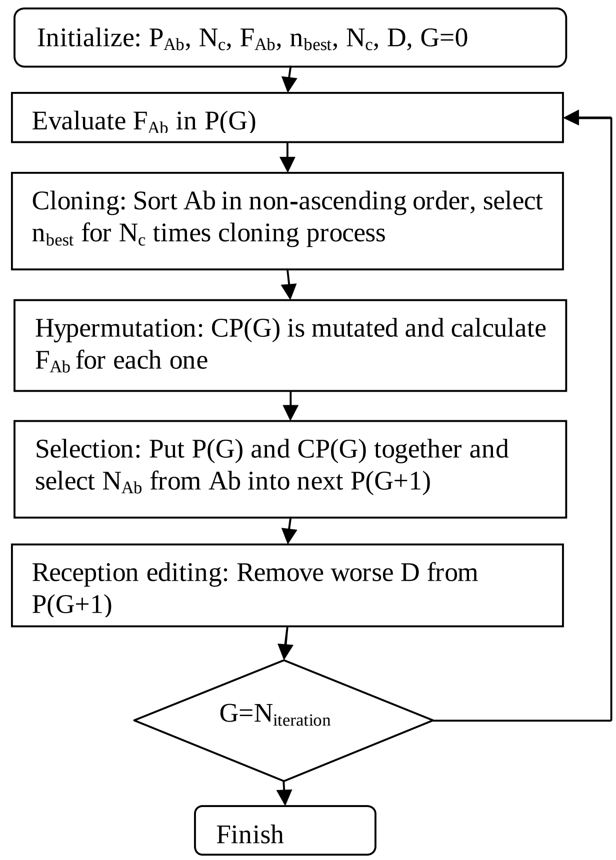


**Figure 3:** CSA Flowcharts

$$N_c = \sum_{i=1}^{N_{Ab}} round\left(\frac{\beta_{CSA} N_{Ab}}{i}\right) \quad (3)$$

Where $N_c$ is the number of cloned antibodies, $P_{Ab}$ is a list of antibodies, $N_{Ab}$is the number of antibodies in Ab, $\beta_{CSA}$ is multiplying factor, D is constantan contain how many antibodies need to be replaced, $n_{best}$ is the best $n_{best}$ number to be selected, $F_{Ab}$ is affinities of Ab.

