Available online at: http://e-jurnal.lppmunsera.org/index.php/JSMI
Jurnal Sistem dan Manajemen Industri
ISSN (Print) 2580-2887 ISSN (Online) 2580-2895

A modified Aquila optimizer algorithm for optimization energy-efficient no-idle permutation flow shop scheduling problem

Dana Marsetiya Utama, Nabilah Sanafa*
Department of Industrial Engineering, University of Muhammadiyah Malang, Jl. Raya Tlogomas No 246, Malang, 65144, Indonesia

ARTICLE INFORMATION

Article history:

Received: April 15, 2023
Revised: July 20, 2023
Accepted: September 02, 2023

Keywords:

Aquila optimizer (AO)
Consumption energy
Flow shop
No idle

A B S T R A C T

Increasing energy consumption has faced challenges and pressures for modern manufacturing operations. The production sector accounts for half of the world's total energy consumption. Reducing idle machine time by employing No-Idle Permutation Flow Shop Scheduling (NIPFSP) is one of the best decisions for reducing energy consumption. This article modifies one of the energy consumption-solving algorithms, the Aquila Optimizer (AO) algorithm. This research contributes by 1) proposing novel AO procedures for solving energy consumption problems with NIPFSP and 2) expanding the literature on metaheuristic algorithms that can solve energy consumption problems with NIPFSP. To analyze whether the AO algorithm is optimal, we compared by using the Grey Wolf Optimizer (GWO) algorithm. It compares these two algorithms to tackle the problem of energy consumption by testing four distinct problems. Comparison of the AO and GWO algorithm is thirty times for each case for each population and iteration. The outcome of comparing the two algorithms is using a t-test on independent samples and ECR. In all case studies, the results demonstrate that the AO algorithm has a lower energy consumption value than GWO. The AO algorithm is therefore recommended for minimizing energy consumption because it can produce more optimal results than the comparison algorithm.

*Corresponding Author

Nabilah Sanafa
E-mail: nabilahsnfa@gmail.com

1. INTRODUCTION

The manufacturing sector faces challenges and pressures due to rising energy consumption and production's negative environmental impact. Globally, production consumes 50% of all energy [1] and is projected to increase to 45% by 2023 [2]. Germany has the highest energy consumption rates, with manufacturing accounting for 47% of total energy consumption [3]. About 34% of all energy consumed in the United States is consumed by the industrial sector, with electricity accounting for 3.4% [4]. China consumed 70.82 percent of the world's energy in 2011, with the industrial sector consuming 81.32 percent [5] and the transportation sector consuming 30 percent [6]. As a result of high fuel costs and harmful environmental effects such as global warming and CO2 emissions, large energy consumption is a major issue for manufacturing companies [7]. As a result of the industrial sector's high energy consumption, manufacturing businesses are responsible for the surrounding environment [8]. They demand a production method that demonstrates efforts to reduce environmental impact [9]. According to one study, environmental management and green principles are applied to manufacturing businesses [10] to reduce energy consumption, which in turn lessens ecological impacts [11], [12].

Other research indicates that reducing machine idle or no-idle time is one method for reducing energy consumption [13] (i) to reduce costs so that human resources, capital, and natural resources can be used for other purposes; (ii) to relieve pressure on petroleum supplies as we enter the 21st century; (iii) to reduce the rate of greenhouse gas emissions, thereby reducing the need for drastic measures in this area; and (iv) to pave the way for the implementation of alternatives to fossil fuels [14]. Several studies have conducted proofs regarding the minimization of energy consumption by employing the No-Idle Permutation Flow Shop Scheduling Problem (NIPFSP) [15], specifically by employing the Iterated Greedy Algorithm [16], the Tabu search (TS) and the Genetic Algorithm (GA) [17], the Iterated reference greedy algorithm [18], the Invasive weed optimization algorithm [19], Memetic algorithm with node and edge histogram [20], collaborative optimization algorithm [21], novel differential evolution algorithm [22], discrete artificial bee colony algorithm [23], a hybrid discrete particle swarm optimization algorithm [24], a hybrid discrete differential evolution algorithm [25], Hybrid Grasshopper Optimization Algorithm [26], the hybrid ant lion optimization flow shop [27]. Some of these studies specifically discuss No-idle, a research Al-Imron *et al.* [28] that aims to minimize energy consumption using the Grey Wolf Optimizer Algorithm. Some studies also analyze Flow Shop Scheduling by using a multi-operator hybrid genetic algorithm [29], multiobjective distributed reentrant permutation flow shop scheduling with sequence-dependent setup time [30], A decision support system for road freight transportation route selection with new fuzzy numbers [31], A systematic literature review on energy-efficient hybrid flow shop scheduling [32], A novel hybrid Archimedes optimization algorithm for energy-efficient hybrid flow shop scheduling [5], and design of decision support system for road freight transportation routing using multilayer zero-one goal programming [33]. Previous research that addresses NIPFSP issues is presented in Table 1.

In this study, a comparison will be made between the no-idle flow shop scheduling problem and the previously studied the Grey Wolf Optimizer (GWO) algorithm [28] as a comparison. The Aquila Optimizer (AO) offers advantages such as fast convergence speed, high search efficiency, and a simple structure [34]. However, it has drawbacks, including slow convergence speed, limited local development ability, and decreasing population diversity. On the other hand, the GWO excels in effective exploration and exploitation, versatility and adaptability, and can be improved through modifications [35]. Nonetheless, it is sensitive to parameter settings, may struggle with complex problems, and lacks a strong theoretical foundation.

Numerous algorithms have been proposed to solve NIPFSP, as demonstrated by some research cited above. Many of these algorithms are inspired by animals' hunting and prey-searching behavior [32]. AO has been employed in several previous studies to address various problems. production forecasting [36], global optimization [37], advance feature extraction and selection [38], industrial engineering optimizations problems [39], boosting covid-19 image classification [40], global optimization and constrained engineering problems [41], optimization of PID parameters [42], selecting effective features from medical data [43]. Therefore, the Aquila algorithm will be utilized in this study to solve energy consumption issues. However, no prior research on reducing energy consumption with the AO algorithm exists. Researchers propose the AO algorithm as an alternative strategy for reducing energy consumption in the NIPFSP case. AO is an algorithm proposed by simulating Aquila's behavior during hunting and displaying Aquila's hunting actions at each step [44]. Real-life applications are not the focus of this research, where real-life applications refer to articles with a multiobjective classification. Instead, this study focuses on energy consumption issues by proposing a new algorithm. This Aquila algorithm will be used to reduce manufacturing companies energy consumption during the production process.

Consequently, this study aims to modify the AO algorithm to reduce NIPFSP energy consumption. An optimal energy consumption value using the AO algorithm will be compared with the Grey Wolf Optimizer (GWO) algorithm [32]. Make span and energy consumption are the objective functions used in this research [34]. This study's findings will be compared to determine the most

 http://dx.doi.org/10.30656/jsmi.v7i2.6446

efficient algorithm. This research contributes by 1) proposing new AO procedures for solving energy consumption problems with NIPFSP and 2) expanding the literature on metaheuristic algorithms that can solve energy consumption problems with NIPFSP

Table 1. Literature review on NIPFSP

Year	Research	Variants problems	Classification method: Heuristic	Meta	Hybrid	Exact	Procedure optimization
2007	Baraz and Mosheiov [45]	*NIPFSP / Min Cmax*	-	√	-	-	Greedy algorithm
2007	Pan and Wang [24]	*NIPFSP / Min Cmax*	-	√	-	-	Swarm optimization
2008	Pan and Wang [22]	*NIPFSP / Min Cmax*	-	√	-	-	Differential evolution algorithm
2010	Ren *et al*. [46]	*NIPFSP / Min Cmax*	-	√	-	-	Tabu search
2011	Ren *et al*. [47]	*NIPFSP / Min Cmax*	-	-	√	-	Hybrid tabu search
2011	Tasgetiren *et al*. [48]	*NIPFSP / Min Cmax , Min Ta*	-	√	-	-	Differential evolution algorithm
2012	Deng and Gu [25]	*NIPFSP / Min Cmax*	-	-	√	-	Hybrid discrete differential evo-lution algorithm
2013	Tasgetiren *et al*. [23]	*NIPFSP / Min Cmax , Min Ta*	-	√	-	-	Artificial bee colony algorithm
2013	Tasgetiren *et al*. [49]	*NIPFSP / Min Cmax*	-	√	-	-	The general variable neigh-borhood search algorithm
2013	Wang and Li [50]	*NIPFS / Min Cmax, Min Tar*	-	√	-	-	Shuffled frog leaping algorithm
2014	Zhou *et al*. [19]	*NFSP / Min Cmax*	-	√	-	-	Invasive weed optimization algorithm
2018	Ling-fang *et al*. [51]	*DNIPFSP / Min Cmax*	-	√	-	-	Two-stage memetic algorithm
2019	Tasgetiren *et al*. [52]	*NIPFSP / Min Cmax*	-	√	-	-	Variable iterated local search algorithm.
2021	Li *et al*. [53]	*NIPFSP / Min Cmax*	-	√	-	-	Adaptive iterated greedy algorithm
2021	Cheng *et al*. [54]	*NIPFSP / Min Cmax*	-	-	√	-	Hybrid terated greedy and local search procedure.

2. RESEARCH METHODS

2.1. Assumptions, notation, and mathematical models

The investigation makes use of several assumptions made by researchers. The following are the permutation no-idle flow shop timing presumptions [55]: (1) The m machines must process every one of the n jobs in the same sequence. (2) At 0, all processes are prepared for processing. (3) The processing start time for the first task on each machine must be delayed to comply with the no idle requirement. (4) Only one job can be processed on each machine at the moment, and only one can be processed on each machine at a given time. (5) Once the first task begins processing, it can only be stopped once the last job has finished processing. (6) Setup time is included in the job execution time. (7) Equipment may be kept active while a task is processed. The following formula is used in the problem [56]:

i	:	job index
j	:	machine index
$C_{i,j}$	:	completion time of the *i* job on the *j* machine
F_j	:	completion time of the *j* machine
S_j	:	*j* machine start time
$C_{i,j}$	:	completion time of the *i* job on the *j* machine
p_{ij}	:	process time of the *i* job
C_m	:	make span value
φ_j	:	energy consumption when the machine is idle
τ_j	:	energy consumption when the *i* machine operates
N	:	number of jobs
M	:	number of machines
TEC	:	total energy consumption
P_{ij}	:	processing energy consumption on j machine
θ_j	:	waiting time on the *j* machine
D	:	permutations to avoid overlap

Model of mixed integer programming (MIP) presents the mathematical formulation for minimizing energy consumption:

Decision Variable

$$Y_{ijr} = \begin{cases} 1, & \text{if job } i \text{ is processed at speed } r \text{ on machine } j \\ 0, & \text{otherwise} \end{cases}$$

$$X_{ijr} = \begin{cases} 1, & \text{if job } i \text{ is the predecessor job of job } k \\ 0, & \text{otherwise } (i<k) \end{cases}$$

Objective function:

$$\text{Min TEC} \quad (1)$$

Constraints :

$$C_{i,1} \geq \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_i = \{1,..,\text{n}\} \quad (2)$$

$$C_{ij} - C_{i,j-1} \geq \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_j = \{2,..,\text{m}\}, i = \{2,..,\text{n}\} \quad (3)$$

$$C_{ij} - C_{kj} + DX_{ik} \geq \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_j = \{1,..,\text{m}\}, i = \{1,..,\text{n}\}, k = \{1,..,\text{n}\} \quad (4)$$

$$C_{ij} - C_{kj} + DX_{ik} \leq D - \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_j = \{1,..,\text{m}\}, i = \{1,..,\text{n}\}, k = \{1,..,\text{n}\} \quad (5)$$

$$C_{max} \geq C_{im} \forall_i = \{1,..,\text{n}\} \quad (6)$$

$$\sum_{r=1}^{l} Y_{ijr} = 1 \forall_i = \{1,..,\text{n}\}, \text{j} = \{1,..,\text{m}\}\text{c z} \quad (7)$$

$$Y_{ijr} = Y_{i,j+1,r} \forall_i = \{1,..,\text{n}\}, \text{j} = \{1,..,\text{m}\}, \text{ r} = \{1,..,\text{l}\} \quad (8)$$

$$\theta_j = C_{max} - \sum_{i=1}^{n} \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_j = \{1,..,\text{m}\} \quad (9)$$

$$\text{TEC} = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{r=1}^{l} P_{i1}\, Y_{ilr}\, Y_{ijr} + \sum_{j=1}^{m} \frac{\varphi_j \theta_j \tau_j}{60} \quad (10)$$

$$S_j \leq C_{ij} - \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_i = \{1,..,\text{n}\}, j = \{1,..,\text{m}\} \quad (11)$$

$$F_j \geq C_{ij} \forall_i = \{1,..,\text{n}\}, j = \{1,..,\text{m}\} \quad (12)$$

$$F_j \geq S_j + \sum_{i=1}^{n} \sum_{r=1}^{l} P_{i1}\, Y_{ilr} \forall_i = \{1,..,\text{n}\}, j = \{1,..,\text{m}\} \quad (13)$$

The TEC minimization goal function is a problem constraint (1). The second problem constraint is when the first machine finishes each task. The next operation will be carried out if the preceding operation has been processed, according to problem constraint (3). The order of each task is defined by the problem restrictions (4) and (5). Calculating the make span satisfies problem restriction number six. Problem limitations (7) and (8) suggest that the same machine machining speed is used for all jobs. The idle period for each machine is constrained (9). There is no idle time between tasks, so idle time only occurs at the start and end of the delay. The calculation of total energy usage is shown in constraint (10). Constraints (11), (12), and (13) make sure that each machine is never idle in between tasks.

2.2. Proposed algorithm

2.2.1. Aquila optimizer (AO) algorithm

Researchers have proposed AO algorithm to conserve energy. Aquila is one of the most well-known raptors. Aquila is the most prevalent species of Aquila that disperses. All birds, including Aquila, are members of the Accipitridae family. Aquila is typically dark brown and has golden-brown feathers on its neck. Young members of this group of Aquila have predominantly white tails and typically faint white

 http://dx.doi.org/10.30656/jsmi.v7i2.6446

markings on their wings. Aquila captures various prey, including rabbits, terns, marmots, squirrels, and other land animals, utilizing their speed, agility, strong legs, and large, sharp claws. Observable in nature are Aquila and their typical behavior. Due to their hunting prowess, Aquila is among the most extensively studied birds in the world. When hunting alone, male Aquilas capture significantly more prey. Aquila pursues squirrels, rabbits, and other animals with their speed and sharp claws. Even fully grown deer have been attacked by them in the past. The ground squirrel is Aquila's diet's next most important species. Aquila is a brilliant and skilled hunter who may be second only to humans [44]. Fig. 1, Fig. 2, Fig. 3, Fig. 4, and Fig. 5 depict the assumptions and various conditions of the AO.

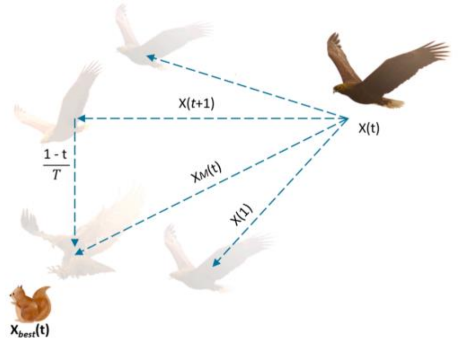

Fig. 1. The soaring behavior of Aquila is characterized by its vertical stoop [44]

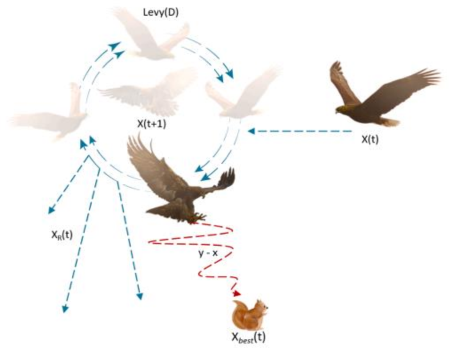

Fig. 2. Aquila exhibits contour flight behavior accompanied by short glide attacks [44]

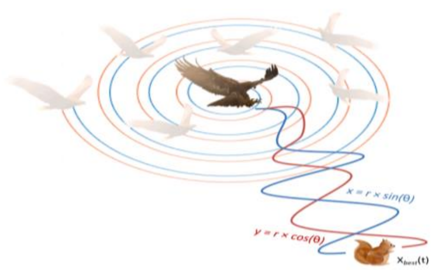

Fig. 3. The behavior of the AO involves a spiral shape [44]

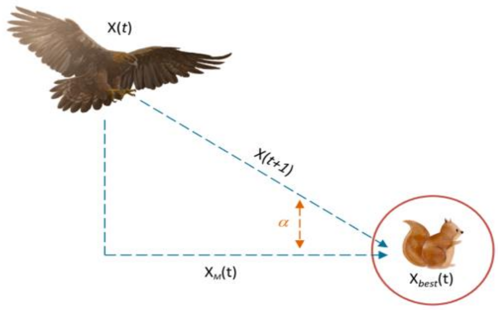

Fig. 4. Aquila demonstrates low flight behavior accompanied by slow descent attacks [44]

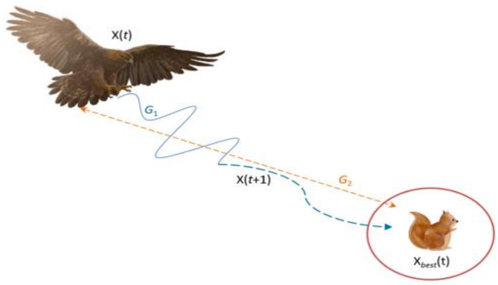

Fig. 5. Aquila exhibits the behavior of walking and grabbing prey [44]

The AO algorithm has eight stages [57]. The initial phase randomizes the parameters and population. The second step determines if the current iteration is greater than or equal to the maximum iteration. The final phase employs any cost function. In the fourth stage, the performance of each agent in the search space is evaluated. The fifth stage evaluates performance based on fitness values to determine the winner. The sixth stage modifies the parameters and average value. The seventh stage involves a condition check. If the condition is acceptable, it will fit again. The third and fourth methods will be utilized if the condition is false. The eighth stage of the Aquila optimization algorithm displays the optimal solution in four distinct ways.

AO can address problem cases that are continuous in nature; however, this article focuses on applying AO to discrete problem cases. The pseudocode of AO has been adapted to handle discrete problem cases by incorporating the Largest Ranked Value (LRV) approach. The LRV method is commonly employed in research to ascertain the maximum value of a given dataset or variable. This technique entails organizing the data in ascending order and selecting the highest value as the outcome.

The population value indicates the number of AOs in this article. The population value used in this research is referenced from several previous articles that employed the Aquila algorithm. For example, Sasmal *et al*. [34] conducted a study using a population of 50, while Lockwood and Cannon [36] used the largest population of 4000. Both studies demonstrated that the research with the largest population yielded more optimal results. Therefore, this research adopts a large population size, specifically 300 and 500 for each case. Additionally, the final criterion of this method involves iterations, specifically 500 and 1000 iterations. Increasing the number of iterations will lead to more optimal values.