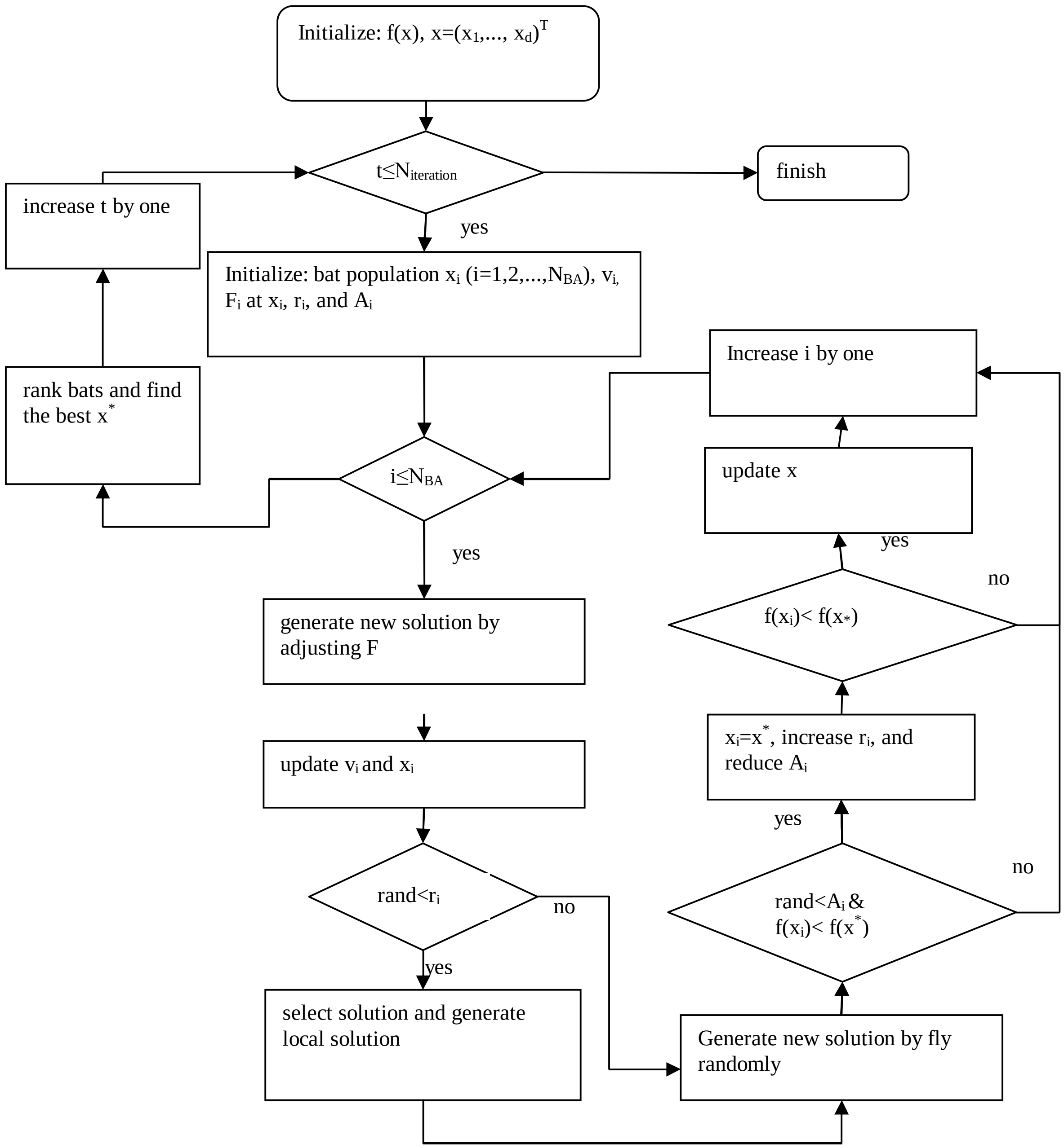


**Figure 4:** BA Flowcharts

*D. Bat Algorithm (BA)*

As the name suggested, it is meta-heuristic that inspired by bats who can travel to search for food even in the dark. Bat has several senses like eye and smell to help them functionalize. However, due to avoiding complex computational processes, BA utilizes echolocation and associated behavior. There are three assumptions while implemented BA which is: bats can differentiate food and barrier, bats fly randomly and able to adapt their wavelength, frequency, and rate of their pulse based on the distance between bats and target, and range of loudness are around large positive to a minimum constant value [29] and capable to outperform PSO [30]. The algorithm is presented in Figure 4.

Update velocity

| | |
|---|---|
| $F_i = F_{\min} + (F_{\max} - F_{\min})\beta_{BA}$ | (4) |
| $v_i^t = v_i^{t-1} + (x_i^t - x_*)F_i$ | (5) |

Update the position/solution

| | |
|---|---|
| $x_i^t = x_i^{t-1} + v_i^t$ | (6) |

Update new solution

| | |
|---|---|
| $x_{new} = x_{old} + \varepsilon A^t$ | (7) |

Update the loudness and rate pulse emission

$$A_i^{t+1} = \alpha_{BA} A_i^t ; r_i^{t+1} = r_i^0 \left[1 - \exp\left(-\gamma_{BA^t}\right)\right] \quad (8)$$

Where v is velocity, x is position or solution, F is frequency, f is the objective function, A is loudness, R is rate pulse emission, $\beta_{BA}$ is random vector $\in [0,1]$, $\alpha_{BA}, \gamma_{BA}$ are constants, $r_i^0$ is initial emission rate, $x_*$ is the best solution.

## 3. METHODOLOGY

This study workflow is presented in Figure 5, it starts from the mathematical model, algorithm implementation, evaluation for each of parameters, and reporting.

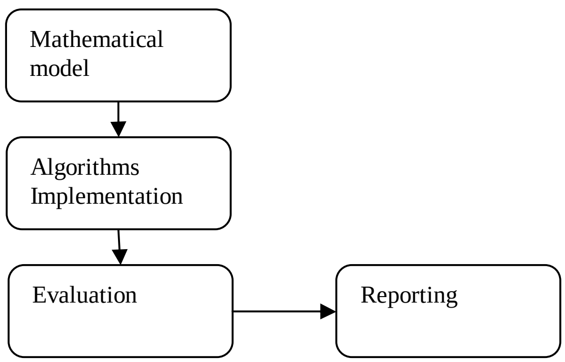


**Figure 5:** Research Workflow

### 3.1 Mathematical Models

Three objectivities of this study are makespan (MS), energy consumption (Etotal), 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 the 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 standard deviation of tasks assigned across all the VMs.