2.2.2. Initialization

The AO algorithm's population-based approach begins with a population of potential solutions (X), as shown in equation (14), which are generated stochastically between the scheduling problem's upper bound (UB) and lower bound (LB). Each iteration selects as the approximate optimal solution the response that is by far the best.

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & x_{1,Dim-1} & x_{1,Dim} \\ x_{2,1} & \cdots & x_{2,j} & \cdots & x_{2,Dim} \\ \cdots \vdots & \cdots \vdots & x_{i,j} \vdots & \cdots \vdots & \cdots \vdots \\ x_{N-1,1} & \cdots & x_{N-1,j} & \cdots & x_{N-1,Dim} \\ x_{N,1} & \cdots & x_{N,j} & x_{N,Dim-1} & x_{N,Dim} \end{bmatrix} \quad (14)$$

Where X represents the current collection of potential answers that were chosen at random using the equation (15), X_i Represents the i solution's judgment value (position), N is the total potential solution (population), and Dim denotes the problem's dimensionality. Where a *rand* is a random number, LB_j Is the lower bound j, and UB_j is the upper bound of the scheduling problem.

$$X_{ij} = rand \times (UB_j - LB_j) + LB_j, \ i = 1,2,\ldots\ldots,N \quad j = 1,2,\ldots.Dim \quad (15)$$

In this instance, the Largest Ranked Value (LRV) criterion will be applied to each Aquila. Using LRV, the mapping of task permutations is efficient. Using the LRV rule, the first order of job permutation is determined by selecting the value with the highest weight. The second order will have the second-highest value [26].

2.2.3. Mathematical model

The AO algorithm simulates Aquila's hunting behavior by displaying the actions of each hunting step and the activities of each hunting ledge. High hovering with vertical bending to select the search space; contour flight with short gliding strikes to explore another search space; low flight with slow descending strikes to exploit a converging search space; and swooping by walking and capturing prey are the four methods used to represent the optimization process of the proposed AO algorithm. If $t \leq \left(\frac{2}{3}\right) * T$, the AO algorithm can use various behaviours to transition from the exploration phase to the exploitation step.

Step 1: Extended exploration (X_1)

In the first method, Aquila uses its high altitude and vertical bending to identify prey and choose the ideal hunting location. Here, the AO moves around to find its prey's place in the search region. It represents the equation (16).

$$X_1(t+1) = X_{best}(t) \times \left(1 - \frac{t}{T}\right) + (X_M(t) - X_{best}(t) * rand), \quad (16)$$

where $X_1(t+1)$ is the solution of the next iteration t, generated by the first search method (X_1). $X_{best}\ (t)$ is the best solution obtained by the solution up to the t iteration. It reflects the estimated prey spot. Equation $\left(\frac{1-t}{T}\right)$ Used to manage the number of repetitions in the extended search (exploration). $X_M(t)$ Represents the average location value of the current answer connected after t iterations using equation (17). A random number between 0 and 1 is called $rand$. T stands for the maximum amount of iterations, and t stands for the current iteration.

$$X_M(t) = \frac{1}{N}\sum_{t=1}^{N} X_i(t), \quad \forall j = 1,1,\ldots\ldots,Dim \quad (17)$$

where N is the number of potential answers, and *Dim* is the problem's dimension (population size).

Step 2: Narrowed exploration (X_2)

When a prey area is located using the second technique (X_2), the Aquila circles over the prey while preparing to land and then launches an attack. The term "contour flight with a short gliding attack" refers to this technique. In this case, the AO narrowly investigates a chosen region of the prey target in order to prepare for the assault. Equation (18) describes this action of Aquila mathematically.

$$X_2(t+1) = X_{best}(t) \times Levy(D) + X_R(t) + (y - x) * rand, \quad (18)$$

where $X_2(t+1)$ represents the answer of step t, produced by the second search technique (X_2).

http://dx.doi.org/10.30656/jsmi.v7i2.6446

Levy (D) is the flight levy distribution function computed using the equation, and *D* is the dimension space. It is described by equation (19).

$$Levy(D) = s \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \tag{19}$$

where u and v are random numbers between 0 and 1, s is a constant value set at 0.01, and these values are determined using the equation (20).

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sine\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2\left(\frac{\beta-1}{2}\right)} \right) \tag{20}$$

where β is a constant with a value of 1.5, the search is represented in equation (18) by a spiral shape that is computed using the variables y and x that are described by equation (21), (22), (23), (24), and (25).

$$y = r \times \cos(\theta) \tag{21}$$

$$x = r \times \sin(\theta) \tag{22}$$

$$r = r_1 + U \times D_1 \tag{23}$$

$$\theta = -\omega \times D_1 + \theta_1 \tag{24}$$

$$\theta_1 = \frac{3 \times \pi}{2} \tag{25}$$

Algorithm 1 : ***Pseudocode of Aquila Optimizer Algorithm***

```
Initialization phase:
Initialize the AO's X population.
Initialize the AO's parameters.
while (The last iteration is not satisfied) do
        Apply LRV to convert vector position Aquila to permutation job scheduling
        Determine the values for TEC in each AO
        Xbest(t)= Calculate the optimal solution based on TEC in population AO
            for (i=1,2...,N) do
            XM (t) mean value should be updated according to the current solution (t)
            Update the x, y, G1, G2, Levy(D), etc.
            if t ≤ (2/3) * T then
                if rand≤ 0:5 then
                    Phase 1: Expanded exploration (X1)
                    The present solution can be updated by using Equation (16).
                        if Fitness(X1(t+1)) < Fitness(X(t)) then
                        X(t) =(X1(t+1))
                            if Fitness(X1(t+1)) < Fitness(Xbest(t)) then
                            Xbest(t) =X1(t+1)
                            end if
                        end if
                else
                    Phase 2: Narrowed exploration (X2)
                    The present solution can be updated by using Equation (18)
                        if Fitness(X2(t+1)) < Fitness(X(t)) then
                        X(t) =(X2(t+1))
                            if Fitness(X2(t+1)) < Fitness(Xbest(t)) then
                            Xbest(t) =X2(t+1)
                            end if
                        end if
                end if
            else
                if rand≤ 0:5 then
                    Phase 3: Expanded exploitation (X3)
                    The present solution can be updated by using Equation (26)
                        if Fitness(X3(t+1)) < Fitness(X(t)) then
                        X(t) =(X3(t+1))
                            if Fitness(X3(t+1)) < Fitness(Xbest(t)) then
                            Xbest(t) =X3(t+1)
                            end if
                        end if
                else
                    Phase 4: Narrowed exploitation (X4)
                    The present solution can be updated by using Equation (27)
                        if Fitness(X4(t+1)) < Fitness(X(t)) then
                        X(t) =(X4(t+1))
                            if Fitness(X4(t+1)) < Fitness(Xbest(t)) then
                            Xbest(t) =X4(t+1)
                            end if
                        end if
                end if
            end if
            end for
    end while
    Return The Optimal solution (Xbest)
```

Step 3: Extended exploitation (X_3)

In the third method (X_3), the Aquila descends vertically with a preliminary strike to gauge the prey's response. After identifying the prey area, precise results have been obtained, and the Aquila is prepared to land and attack. It is referred to as a low flight with a gradual downward assault. A slow descending attack by AO mimics Aquila's low-flying behavior by taking advantage of the chosen target region. Equation (26) is a description of the algebraic behavior.

$$X_3(t+1) = \left(X_{best}(t) - X_M(t)\right) \times a - rand + \left((Ub - Lb) \times rand + Lb\right) \times \delta \quad (26)$$

where $X_3(t+1)$ is the solution of the next iteration *t*, generated by the third search method (X_3). $X_{best}\ (t)$ alludes to iteration *i* estimated prey location (the best solution obtained), and $X_M\ (t)$ shows the present solution's average value at the *t* iteration. Between 0 and 1, the number *rand* is random. α and δ is a valuable utilisation adjustment parameter (0,1). The letters LB and UB indicate the problem's lower and upper bounds, respectively.

Step 4: Narrowed exploitation (X_4)

In the fourth technique (X_4), the Aquila will attack its prey on the ground as it gets close to it using stochastic movements. It is known as "walking and grabbing prey." The AO now engages the prey at the previous position. Equation (27) shows that the algebraic description fits this behavior.

$$X_4(t+1) = QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times Levy(D) + rand\ \times G_1 \quad (27)$$

where $X_4(t+1)$ is the result of the fourth search technique (X_4) and represents the solution of the subsequent iteration t. *QF* indicates the quality function computed using the equation used to balance the search strategy (28). G_1 represents the AO's range of motion during mating passes, calculated using equation (29). G_2 indicates a value ranging from 2 to 0, showing the flight inclination of the AO used to track the prey during the elope from the starting point (1) to the finishing point. *(t)*, which is produced by equation (30).

$$QF(t) = t^{\frac{2\times rand()-1)}{(1-T)^2}} \quad (28)$$

$$G_1 = 2 \times rand() - 1 \quad (29)$$

$$G_2 = 2 \times (1 - \frac{t}{T}) \quad (30)$$

2.2.4. *Research data*

4 variations of job and machine data are presented in this study using process times derived from previous studies. The data and the job and machine combinations used for the study are presented in Table 2. For this study's Case 1 with a small category issue (Table 3), which consists of a problem with ten jobs and six machines, data from research Carlier [58]. Then, study data from research Reeves [59], there is a problem named Case 2 with a medium category that involves 30 jobs and ten machines (Appendix 1). Additionally, research Reeves [59] was used to compile the research data (Appendix 2), which includes Case 3, an issue with 50 jobs and 20 machines and a large category. Last, the study Heller [60] is where the information for a problem with 100 jobs, five machines, and a huge category (Appendix 3).

2.2.5. *Procedure*

The population used for each research parameter is 300 and 500. The iterations used for each population are 500 and 1000. The algorithms are compared to determine the optimal algorithm for minimizing energy consumption. Experimental calculations were performed with Matlab R2017b software on Windows 8 with Intel Celeron quad-core processor N22930. Each experiment recorded carbon emissions and computation time. This study benchmarks the solution quality of energy consumption to evaluate the proposed algorithm's solution quality. A big population and iteration are used to compare algorithms, specifically Population 500 x Iteration 500. All methods are solved using Matlab R2017b, running on an Intel Celeron quad-core processor N22930 under Windows 8. The effectiveness of the approach was evaluated using the independent sample t-test.

Table 2. Research data

Problem	Job and machine	Case category	Reference
Case 1	10 jobs, 6 machine	Small	Carlier [58]
Case 2	30 jobs, 10 machine	Medium	Reeves [59]
Case 3	50 jobs, 20 machine	Large	Reeves [59]
Case 4	100 job 5 machine	Huge	Heller [60]

http://dx.doi.org/10.30656/jsmi.v7i2.6446

Table 3. Research data case 1

Job	Machine 1	2	3	4	5	6
1	333	991	996	123	145	234
2	333	111	663	456	785	532
3	252	222	222	789	214	586
4	222	204	114	876	752	532
5	255	477	123	543	143	142
6	555	566	456	210	698	573
7	558	899	789	124	532	12
8	888	965	876	537	145	14
9	889	588	543	854	247	527
10	999	889	210	632	451	856
τ_j	0.4507	0.5582	0.3014	0.9409	0.7859	0.2709
φ_j	0.00663	0.0164	0.0048	0.1007	0.0871	0.00296

3. RESULTS AND DISCUSSION

Table 4 displays the population comparison and iteration trial outcomes using the AO and GWO algorithms. According to the experimental findings, iteration 500 and population 300 performed best in case 1. In Case 2, 1000 iterations and 500 populations yield the greatest results. In Case 3, iteration 1000 and population 300 yield the greatest results. In Case 4, iteration 1000 and population 300 yield the greatest results. The experimental findings also demonstrate that the TEC decreases with increasing population and iteration.

On the other hand, if the number and iterations decrease, the resulting TEC increases. The parameters used in this problem are the number of iterations and the population of each case. Using population parameters 300 and 500, as well as iterations 500 and 1000, will be analyzed in the case of small jobs, and large jobs will be optimal when using which parameters.

Table 5 displays the output of TEC using the AO and GWO methods for 30 replications. Cases 1, 2, 3, and 4 demonstrate that TEC has distinct values based on the results. The data distribution for each instance is known as boxplot (Fig. 6, Fig. 7, Fig. 8, and Fig. 9). A boxplot, also referred to as a box-and-whisker plot, is a visual representation of numerical data that illustrates the dataset's median, quartiles, and range. The box portion of the plot depicts the interquartile range (IQR), which represents the range between the first quartile (Q1) and the third quartile (Q3). The line inside the box corresponds to the median, which denotes the middle value of the dataset. The whiskers extend from the box to the minimum and maximum values of the dataset, excluding any outliers. Outliers refer to data points located outside the whiskers of the box plot. Boxplots are valuable tools for identifying a dataset's dispersion, skewness, and outliers. They are frequently employed in statistical analysis and data

Table 4. Result of population and iteration on energy consumption

Population	Case	Job	Machine	Iteration 500 TEC AO	500 TEC GWO	1000 TEC AO	1000 TEC GWO
300	1	10	6	16868.961	17076	16869.097	17055
	2	30	10	7617.104	7662	7618.080	7659
	3	50	10	11995.27	12056	11994.97	12054
	4	100	10	2071.113	2073.5569	2070.696	2073.4747
500	1	10	6	16869.097	17055	16868.961	17143
	2	30	10	7616.920	7654	7614.161	7662
	3	50	10	11995.307	12053	11995.307	12061
	4	100	10	2070.696	2073.5158	2070.819	2073.7624

visualization to compare multiple data groups and identify patterns and trends. Boxplots can be drawn either vertically or horizontally, and they can be customized to include additional information, such as mean values or confidence intervals. The spread of the AO data is smaller than the GWO distribution (Fig. 6, Fig. 7, and Fig. 8). The comparison of the data distribution for AO and GWO (Fig. 9), shows the AO algorithm has broader data distribution than GWO.

Table 5. Energy consumption results of AO and GWO algorithms

Replication	Case 1		Case 2		Case 3		Case 4	
	TEC AO	TEC GWO	TEC AO	TEC GWO	TEC AO	TEC GWO	TEC AO	TEC GWO
1	16871.50	17054.63	7615.98	7659.51	11994.42	12059.67	2071.03	2073.39
2	16868.96	17143.38	7618.55	7657.34	11994.33	12059.52	2071.02	2073.64
3	16888.65	17054.63	7612.02	7658.56	11993.22	12058.03	2071.18	2073.39
4	16868.96	17143.38	7622.61	7661.77	11993.76	12058.03	2070.92	2073.60
5	16869.10	17054.63	7612.64	7658.04	11995.72	12053.84	2070.92	2073.27
6	16867.14	17076.05	7614.55	7658.99	11994.30	12058.46	2071.03	2073.27
7	16869.17	17054.63	7618.48	7658.04	11992.82	12058.46	2071.02	2073.39
8	16898.25	17054.63	7613.34	7657.17	11994.45	12052.92	2071.18	2073.39
9	16895.01	17143.38	7613.31	7655.09	11995.05	12054.34	2070.91	2073.52
10	16872.99	17054.63	7614.95	7657.52	11995.17	12057.47	2070.92	2073.56
11	16908.51	17076.05	7614.95	7660.12	11994.85	12052.50	2070.92	2073.35
12	16891.19	17076.05	7616.46	7664.20	11994.86	12059.45	2070.92	2073.19
13	16888.65	17054.63	7615.80	7658.56	11995.81	12055.83	2071.03	2073.27
14	16888.65	17054.63	7613.99	7660.12	11994.00	12057.47	2071.02	2073.39
15	16868.96	17076.05	7615.20	7655.43	11995.14	12060.59	2071.18	2073.93
16	16869.10	17054.63	7613.04	7656.91	11994.69	12057.11	2070.91	2073.52
17	16871.50	17054.63	7613.44	7658.04	11994.81	12056.68	2070.92	2073.35
18	16867.14	17054.63	7618.63	7653.79	11995.86	12059.31	2070.92	2073.56
19	16868.96	17054.63	7616.07	7653.79	11993.99	12056.76	2070.92	2073.35
20	16895.55	17054.63	7615.05	7660.47	11996.08	12055.90	2071.03	2073.39
21	16911.78	17076.05	7614.50	7656.74	11997.31	12052.28	2071.02	2073.19
22	16888.65	17076.05	7617.77	7659.43	11994.97	12059.60	2071.36	2073.76
23	16869.10	17076.05	7615.17	7657.52	11993.10	12058.32	2071.43	2073.76
24	16868.85	17054.63	7615.75	7660.30	11994.60	12059.81	2071.04	2073.31
25	16868.96	17143.38	7615.11	7661.42	11992.91	12053.13	2071.16	2073.23
26	16869.17	17054.63	7614.91	7664.37	11994.54	12055.26	2070.89	2073.23
27	16869.10	17054.63	7614.91	7661.86	11993.34	12052.00	2071.03	2073.64
28	16871.50	17054.63	7614.01	7664.81	11993.53	12060.80	2071.06	2073.52
29	16869.10	17076.05	7613.94	7659.25	11993.53	12052.57	2071.13	2073.56
30	16869.10	17054.63	7615.45	7658.73	11993.53	12057.11	2071.03	2073.39