### 3.2 Algorithm Implementation

The simulation has been conducted on Java Netbean 8.2 The detail of the experimental setting is listed in Table 2.

### 3.3 Evaluation

This study will implement four algorithms which are PSO, GA, CSA, and BA. The testing process will be repeated ten times for each dataset to determine each algorithm best, average, and worst result. The parameters that will be evaluated are fitness and algorithm running time. To find the best condition for the four algorithms, each of the algorithms is tested with different iterations to find the optimum iteration then the chosen iteration for each meta-heuristic will be used for comparison across the four algorithms.

**Table 2:** Experiment Settings

| 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 | [100-3000] |
| Task Instruction Length | [200-15000]M |
| Number of testing for each dataset | 10 |
| Number of iteration | 50-1000 |
| 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 |

## 4. METHEMATICAL 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 tasks 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.

### 4.1 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 independent of 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 [11][9]. 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.

$$MS = \max\{C_{ij}, \forall i, \forall j\} \tag{9}$$

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

### 4.2 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. Since the VM(s) have an identical core, the energy will be represented for each core by 1J/s.
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 [14]. This study will count the energy used during the VM idle time [16]. 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.
Decision variable for energy consumption are Ec and Eidle

$$Ec_i = C_{ij} \times \mathrm{P_i} \tag{10}$$

$$Eidle_i = (MS - C_{ij}) \times \mathrm{Pidle_i} \tag{11}$$

$$E_{total} = \sum_{i=1}^{m} (Ec_i + Eidle_i) \tag{12}$$

Where Etotal is the sum of energy needed during computation and idle time, Ec is the energy consumption and Eidle is the energy during idle time. $Pidle_i$ is the power used by VM index i during idle time, $P_i$ is core optimum power, MS is the makespan, Cij is computation time needed by VM index i to solve task j.

### 4.3 Load Balancing

This study will implement four algorithms which are PSO, GA, CSA, and BA. The testing process will be repeated ten times for each dataset to determine each algorithm best, average, and worst result. The parameters that will be evaluated are fitness and algorithm running time. To find the best condition for the four algorithms, each of the algorithms is tested with different iterations to find the optimum iteration then the chosen iteration for each meta-heuristic will be used for comparison across the four algorithms.
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 standard deviation formula should equal zero, therefore lowering the standard deviation result means that the tasks are closer to be equally distributed. The standard deviation function used in the study is

$$\overline{\mathrm{Task}} = \frac{\sum_{j=1}^{n} (Task_j)}{m} \quad \mathrm{j} \in \{1,2,3..., \mathrm{n}\} \tag{13}$$

$$LB = \sqrt{\frac{\sum_{i=1}^{m} \left({}^{\#}Task_{ij} - \overline{Task}\right)^2}{m}} \tag{14}$$

$$\{1,2,3..., \mathrm{Task_{ij}}, i \in m\},\ \mathrm{j} \in \{1,2,3...,\mathrm{n}\}$$

Where LB is the standard deviation of load balancing, ${}^{\#}Task_{ij}$ is the sum of instruction length $task_j$ in VM i.

### 4.4 Fitness Function

Reducing the makespan, energy consumption, and load balancing using the task scheduling approach is the main goal of this study. Therefore, function addressing three of these objectives need to be delivered. One may found the other to be more important than the other aspect. Therefore, the value of α, β, and γ is used to determine to prioritize the fitness function.

$$\min F = \alpha \times MS + \beta \times Etotal + \gamma \times LB \tag{15}$$

$$\alpha + \beta + \gamma = 1$$

## 5. RESULT AND DISCUSSION

This section will discuss the optimum iteration for each algorithm then the comparison of GA, PSO, CSA, and BA for fitness, makespan, energy consumption, load balancing, and running time.