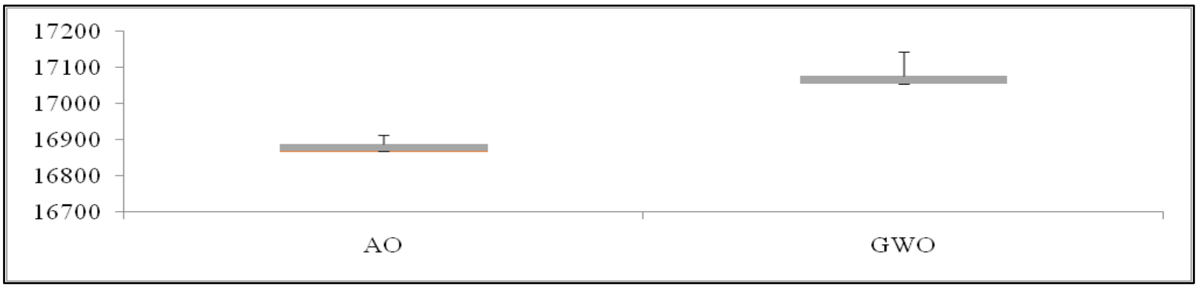

Fig. 6. Boxplot case 1

 http://dx.doi.org/10.30656/jsmi.v7i2.6446

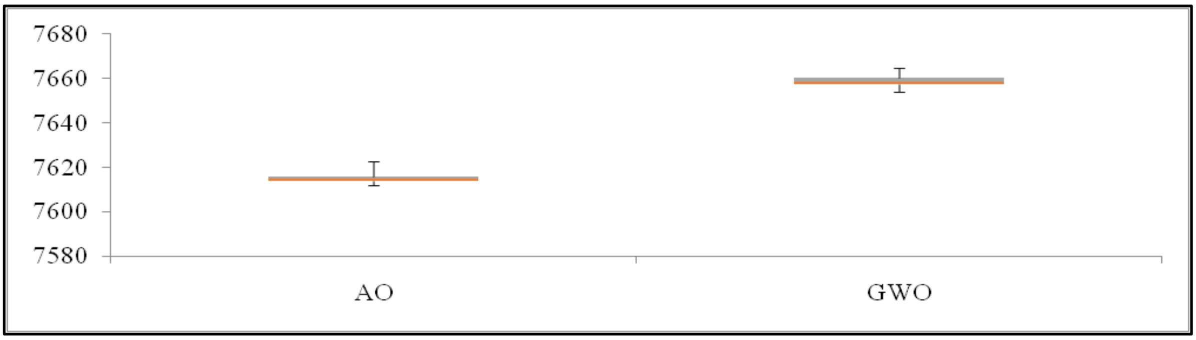

Fig. 7. Boxplot case 2

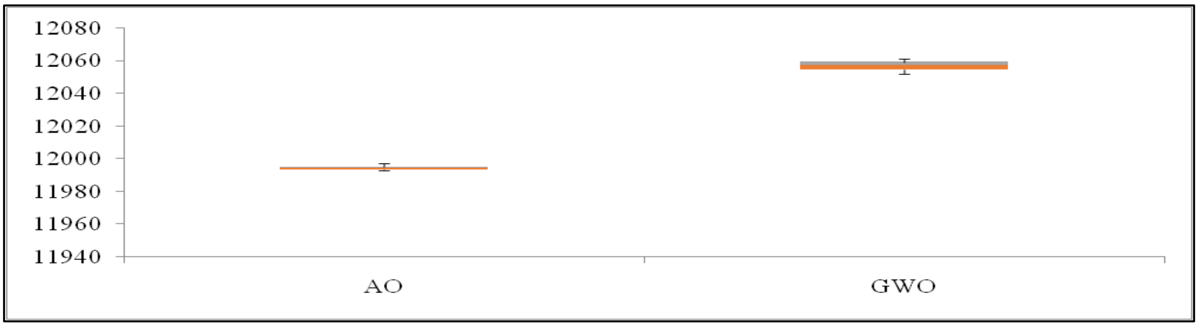

Fig. 8. Boxplot case 3

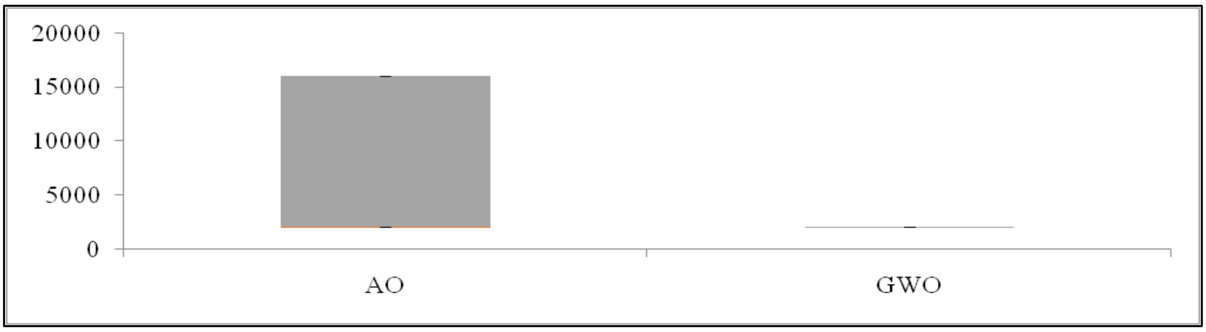

Fig. 9. Boxplot case 4

In each of the four cases, the results of energy optimization are compared to the average test results. The test was conducted with the support of SPSS and the Independent sample t-test. Table 6 summarizes the results of the comparison test. In four distinct situations, the AO algorithm consumes less energy on average than the GWO algorithm. This result is evident given that the Sig 2-tailed is <0.05.

The comparison results of computation time between AO and GWO (Table 7), reveal that AO's computation time has a smaller value than GWO's. Thus, AO exhibits a faster computation time than GWO. The proposed AO algorithm yields an optimal computation time.

Energy Consumption Ratio (ECR) is also used to evaluate the performance of the algorithms. The Energy Consumption Ratio (ECR) is defined as the Energy Consumption (EC) of the proposed AO algorithm divided by various algorithms, that is, GWO (Equation 31). If ECR 1, the AO algorithm has a higher performance than other algorithms, but if ECR = 1, it has the same performance as different algorithms. In addition, the other algorithm performs better if ECR > 1.

$$ECR = \frac{EC\ proposed\ algorithm}{EC\ other\ algorithm} \quad (31)$$

A comparison of the algorithm energy consumption and the ratio of the AO algorithms between GWO algorithms was made and presented in Table 8. The ECR results show that the average GWO values are significant at 0.994 and 1 for AO.

Table 6. Results of energy consumption with independent sample t-test

	Case1		Case2		Case3		Case4	
	AO	GWO	AO	GWO	AO	GWO	AO	GWO
Mean	16878.11	17072.18	7615.35	7658.93	11994.49	12056.77	2071.04	2073.44
Std. Deviation	13.413	29.908	2.160	2.778	1.042	2.726	0.132	0.183
t	0.285		0.282		0.316		0.281	
Sig.(2-tailed)	0.000		0.000		0.000		0.000	

Table 7. Comparison of computation time

Population	Case	Job	Machine	Computation time			
				500		1000	
				CT AO	CT GWO	CT AO	CT GWO
300	1	10	6	47095	94190	47096	94191
	2	30	10	701522	1403044	701523	1403045
	3	50	10	840223	1680446	840224	1680445
	4	100	10	978933	1957866	978934	1957867
500	1	10	6	47083	94191	47084	94192
	2	30	10	701511	1403045	701512	1403046
	3	50	10	840212	1680447	840213	1680448
	4	100	10	978922	1957867	978923	1957868

Table 8. Comparison of the algorithms for energy consumption and ECR

Population	Case	Job	Machine	Iteration				ECR			
				500		1000		500		1000	
				TEC AO	TEC GWO	TEC AO	TEC GWO	AO	GWO	AO	GWO
300	1	10	6	16868.961	17076	16869.097	17055	1	0.988	1	0.989
	2	30	10	7617.104	7662	7618.080	7659	1	0.994	1	0.995
	3	50	10	11995.27	12056	11994.97	12054	1	0.995	1	0.995
	4	100	10	2071.113	2073.5569	2070.696	2073.4747	1	0.999	1	0.999
500	1	10	6	16869.097	17055	16868.961	17143	1	0.989	1	0.984
	2	30	10	7616.920	7654	7614.161	7662	1	0.995	1	0.994
	3	50	10	11995.307	12053	11995.307	12061	1	0.995	1	0.995
	4	100	10	2070.696	2073.5158	2070.819	2073.7624	1	0.999	1	0.999
Average energy consumption rasio								**1**	**0.994**	**1**	**0.994**

Table 9. Comparison with gurobi

Population	Case	Job	Machine	Iteration				Gurobi
				500		1000		
				TEC AO	TEC GWO	TEC AO	TEC GWO	Obj Value
300	1	10	6	16868.961	17076	16869.097	17055	8969
	2	30	10	7617.104	7662	7618.080	7659	-
	3	50	10	11995.27	12056	11994.97	12054	-
	4	100	10	2071.113	2073.5569	2070.696	2073.4747	-
500	1	10	6	16869.097	17055	16868.961	17143	9684
	2	30	10	7616.920	7654	7614.161	7662	-
	3	50	10	11995.307	12053	11995.307	12061	-
	4	100	10	2070.696	2073.5158	2070.819	2073.7624	-

http://dx.doi.org/10.30656/jsmi.v7i2.6446

This research was also conducted using the exact method, namely Gurobi (Table 9). Based on the research results, Gurobi can only be applied to case 1 because it can only run for 1 hour, while case 2 to case 4 require more than 1 hour, making this method inapplicable. Comparing the results, Gurobi's values are closer to AO's, whereas GWO has higher values. Therefore, the AO method is more optimal.

This research contributes to developing a new AO procedure to reduce energy consumption in no-idle flow shop scheduling problems. Based on the literature review, this study is the first to propose the AO algorithm to solve the no-idle flow shop scheduling issue. This study suggests the LRV procedure in the proposed algorithm to convert the Aquila position to the schedule sequence. This study also tests the population and iteration parameters of the AO algorithm to solve the energy consumption minimization issue in the no-idle flow shop scheduling problem. Four experimental cases yielded diverse energy consumption optimizations due to their practical outcomes. The population and iteration parameters recommended for optimization in the case of a small number of jobs are small, according to the findings of this study. In the case of a large number of jobs, on the other hand, the population and iteration parameters are set to use a large population and iteration. Moreover, the resulting energy consumption is reduced when the population and number of iterations are increased in large cases. Therefore, decision-makers can utilize the proper iteration and population parameters for no-idle flow shop scheduling. The comparison between the AO and GWO algorithms demonstrates that the proposed algorithm reduces energy usage. The independent sample t-test supports this conclusion. Therefore, the proposed algorithm can resolve the no-idle flow shop scheduling issue.

4. CONCLUSION

This article discusses the challenges and pressures manufacturing industries face due to rising energy consumption caused by production activities. This study proposes the Aquila Optimizer (AO) algorithm to reduce energy usage. By presenting a new procedure AO and expanding the literature on metaheuristic algorithms that can solve energy consumption problems with NIPFSP, this study contributes to a greater understanding of solving energy consumption problems with NIPFSP. It is preferable to use small populations and iterations for small tasks. In contrast, using significant populations and iterations for large jobs is preferable.

Additionally, AO is compared to several other procedures. Computational investigations demonstrate that AO maximizes energy efficiency. Several aspects of this research can be investigated in future studies. This study suggests that AO can be a starting point for other metaheuristic algorithms. In conclusion, the proposed AO algorithm can be used to minimize energy consumption. Consequently, the suggestion for future research is to develop the AO algorithm by integrating other procedures to solve more complicated scheduling issues.