### 5.1 Optimum Iteration

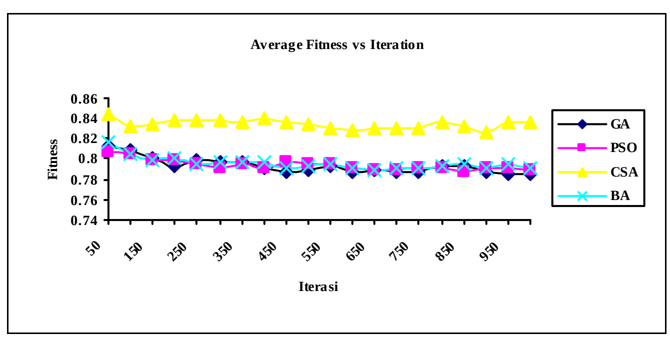


**Figure 6:** Four Meta-Heuristic Fitness for Each Iteration

To have fair treatment conditions for comparison, one should find the best optimum iteration used by each algorithm to solve the task scheduling problem in one dataset. In this section, the study uses 1000 datasets with ten times repeated tests for 50-1000 iteration. From the data come in Figure 6 and Figure 7 the optimum iteration has been chosen for each algorithm such as GA will have 200 iterations, PSO 300 iterations, CSA using 600 iterations and BA will run for 450 iterations.

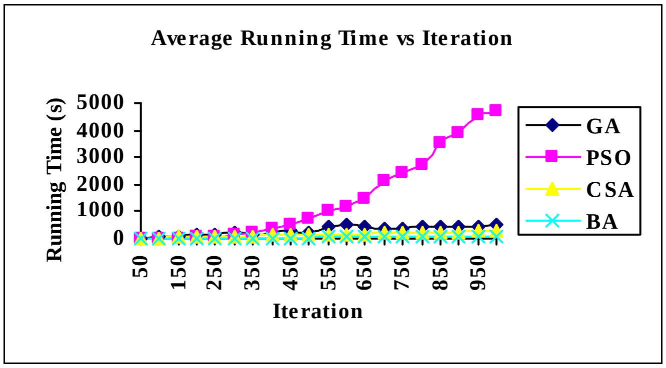


**Figure 7:** Four Meta-Heuristic Running Time for Each Iteration

### 5.2 Result

The experiment conducted by divided the dataset into three groups which are small dataset, medium dataset, and large dataset for 100-3000 tasks.

**Table 3:** Fitness Comparison between Four Meta-Heuristic for Small Dataset.

| Tasks | Aggregate | GA | PSO | CSA | BA |
|---|---|---|---|---|---|
| 100 | min | 0.8774 | 0.8731 | 0.9559 | **0.8719** |
| | avg | 0.9019 | 0.8997 | 0.9828 | **0.8917** |
| | max | 0.9325 | 0.9269 | 1.0419 | **0.9054** |
| 150 | min | 0.8259 | 0.8327 | 0.9372 | **0.8244** |
| | avg | 0.8691 | **0.8613** | 0.9779 | 0.8667 |
| | max | 0.9064 | 0.8949 | 1.0151 | **0.8862** |
| 200 | min | 0.7925 | 0.7908 | 0.8651 | **0.7892** |
| | avg | 0.8327 | 0.831 | 0.9125 | **0.8303** |
| | max | 0.8829 | 0.8734 | 0.9625 | **0.8505** |
| 250 | min | 0.8147 | 0.7921 | 0.8492 | **0.7914** |
| | avg | 0.8344 | 0.8306 | 0.8924 | **0.8151** |
| | max | 0.8616 | 0.8503 | 0.9292 | **0.8345** |
| 300 | min | 0.818 | **0.7949** | 0.8712 | 0.8095 |
| | avg | 0.8336 | **0.8181** | 0.8912 | 0.8213 |
| | max | 0.8455 | 0.8428 | 0.9244 | **0.8367** |

*A. Small Dataset*

The small dataset made up of 100-300 tasks with 50 tasks different for each dataset. Therefore in a small dataset, there are five datasets. From the fitness average result, BA gives the best performance for three datasets, followed by PSO with two datasets. However, for the best maximum and minimum BA shows the best performance in almost all of the dataset. In average running time GA gives the fastest running time for four datasets, and all of the datasets for best maximum and minimum. The detail result for small dataset is presented in Table 3.