REFERENCES

[1] K. Fang, N. Uhan, F. Zhao, and J. W. Sutherland, ‘A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction’, *J. Manuf. Syst.*, vol. 30, no. 4, pp. 234–240, 2011, doi: 10.1016/j.jmsy.2011.08.004.

[2] Y. Seow and S. Rahimifard, ‘A framework for modelling energy consumption within manufacturing systems’, *CIRP J. Manuf. Sci. Technol.*, vol. 4, no. 3, pp. 258–264, 2011, doi: 10.1016/j.cirpj.2011.03.007.

[3] G. Gong *et al.*, ‘Energy-efficient flexible flow shop scheduling with worker flexibility’, *Expert Syst. Appl.*, vol. 141, p. 112902, Mar. 2020, doi: 10.1016/j.eswa.2019.112902.

[4] G. Mouzon, M. B. Yildirim, and J. Twomey, ‘Operational methods for minimization of energy consumption of manufacturing equipment’, *Int. J. Prod. Res.*, vol. 45, no. 18–19, pp. 4247–4271, Sep. 2007, doi: 10.1080/00207540701450013.

[5] D. M. Utama, A. A. P. Salim, and D. S. Widodo, ‘A novel hybrid archimedes optimization algorithm for energy-efficient hybrid flow shop scheduling’, *Int. J. Adv. Intell. Informatics*, vol. 8, no. 2, pp. 237–250, 2022, doi: 10.26555/ijain.v8i2.724.

[6] D. M. Utama, D. S. Widodo, M. F. Ibrahim, and S. K. Dewi, ‘A New Hybrid Butterfly Optimization Algorithm for Green Vehicle Routing Problem’, *J. Adv. Transp.*, vol. 2020, pp. 1–14, Dec. 2020,

doi: 10.1155/2020/8834502.

[7] H. Öztop, M. F. Tasgetiren, D. T. Eliiyi, Q.-K. Pan, and L. Kandiller, 'An energy-efficient permutation flowshop scheduling problem', *Expert Syst. Appl.*, vol. 150, p. 113279, Jul. 2020, doi: 10.1016/j.eswa.2020.113279.

[8] M. Dai, D. Tang, A. Giret, M. A. Salido, and W. D. Li, 'Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm', *Robot. Comput. Integr. Manuf.*, vol. 29, no. 5, pp. 418–429, 2013, doi: 10.1016/j.rcim.2013.04.001.

[9] A. Dietmair and A. Verl, 'A generic energy consumption model for decision making and energy efficiency optimisation in manufacturing', *Int. J. Sustain. Eng.*, vol. 2, no. 2, pp. 123–133, 2009, doi: 10.1080/19397030902947041.

[10] S. Li, F. Liu, and X. Zhou, 'Multi-objective energy-saving scheduling for a permutation flow line', *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.*, vol. 232, no. 5, pp. 879–888, 2018, doi: 10.1177/0954405416657583.

[11] S. Ene, İ. Küçükoğlu, A. Aksoy, and N. Öztürk, 'A genetic algorithm for minimizing energy consumption in warehouses', *Energy*, vol. 114, pp. 973–980, 2016, doi: 10.1016/j.energy.2016.08.045.

[12] C. W. Park *et al.*, 'Energy consumption reduction technology in manufacturing - A selective review of policies, standards, and research', *Int. J. Precis. Eng. Manuf.*, vol. 10, no. 5, pp. 151–173, 2009, doi: 10.1007/s12541-009-0107-z.

[13] X. Liu, L. Wang, L. Kong, F. Li, and J. Li, 'A Hybrid Genetic Algorithm for Minimizing Energy Consumption in Flow Shops Considering Ultra-low Idle State', *Procedia CIRP*, vol. 80, pp. 192–196, 2019, doi: 10.1016/j.procir.2018.12.013.

[14] M. Ross, 'Efficient energy use in manufacturing.', *Proc. Natl. Acad. Sci.*, vol. 89, no. 3, pp. 827–831, Feb. 1992, doi: 10.1073/pnas.89.3.827.

[15] M. S. Nagano and J. V. Moccellin, 'A new constructive heuristic method for minimizing makespan in permutation flow shop scheduling', in *Encontro Nacional de Engenharia de Produção*, 2012, pp. 10–18, [Online]. Available: http://www.din.uem.br/sbpo/sbpo2012/pdf/arq0162.pdf.

[16] V. Riahi, R. Chiong, and Y. Zhang, 'A new iterated greedy algorithm for no-idle permutation flowshop scheduling with the total tardiness criterion', *Comput. Oper. Res.*, vol. 117, p. 104839, May 2020, doi: 10.1016/j.cor.2019.104839.

[17] H. Harbaoui and S. Khalfallah, 'Tabu-search optimization approach for no-wait hybrid flow-shop scheduling with dedicated machines', *Procedia Comput. Sci.*, vol. 176, pp. 706–712, 2020, doi: 10.1016/j.procs.2020.09.043.

[18] K.-C. Ying, S.-W. Lin, C.-Y. Cheng, and C.-D. He, 'Iterated reference greedy algorithm for solving distributed no-idle permutation flowshop scheduling problems', *Comput. Ind. Eng.*, vol. 110, pp. 413–423, Aug. 2017, doi: 10.1016/j.cie.2017.06.025.

[19] Y. Zhou, H. Chen, and G. Zhou, 'Invasive weed optimization algorithm for optimization no-idle flow shop scheduling problem', *Neurocomputing*, vol. 137, pp. 285–292, Aug. 2014, doi: 10.1016/j.neucom.2013.05.063.

[20] W. Shao, D. Pi, and Z. Shao, 'Memetic algorithm with node and edge histogram for no-idle flow shop scheduling problem to minimize the makespan criterion', *Appl. Soft Comput.*, vol. 54, pp. 164–182, May 2017, doi: 10.1016/j.asoc.2017.01.017.

[21] J. Chen, L. Wang, and Z. Peng, 'A collaborative optimization algorithm for energy-efficient multi-objective distributed no-idle flow-shop scheduling', *Swarm Evol. Comput.*, vol. 50, no. April, p. 100557, Nov. 2019, doi: 10.1016/j.swevo.2019.100557.

[22] Q. K. Pan and L. Wang, 'A novel differential evolution algorithm for no-idle permutation flow-shop scheduling problems', *Eur. J. Ind. Eng.*, vol. 2, no. 3, pp. 279–297, 2008, doi: 10.1504/EJIE.2008.017687.

[23] M. Fatih Tasgetiren, Q. K. Pan, P. N. Suganthan, and A. Oner, 'A discrete

artificial bee colony algorithm for the no-idle permutation flowshop scheduling problem with the total tardiness criterion', *Appl. Math. Model.*, vol. 37, no. 10–11, pp. 6758–6779, 2013, doi: 10.1016/j.apm.2013.02.011.

[24] Q. K. Pan and L. Wang, 'No-idle permutation flow shop scheduling based on a hybrid discrete particle swarm optimization algorithm', *Int. J. Adv. Manuf. Technol.*, vol. 39, no. 7–8, pp. 796–807, 2008, doi: 10.1007/s00170-007-1252-0.

[25] G. Deng and X. Gu, 'A hybrid discrete differential evolution algorithm for the no-idle permutation flow shop scheduling problem with makespan criterion', *Comput. Oper. Res.*, vol. 39, no. 9, pp. 2152–2160, 2012, doi: 10.1016/j.cor.2011.10.024.

[26] D. M. Utama, T. Baroto, and D. S. Widodo, 'Energy-Efficient Flow Shop Scheduling Using Hybrid Grasshopper Algorithm Optimization', *J. Ilm. Tek. Ind.*, vol. 19, no. 1, pp. 30–38, 2020, doi: 10.23917/jiti.v19i1.10079.

[27] D. S. Widodo and D. M. Utama, 'The Hybrid Ant Lion Optimization Flow Shop Scheduling Problem for Minimizing Completion Time', *J. Phys. Conf. Ser.*, vol. 1569, no. 2, p. 022097, Jul. 2020, doi: 10.1088/1742-6596/1569/2/022097.

[28] C. N. Al-Imron, D. M. Utama, and S. K. Dewi, 'An Energy-Efficient No Idle Permutations Flow Shop Scheduling Problem Using Grey Wolf Optimizer Algorithm', *J. Ilm. Tek. Ind.*, vol. 21, no. 1, pp. 1–10, Jun. 2022, doi: 10.23917/jiti.v21i1.17634.

[29] A. P. Rifai, P. A. Kusumastuti, S. T. W. Mara, R. Norcahy, and S. Z. Dawal, 'Multi-operator hybrid genetic algorithm-simulated annealing for reentrant permutation flow-shop scheduling', *ASEAN Eng. J.*, vol. 11, no. 3, pp. 109–126, 2021, doi: 10.11113/AEJ.V11.16875.

[30] A. P. Rifai, S. T. W. Mara, and A. Sudiarso, 'Multi-objective distributed reentrant permutation flow shop scheduling with sequence-dependent setup time', *Expert Syst. Appl.*, vol. 183, no. June, p. 115339, Nov. 2021, doi: 10.1016/j.eswa.2021.115339.

[31] W. Meethom and N. Koohathongsumrit, 'A decision support system for road freight transportation route selection with new fuzzy numbers', *foresight*, vol. 22, no. 4, pp. 505–527, Aug. 2020, doi: 10.1108/FS-10-2019-0090.

[32] D. M. Utama, M. D. Primayesti, S. Z. Umamy, B. M. N. Kholifa, and A. D. Yasa, 'A systematic literature review on energy-efficient hybrid flow shop scheduling', *Cogent Eng.*, vol. 10, no. 1, p. 2206074, Dec. 2023, doi: 10.1080/23311916.2023.2206074.

[33] W. Meethom and N. Koohathongsumrit, 'Design of Decision Support System for Road Freight Transportation Routing Using Multilayer Zero One Goal Programming', *Eng. J.*, vol. 22, no. 6, pp. 185–205, Dec. 2018, doi: 10.4186/ej.2018.22.6.185.

[34] B. Sasmal, A. G. Hussien, A. Das, and K. G. Dhal, 'A Comprehensive Survey on Aquila Optimizer', *Arch. Comput. Methods Eng.*, vol. 30, no. 7, pp. 4449–4476, Sep. 2023, doi: 10.1007/s11831-023-09945-6.

[35] Q. Fan, H. Huang, Y. Li, Z. Han, Y. Hu, and D. Huang, 'Beetle antenna strategy based grey wolf optimization', *Expert Syst. Appl.*, vol. 165, no. May 2020, p. 113882, Mar. 2021, doi: 10.1016/j.eswa.2020.113882.

[36] G. W. Lockwood and D. E. Cannon, 'Production Forecasting', in *Harness Oil and Gas Big Data with Analytics*, Wiley, 2014, pp. 213–242, doi: 10.1002/9781118910948.ch7.

[37] S. Wang, H. Jia, Q. Liu, and R. Zheng, 'An improved hybrid Aquila Optimizer and Harris Hawks Optimization for global optimization', *Math. Biosci. Eng.*, vol. 18, no. 6, pp. 7076–7109, 2021, doi: 10.3934/mbe.2021352.

[38] A. Fatani, A. Dahou, M. A. A. Al-qaness, S. Lu, and M. A. Abd Elaziz, 'Advanced Feature Extraction and Selection Approach Using Deep Learning and Aquila Optimizer for IoT Intrusion Detection System', *Sensors*, vol. 22, no. 1, p. 140, Dec. 2021, doi: 10.3390/s22010140.

[39] S. Wang, H. Jia, L. Abualigah, Q. Liu, and R. Zheng, 'An Improved Hybrid Aquila Optimizer and Harris Hawks Algorithm for Solving Industrial Engineering Optimization Problems', *Processes*, vol. 9, no. 9, p. 1551, Aug. 2021, doi: 10.3390/pr9091551.

[40] M. Abd Elaziz, A. Dahou, N. A. Alsaleh, A. H. Elsheikh, A. I. Saba, and M. Ahmadein, 'Boosting covid-19 image classification using mobilenetv3 and aquila optimizer algorithm', *Entropy*, vol. 23, no. 11, p. 1383, Oct. 2021, doi: 10.3390/e23111383.

[41] H. Yu, H. Jia, J. Zhou, and A. G. Hussien, 'Enhanced Aquila optimizer algorithm for global optimization and constrained engineering problems', *Math. Biosci. Eng.*, vol. 19, no. 12, pp. 14173–14211, 2022, doi: 10.3934/mbe.2022660.

[42] W. Aribowo, S. Supari, and B. Suprianto, 'Optimization of PID parameters for controlling DC motor based on the aquila optimizer algorithm', *Int. J. Power Electron. Drive Syst.*, vol. 13, no. 1, pp. 216–222, Mar. 2022, doi: 10.11591/ijpeds.v13.i1.pp216-222.

[43] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, and L. Abualigah, 'Binary Aquila Optimizer for Selecting Effective Features from Medical Data: A COVID-19 Case Study', *Mathematics*, vol. 10, no. 11, pp. 1–24, Jun. 2022, doi: 10.3390/math10111929.

[44] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. A. Al-qaness, and A. H. Gandomi, 'Aquila Optimizer: A novel meta-heuristic optimization algorithm', *Comput. Ind. Eng.*, vol. 157, no. February, p. 107250, Jul. 2021, doi: 10.1016/j.cie.2021.107250.

[45] D. Baraz and G. Mosheiov, 'A note on a greedy heuristic for flow-shop makespan minimization with no machine idle-time', *Eur. J. Oper. Res.*, vol. 184, no. 2, pp. 810–813, 2008, doi: 10.1016/j.ejor.2006.11.025.

[46] W.-J. Ren, Quan-Ke Pan, Jun-Qing Li, and J. J. Liang, 'Tabu search algorithm for solving No-idle permutation Flow Shop Scheduling Problem', in *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, Sep. 2010, no. 2, pp. 449–453, doi: 10.1109/BICTA.2010.5645167.

[47] W.-J. Ren, J.-H. Duan, F. Zhang, H. Han, and M. Zhang, 'Hybrid Tabu Search Algorithm for bi-criteria No-idle permutation flow shop scheduling problem', in *2011 Chinese Control and Decision Conference (CCDC)*, May 2011, pp. 1699–1702, doi: 10.1109/CCDC.2011.5968469.

[48] M. F. Tasgetiren, Q. K. Pan, P. N. Suganthan, and T. Jin Chua, 'A differential evolution algorithm for the no-idle flowshop scheduling problem with total tardiness criterion', *Int. J. Prod. Res.*, vol. 49, no. 16, pp. 5033–5050, 2011, doi: 10.1080/00207543.2010.497781.

[49] M. F. Tasgetiren, O. Buyukdagli, Q.-K. Pan, and P. N. Suganthan, 'A General Variable Neighborhood Search Algorithm for the No-Idle Permutation Flowshop Scheduling Problem', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8297 LNCS, no. PART 1, 2013, pp. 24–34, doi: 10.1007/978-3-319-03753-0_3.

[50] Y. Wang and X. Li, 'Shuffled Frog Leaping Algorithm for a Bi-objective No-Idle Permutation Flow Shop', in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 2013, pp. 2808–2813, doi: 10.1109/SMC.2013.479.

[51] C. Ling-Fang, W. Ling, and W. Jing-jing, 'A Two-Stage Memetic Algorithm for Distributed No-Idle Permutation Flowshop Scheduling Problem', in *2018 37th Chinese Control Conference (CCC)*, Jul. 2018, vol. 2018-July, no. 1, pp. 2278–2283, doi: 10.23919/ChiCC.2018.8483752.

[52] M. Fatih Tasgetiren, H. Öztop, L. Gao, Q. K. Pan, and X. Li, 'A Variable Iterated Local Search Algorithm for Energy-Efficient No-idle Flowshop Scheduling Problem', *Procedia Manuf.*, vol. 39, no. 2019, pp. 1185–1193, 2019, doi: 10.1016/j.promfg.2020.01.351.

[53] Y.-Z. Li, Q.-K. Pan, J.-Q. Li, L. Gao, and M. F. Tasgetiren, 'An Adaptive Iterated

Greedy algorithm for distributed mixed no-idle permutation flowshop scheduling problems', *Swarm Evol. Comput.*, vol. 63, no. March, p. 100874, Jun. 2021, doi: 10.1016/j.swevo.2021.100874.

[54] C.-Y. Cheng, S.-W. Lin, P. Pourhejazy, K.-C. Ying, and Y.-Z. Lin, 'No-Idle Flowshop Scheduling for Energy-Efficient Production: An Improved Optimization Framework', *Mathematics*, vol. 9, no. 12, pp. 1–18, Jun. 2021, doi: 10.3390/math9121335.

[55] R. Ruiz, E. Vallada, and C. Fernández-Martínez, 'Scheduling in Flowshops with No-Idle Machines', in *Studies in Computational Intelligence*, vol. 230, 2009, pp. 21–51, doi: 10.1007/978-3-642-02836-6_2.

[56] D. M. Utama, D. S. Widodo, W. Wicaksono, and L. R. Ardiansyah, 'A new hybrid metaheuristics algorithm for minimizing energy consumption in the flow shop scheduling problem', *Int. J. Technol.*, vol. 10, no. 2, pp. 320–331, 2019, doi: 10.14716/ijtech.v10i2.2194.

[57] D. M. Utama and M. D. Primayesti, 'A novel hybrid Aquila optimizer for energy-efficient hybrid flow shop scheduling', *Results Control Optim.*, vol. 9, no. 246, p. 100177, Dec. 2022, doi: 10.1016/j.rico.2022.100177.

[58] J. Carlier, 'Ordonnancements à contraintes disjonctives', *RAIRO - Oper. Res.*, vol. 12, no. 4, pp. 333–350, 1978, doi: 10.1051/ro/1978120403331.

[59] C. R. Reeves, 'A genetic algorithm for flowshop sequencing', *Comput. Oper. Res.*, vol. 22, no. 1, pp. 5–13, Jan. 1995, doi: 10.1016/0305-0548(93)E0014-K.

[60] J. Heller, ' Some Numerical Experiments for an M × J Flow Shop and its Decision-Theoretical Aspects ', *Oper. Res.*, vol. 8, no. 2, pp. 178–184, 1960, doi: 10.1287/opre.8.2.178.