**Table 4:** Fitness Comparison between Four Meta-Heuristic for Medium Dataset.

| Tasks | Aggregate | GA | PSO | CSA | BA |
|---|---|---|---|---|---|
| 400 | min | 0.7912 | 0.8141 | 0.8291 | **0.7899** |
| | avg | 0.8176 | 0.8229 | 0.8714 | **0.8039** |
| | max | 0.8352 | 0.8295 | 0.9004 | **0.8211** |
| 500 | min | **0.7867** | 0.7966 | 0.8518 | 0.8043 |
| | avg | 0.8114 | **0.805** | 0.8654 | 0.8129 |
| | max | 0.8249 | **0.8145** | 0.8755 | 0.8253 |
| 600 | min | 0.789 | **0.7846** | 0.8142 | 0.7879 |
| | avg | 0.8022 | **0.8014** | 0.8432 | 0.802 |
| | max | 0.8166 | 0.8166 | 0.8725 | **0.8089** |
| 700 | min | **0.7845** | 0.7899 | 0.8219 | 0.793 |
| | avg | 0.8006 | 0.8009 | 0.8393 | **0.8003** |
| | max | 0.8121 | **0.8087** | 0.8501 | 0.8116 |
| 800 | min | 0.7912 | 0.7917 | 0.8143 | **0.7798** |
| | avg | 0.8024 | 0.7984 | 0.8385 | **0.7948** |
| | max | 0.8124 | 0.8057 | 0.8469 | **0.8047** |
| 900 | min | 0.7883 | 0.7894 | 0.8178 | **0.7777** |
| | avg | 0.8007 | 0.7997 | 0.835 | **0.7916** |
| | max | 0.8118 | 0.8052 | 0.8519 | **0.8041** |

**Table 5:** Fitness Comparison between Four Meta-Heuristic for Large Dataset.

| Tasks | Aggregate | GA | PSO | CSA | BA |
|---|---|---|---|---|---|
| 1000 | min | 0.7807 | **0.7789** | 0.8105 | 0.7818 |
| | avg | 0.7926 | **0.7903** | 0.8292 | 0.7911 |
| | max | **0.8008** | 0.8059 | 0.8476 | 0.8022 |
| 1500 | min | **0.7794** | 0.7834 | 0.8155 | 0.7881 |
| | avg | 0.7917 | **0.7913** | 0.8229 | 0.7922 |
| | max | 0.8032 | 0.8019 | 0.8286 | **0.7975** |
| 2000 | min | 0.7819 | **0.7751** | 0.7974 | 0.7762 |
| | avg | 0.7902 | 0.7868 | 0.8142 | **0.7862** |
| | max | 0.8015 | 0.7953 | 0.8215 | **0.7892** |
| 2500 | min | 0.7787 | 0.782 | 0.7956 | **0.7782** |
| | avg | 0.7897 | 0.788 | 0.8126 | **0.787** |
| | max | 0.8034 | 0.793 | 0.8207 | **0.7909** |
| 3000 | min | 0.7838 | **0.7752** | 0.799 | 0.7814 |
| | avg | 0.7911 | **0.7836** | 0.8062 | 0.7866 |
| | max | 0.7986 | **0.7898** | 0.8135 | 0.7913 |

*B. Medium Dataset*

The medium dataset made up of 400-900 tasks with 100 tasks different for each dataset. Therefore in the medium dataset, there are six datasets. From the average fitness, result BA yields the best result in four datasets followed by PSO, then for best maximum and minimum BA give the best result almost in all of the datasets. For average running time, maximum, and minimum BA gives the smallest running time for all of the datasets in a medium dataset. The detail result for medium dataset is presented in Table 4.

### *C. Large Dataset*

The large dataset made up of 1000-3000 tasks with 500 tasks different for each dataset. Therefore in a large dataset, there are five datasets. Average fitness results from a large dataset, PSO gives the best result for three datasets then BA for two datasets. While for the best maximum and minimum PSO and BA yield the best result for four datasets while GA for two datasets. For average running time, maximum, and minimum BA gives the smallest running time for all of the datasets in large datasets. The detail result for large dataset is presented in Table 5.