Appendix 1. Research data case 2

Job	Machine									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	12	89	69	94	22	31	22	95	36	38
2	55	2	94	19	43	74	72	10	99	16
3	34	78	27	40	21	17	15	62	96	63
4	100	14	23	42	52	18	29	70	21	47
5	41	92	88	52	99	41	68	22	57	66
6	24	62	19	35	24	49	100	65	13	41
7	29	34	38	72	29	83	44	91	65	100
8	89	62	32	54	93	59	8	24	86	66
9	23	34	89	66	10	48	27	11	94	45
10	81	57	35	78	23	66	1	3	77	14
11	72	47	75	27	66	64	30	49	42	7
12	10	15	26	12	98	12	53	81	46	3
13	47	54	58	73	44	87	87	98	34	15
14	13	27	29	85	29	64	62	62	79	74
15	49	57	24	44	18	97	59	75	17	22
16	50	11	93	53	52	13	51	76	87	95
17	99	87	85	9	87	98	34	22	66	11
18	7	8	90	95	29	79	70	79	6	58
19	48	100	74	60	74	19	21	6	77	84
20	96	86	19	15	45	1	90	49	98	80
21	68	73	55	13	28	16	57	20	76	71
22	53	38	4	43	11	49	12	91	47	3
23	57	13	12	12	21	68	2	80	9	28
24	37	79	92	35	63	13	58	36	65	94
25	39	49	57	23	53	80	42	29	52	33
26	36	54	59	69	62	12	77	37	87	47
27	63	35	26	38	47	82	89	34	1	93
28	60	28	1	51	94	86	42	75	76	77
29	2	51	79	74	51	28	78	87	81	35
30	45	94	42	9	70	4	52	54	16	27
τ_j	0.2069	0.4754	0.9815	0.3853	0.1825	0.5439	0.2389	0.5302	0.6656	0.794
φ_j	0.0062	0.0096	0.0236	0.0079	0.0048	0.0070	0.0033	0.0089	0.0080	0.0075

http://dx.doi.org/10.30656/jsmi.v7i2.6446

Appendix 2. Research data case 3

Job	Machine 1	2	3	4	5	6	7	8	9	10
1	100	72	76	100	16	9	5	87	34	15
2	19	3	19	68	29	22	16	13	87	70
3	70	56	39	71	29	91	100	86	88	99
4	50	93	100	71	84	64	67	29	28	81
5	80	97	3	10	14	32	92	67	72	68
6	47	59	29	3	26	20	50	26	1	70
7	40	63	69	21	56	73	56	10	46	40
8	84	80	68	82	4	45	100	96	29	67
9	85	46	59	35	68	84	89	18	97	58
10	60	60	2	50	90	20	78	56	62	27
11	78	64	21	5	85	55	15	23	36	87
12	98	31	42	73	83	48	71	49	72	30
13	4	57	30	11	67	4	82	77	98	21
14	45	45	25	45	7	59	88	42	57	81
15	73	94	83	59	1	72	65	62	45	76
16	77	84	11	82	10	9	67	27	43	8
17	22	66	5	77	97	28	61	82	62	96
18	90	51	87	27	65	76	67	20	75	67
19	12	92	43	21	92	64	94	67	60	46
20	9	76	62	46	71	65	76	65	30	38
21	29	12	71	70	46	96	12	70	76	19
22	83	15	73	32	51	6	3	29	3	24
23	83	95	87	29	46	67	89	73	69	33
24	83	46	82	2	55	54	85	3	20	57
25	11	32	15	27	2	43	23	79	28	29
26	10	74	73	99	54	89	83	5	28	90
27	73	40	4	20	51	18	37	18	61	75
28	85	30	58	89	48	15	82	77	2	3
29	56	63	26	87	53	8	80	46	5	62
30	59	67	73	65	60	61	94	86	38	1
31	70	66	80	32	93	56	26	41	21	9
32	4	66	79	43	39	83	55	25	62	13
33	51	42	90	85	84	29	73	8	95	57
34	18	30	61	67	57	60	25	10	20	95
35	61	9	3	2	61	18	44	78	38	74
36	25	91	31	2	14	97	91	84	88	26
37	84	8	95	61	85	41	88	4	86	51
38	74	2	24	42	33	24	62	13	62	10
39	33	7	62	68	42	41	78	67	99	6
40	38	43	2	4	62	95	76	91	67	78
41	43	98	28	51	43	84	13	71	64	81
42	15	19	50	30	75	90	94	35	51	83
43	75	98	42	67	24	63	15	45	92	44
44	29	60	80	86	70	13	100	86	88	6
45	14	49	78	93	45	94	35	46	18	85
46	29	20	27	66	70	95	7	11	75	52
47	73	19	33	36	93	21	44	51	4	24
48	87	79	52	85	24	89	50	4	37	50
49	86	99	31	25	78	10	41	66	35	1
50	2	41	41	88	6	77	89	80	21	54
τ_j	0.8785	0.2544	0.8840	0.1757	0.6843	0.4324	0.4011	0.1903	0.4123	0.3214
φ_j	0.0091	0.0020	0.0380	0.0009	0.0048	0.0034	0.0033	0.0016	0.0038	0.0041

Appendix 3. Research data case 4

Job	Machine									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	1	1	4	3	5	5	7	6	4
2	2	5	4	3	1	9	5	4	7	0
3	5	6	8	4	4	2	5	6	7	5
4	4	1	5	6	5	7	9	2	6	2
5	4	4	2	7	3	6	5	8	4	1
6	7	6	2	5	4	1	4	7	5	5
7	8	5	8	7	9	5	3	5	1	5
8	4	2	5	8	9	9	4	7	5	8
9	2	7	4	2	5	4	5	8	4	3
10	6	5	1	9	4	4	7	6	5	1
11	5	4	7	3	9	1	4	7	3	2
12	2	4	9	2	4	5	2	1	4	2
13	4	0	1	2	2	3	1	4	2	8
14	1	2	5	7	8	6	2	1	4	8
15	6	4	5	1	2	4	5	6	2	9
16	4	5	3	1	8	7	0	1	4	6
17	7	3	1	4	7	0	4	1	5	6
18	5	2	4	1	2	7	5	3	2	3
19	8	6	8	5	7	4	2	5	9	5
20	4	5	3	5	7	9	2	4	5	8
21	3	5	7	9	6	2	4	4	7	3
22	0	2	4	5	4	7	4	5	4	8
23	4	2	5	7	4	5	3	2	8	5
24	7	8	2	1	9	6	7	8	4	1
25	4	8	5	2	6	8	9	5	8	5
26	4	5	7	2	3	7	3	6	5	4
27	4	2	1	5	1	3	5	6	5	5
28	5	8	5	7	8	2	5	8	3	5
29	5	4	5	4	5	7	6	2	5	9
30	8	2	1	5	5	6	7	8	7	5
31	8	3	5	9	5	4	5	2	4	2
32	8	5	2	5	7	6	2	8	9	5
33	3	7	4	6	8	2	4	5	2	3
34	5	5	4	7	9	8	2	5	2	5
35	5	2	5	2	5	4	8	2	1	3
36	5	5	9	5	4	9	8	5	3	5
37	2	1	2	1	4	3	3	5	2	6
38	8	8	4	7	2	6	8	6	3	5
39	9	7	5	8	5	6	5	8	9	4
40	5	6	9	6	5	3	1	8	7	4
41	6	4	7	4	3	6	1	4	5	8
42	4	3	7	5	1	9	2	4	2	5
43	4	2	8	7	3	4	9	8	7	4
44	2	5	9	4	2	5	3	0	4	7
45	9	5	4	2	3	7	0	2	1	6
46	2	3	2	5	1	0	8	9	5	3
47	5	2	7	9	4	3	6	2	5	0
48	7	8	2	1	4	7	5	8	9	4
49	1	4	2	3	6	8	2	4	7	5
50	2	5	4	5	6	8	4	1	7	5
51	8	3	0	2	5	6	8	2	9	4

 http://dx.doi.org/10.30656/jsmi.v7i2.6446

Appendix 3. Research data case 4 (continued)

Job	**Machine**									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
52	7	2	4	3	6	2	9	4	1	8
53	3	5	7	5	3	8	6	4	8	1
54	5	0	5	6	0	0	2	4	7	8
55	1	9	5	2	4	7	5	0	2	5
56	0	2	9	6	1	4	0	0	5	2
57	0	2	5	8	3	6	9	1	2	4
58	7	9	6	3	5	1	7	5	4	5
59	4	3	5	2	1	4	9	7	4	1
60	0	3	5	2	4	9	4	7	5	4
61	7	8	5	6	3	9	8	7	4	6
62	1	9	6	7	0	2	4	8	3	6
63	6	1	2	0	3	5	4	1	7	3
64	6	5	1	4	9	7	3	5	6	4
65	1	8	2	6	9	4	7	5	8	4
66	0	1	6	2	9	4	8	5	7	6
67	4	2	5	6	8	5	6	4	1	4
68	3	4	5	8	4	1	2	3	6	8
69	9	8	2	3	1	4	0	2	4	5
70	4	3	2	5	6	4	1	8	9	2
71	5	7	1	2	6	8	2	3	4	7
72	2	1	4	3	8	4	6	2	4	5
73	6	7	9	2	4	3	2	5	6	7
74	2	4	0	2	5	3	4	7	8	6
75	2	7	5	4	3	1	5	6	0	2
76	3	5	7	0	2	4	5	2	5	7
77	2	4	5	3	4	7	8	3	2	4
78	9	9	1	4	5	7	6	5	3	2
79	8	2	5	2	2	5	1	5	7	8
80	4	5	2	4	7	9	5	4	2	4
81	5	5	1	2	4	2	3	8	5	1
82	0	2	9	5	4	2	5	9	6	5
83	1	2	3	5	6	2	4	0	2	5
84	4	2	3	5	4	2	3	5	4	2
85	4	4	5	8	9	8	5	2	8	3
86	4	2	3	3	2	5	8	8	1	2
87	5	3	6	2	5	6	4	7	9	3
88	4	2	3	6	8	5	3	4	7	2
89	5	8	8	3	5	6	5	6	5	2
90	5	6	4	2	5	4	6	5	8	4
91	2	1	4	7	4	5	9	8	5	6
92	2	1	4	6	5	8	6	1	3	5
93	9	5	1	3	5	7	9	1	2	5
94	3	5	4	9	7	2	6	5	2	1
95	2	5	0	3	2	4	7	8	9	5
96	5	3	5	7	9	2	4	5	5	8
97	6	3	1	5	0	1	4	8	9	8
98	3	0	4	3	7	2	6	9	4	1
99	1	7	4	2	2	4	5	0	6	9
100	1	7	8	4	6	5	4	8	5	2
τ_j	0.7056	0.2464	0.7547	0.3150	0.5565	0.2205	0.4827	0.5384	0.4622	0.2778
φ_j	0.0068	0.0022	0.0049	0.0036	0.0051	0.0010	0.0068	0.0053	0.0031	0.0023