**Table 6:** Average Fitness and Running Time(s) from Four Meta-Heuristic for Each Dataset Group

| Tasks | Aggregate | GA | | PSO | | CSA | | BA | |
|---|---|---|---|---|---|---|---|---|---|
| | | Fitness | Running Time | Fitness | Running Time | Fitness | Running Time | Fitness | Running Time |
| Avg small dataset | min | 0.826 | **1.316** | **0.817** | 28.313 | 0.896 | 32.222 | 0.817 | 1.968 |
| | avg | 0.854 | **1.59** | 0.848 | 31.082 | 0.931 | 35.17 | **0.845** | 2.145 |
| | max | 0.886 | **1.826** | 0.878 | 35.77 | 0.975 | 38.668 | **0.863** | 2.566 |
| Percentage from optimum result | min | 1.09% | 0.00% | 0.00% | 95.35% | 8.82% | 95.92% | 0.07% | 33.13% |
| | avg | 1.09% | 0.00% | 0.37% | 94.88% | 9.27% | 95.48% | 0.00% | 25.86% |
| | max | 2.61% | 0.00% | 1.71% | 94.90% | 11.49% | 95.28% | 0.00% | 28.87% |
| Avg medium dataset | min | **0.788** | 31.732 | 0.794 | 96.306 | 0.825 | 100.857 | 0.789 | **11.622** |
| | avg | 0.806 | 34.452 | 0.805 | 101.763 | 0.849 | 106.344 | **0.801** | **13.622** |
| | max | 0.819 | 37.181 | 0.813 | 109.697 | 0.866 | 112.063 | **0.813** | **16.131** |
| Percentage from optimum result | min | 0.00% | 63.38% | 0.74% | 87.93% | 4.41% | 88.48% | 0.04% | 0.00% |
| | avg | 0.61% | 60.46% | 0.47% | 86.61% | 5.64% | 87.19% | 0.00% | 0.00% |
| | max | 0.76% | 56.61% | 0.09% | 85.29% | 6.19% | 85.61% | 0.00% | 0.00% |
| Avg large dataset | min | 0.781 | 958.049 | **0.779** | 339.405 | 0.804 | 287.486 | 0.781 | **98.813** |
| | avg | 0.791 | 1010.441 | **0.788** | 354.992 | 0.817 | 300.82 | 0.789 | **118.094** |
| | max | 0.802 | 1042.02 | 0.797 | 377.742 | 0.826 | 309.298 | **0.794** | **141.999** |
| Percentage from optimum result | min | 0.25% | 89.69% | 0.00% | 70.89% | 3.07% | 65.63% | 0.28% | 0.00% |
| | avg | 0.39% | 88.31% | 0.00% | 66.73% | 3.55% | 60.74% | 0.08% | 0.00% |
| | max | 0.91% | 86.37% | 0.37% | 62.41% | 3.89% | 54.09% | 0.00% | 0.00% |

## 5.3 Discussion

To ensure that meta-heuristic give their optimum solution the number of iteration is tested for each algorithm so that increasing the number of iterations will not give huge influence on the result. Combining the fitness result and running time needed to solve the task scheduling. The study used 200 iterations for GA, PSO 300, CSA 600, and BA use 450 iterations.

Some information can be derived from the result such as optimum makespan results that resemble energy consumption. This is caused by the fact of energy consumption using Makespan in the calculation process, especially during idle time. While load balancing has an opposite behavior, this is caused by load-balancing goal is having equally distributed tasks across the VM however since the VM which have different core as its specification making them have different speed. Therefore, load balancing has an inverse relationship with makespan. The table shows the conclusion of the fitness result from the four meta-heuristics.

Genetic Algorithm (GA) in this study using a half-point crossover with the swap position. GA required fast running time for small datasets. However, if the number of tasks is increasing the time required will expand. In its best condition GA able to outperform other algorithms in makespan, energy consumption, and load balancing but not for fitness.

PSO using its velocity and position to determine how many times the swap position is required to be done to the previous solutions to yield a better result. During PSO best performance it able to yield fitness best results for small and medium datasets, while for large dataset PSO able to beat BA in three datasets. In this simulation, PSO shows its competitive side however PSO required a quite large running time compared to other algorithms.

During simulation, CSA require large memory usage for cloning process, therefore this study limit the cloning number three to ten times to avoid large memory usage for the scheduling process. Even though CSA does not give the closest result to the best solution and CSA required longer running time for small and medium datasets. Nevertheless, since there is cloning limitation in CSA the running time required for larger datasets much more stable, that why CSA has faster running time compare to GA in large datasets.

Bat Algorithm (BA) shows competitive performance especially in small and medium datasets, and for large dataset BA just loses once to PSO. BA is known for fast convergent rate and it is shown in this simulation especially for medium and large datasets since BA gives the fastest running time, and come in second place after GA in running time for small dataset.

In several occurrences BA yield the best maximum and minimum in almost all of dataset, this is caused by BA behavior which does not only rely on fitness result to determine the next solution. BA adopts a random flying technique to give a larger solution space. It makes BA able to give the optimum solution during maximum and minimum. The detail performance of fitness and running time comparison can be seen in Table 6.

## 6. CONCLUSION

Meta-heuristic algorithms have been implemented to solve NP-hard problems like task scheduling whether it is for computation process, industry, and employee scheduling. Based on the previous studies there are four potential meta-heuristic algorithms to solve the scheduling process which are GA, PSO, CSA, and BA. Besides, the latest task scheduling takes interest in multi-objectivities. Based on the study that has been conducted BA and PSO show good performance to solve makespan, energy consumption, and load balancing.

For future references, several points might be used for future work in task scheduling in data center cloud computing such as:

1) For task scheduling which required fast running time, GA will become a suitable choice for small datasets while for larger dataset BA is a better option.
2) The optimum result for task scheduling one can use GA with 200 iterations, PSO with 300 iterations, CSA with 600 iterations, and BA with 450 iterations.
3) PSO and BA is promising algorithms for hybrid
4) Find the objectivity which has not been used before or find the combination of two or more objectivities for task scheduling optimization.

## REFERENCES

[1] 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.

[2] IAE, “Electricity Statistics,” 2017. [Online]. Available: https://www.iea.org/statistics/electricity/. [Accessed: 03-Oct-2019].

[3] 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. https://doi.org/10.1016/j.eij.2017.07.001

[4] M. N. Prasadhu and M. Mehfooza, “An Efficient Hybrid Load Balancing Algorithm for Heterogeneous Data Centers in Cloud Computing,” *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 3, pp. 3078–3085, 2020. https://doi.org/10.30534/ijatcse/2020/89932020

[5] A. Qadir and G. Ravi, “Dual Objective Task Scheduling Algorithm in Cloud Environment,” *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 3, pp. 2527–2534, 2020. https://doi.org/10.30534/ijatcse/2020/07932020

[6] S. K. Panda and P. K. Jana, “Load balanced task scheduling for cloud computing : a probabilistic approach,” *Knowl Inf Syst*, 2019.

[7] J. A. Jennifa, S. T. Revathi, and T. S. S. Priya, “Smart PSO-based secured scheduling approaches for scientific workflows in cloud computing,” *Soft Comput*, vol. 23, no. 5, pp. 1745–1765, 2019.

[8] 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.

[9] 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.

[10] 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. https://doi.org/10.1016/j.icte.2017.08.001

[11] G. Natesan and A. Chokkalingam, “Task scheduling in heterogeneous cloud environment using mean grey wolf optimization algorithm,” *ICT Express*, vol. 5, 2018.

[12] 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.

[13] 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.

[14] R. K. Jena, "Energy Efficient Task Scheduling in Cloud Environment," *Energy Procedia*, vol. 141, pp. 222–227, 2017.

[15] 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.

[16] 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.
https://doi.org/10.1007/s10723-017-9391-5

[17] G. Xie, G. Zeng, X. Xiao, and R. Li, "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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.
https://doi.org/10.1186/s13677-015-0031-y

[23] 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.

[24] 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.

[25] O. Kramer, *Genetic Algorithm Essentials*. Springer International Publishing AG, 2017.

[26] M. Couceiro and P. Ghamisi, *Fractional Order Darwinian Particle Swarm Optimization: Applications and Evaluation of an Evolutionary Algorithm*. Springer, 2016.

[27] W. Luo, X. Lin, T. Zhu, and P. Xu, "A clonal selection algorithm for dynamic multimodal function optimization," *Swarm Evol. Comput. BASE DATA*, 2018.
https://doi.org/10.1016/j.swevo.2018.10.010

[28] L. N. De Castro and F. J. Von Zuben, "Learning and Optimization Using the Clonal Selection Principle," *IEEE Trans. Evol. Comput. Spec. Issue Artif. Immune Syst.*, vol. 6, no. 3, pp. 239–251, 2002.

[29] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. Gonz, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds. 2010, pp. 65–74.

[30] X.-S. Yang, "Bat algorithm : literature review and applications Xingshi He," *Int. J. Bio-Inspired Comput.*, vol. 5, no. 3, p. 2013, 2013.

[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.
https://doi.org/10.1016/j.future.2011.04.017