JURNAL ILMIAH TEKNIK INDUSTRI
ISSN: 1412-6869 (Print), ISSN: 2460-4038 (Online)
Journal homepage: http://journals.ums.ac.id/index.php/jiti/index
doi: 10.23917/jiti.v22i1.21128

No-Wait Flowshop Permutation Scheduling Problem: Fire Hawk Optimizer Vs Beluga Whale Optimization Algorithm

Muhammad Aghniya Baihaqi[1a]◆, Dana Marsetiya Utama[1b]

Abstract. *No-Wait Flowshop Permutation Scheduling Problem (NWPFSP) is a scheduling problem that states that every job completed on machine n must be processed immediately on the next machine. The NWPFSP problem is an extension of the flowshop problem. This article proposes two new algorithms fire hawk optimization and beluga whale optimization, to solve the NWPFSP problem and minimize makespan. The two new algorithms developed to solve the NWPFSP problem are tested on three different cases. Each algorithm was run 30 times and was compared using an independent sample t-test. The results were also compared with the Campbell Dudek Smtih algorithm. In addition, the effectiveness of the FHO and BWO algorithms was assessed against the CDS algorithm using the Relative Error Percentage (REP) method. The results show that the FHO and BWO algorithms are better at solving NWPFSP problems when compared to the CDS algorithm. However, the BWO algorithm is more recommended in cases with large data because it can provide better results.*

Keywords: *Fire hawk opti,izer, Beluga Whale Optimization, No-Wait Flowshop, Scheduling*

I. INTRODUCTION

Production scheduling is one of the important stages before the start of production activities to complete jobs effectively and efficiently. Product completion time must be taken into account by the company. Production delays can cause losses to the company because it can reduce customer confidence in the company. (Nadia, Dewi, & Sianto, 2017; Michael L. Pinedo, 2012). The objectives of scheduling include increasing productivity and reducing idle, reducing production process time, reducing customer waiting time, and reducing energy consumption (Baker & Trietsch, 2009, 2013). A common problem in production is the inaccuracy in order delivery due to non-optimal production schedules. The company needs to estimate how long it will take to complete the order or when it will be ready to be shipped to the customer (Hernanda & Hariastuti). In addition, scheduling problems are also caused by limited facilities and the number of machines, which results in jobs experiencing queues when they are processed because the machines are busy (Baker & Trietsch, 2013; M. L. Pinedo, 2016).

Flowshop scheduling problems naturally arise in many conditions, as there are many practical and essential applications for jobs to be processed sequentially with more than one processing stage in the industry. (Firmansyah, Utomo, & Irawan, 2016; Grabowski & Pempera, 2005). In many flowshop scheduling, there is a constraint that once the processing of a job starts, the subsequent processing must be done without any delay in the machining process from machine to machine. If necessary, the start of job processing is delayed on the first machine so that the job does not have to wait for processing on the next machine. Such a flowshop can be referred to as a 'no-wait flowshop (Rajendran, 1994; Ye, Li, & Miao, 2016). Many industries use no-wait flowshop scheduling systems to perform scheduling systems on production processes, such as the chemical industry (Raaymakers & Hoogeveen, 2000; Rajendran, 1994), steel and aluminum production (Wismer, 1972), plastic molding (Ding, Song, Zhang, Gupta, & Wu, 2015), pharmaceutical industry (Raaymakers & Hoogeveen, 2000) and many more industries that use this system. (Ye et al., 2016). Thus, this flowshop scheduling problem is interesting to be

[1] Industrial Engineering Department, Universitas Muhammadiyah Malang, Jln Tlogo Mas, Malang Indonesia 61257.

[a] email: hqfuse@gmail.com
[b] email: dana@umm.ac.id
◆ corresponding author

Submited: 28-12-2022 Revised: 26-05-2023
Accepted: 20-06-2023

studied further, especially the no-wait permutation flowshop scheduling problem (NWPFSP).

There are many kinds of research focusing on NWPFSP. Bertolissi (2000) proposed a Heuristic algorithm for scheduling in the no wait flowshop. Rajendran (1994) proposed A No-wait Flowshop Scheduling Heuristic to minimize the makespan value. Nailwal, Gupta, and Jeet (2016) also proposed A heuristic for no-wait flow shop scheduling. In addition, some studies take an approach by applying new algorithms to solve the NWPFSP problem. Pan, Wang, and Zhao (2008) proposed An improved iterated greedy algorithm Pan, Wang, and Qian (2009) with A novel differential evolution. Engin and Güçlü (2018) with A new hybrid ant colony optimization algorithm.

Although many studies have been proposed related to NWPFSP, only a few use the latest algorithms to solve NWPFSP. Studies that have used metaheuristic algorithms to solve NWPFSP are Pan et al. (2008), Engin and Güçlü (2018), and Pan et al. (2009). Unfortunately, these studies did not focus on makespan minimization. Since there are few studies, this research proposes Fire Hawk Optimizer (FHO) and Beluga Whale Optimization (BWO) algorithms for makespan minimization in NWPFSP. The FHO algorithm is a new algorithm that mimics the behavior of a Hawk in searching for its prey, proposed by Azizi, Talatahari, and Gandomi (2022). Likewise, BWO, an algorithm that mimics the behavior of beluga whales in nature proposed by Zhong, Li, and Meng (2022).FHO has been used in various studies, such as in BIM-Based Resource Tradeoff in Project Scheduling Using Fire Hawk Optimizer (FHO). Shishehgarkhaneh, Azizi, Basiri, and Moehler (2022). However, the BWO algorithm has never been proposed for NWPFSP problem solving.

Based on these reasons, we conclude that the NWPFSP problem in makespan minimization needs further investigation using new advanced algorithms. FHO and BWO algorithms were chosen because they are new and have never been applied or tested in solving NWPFSP problems focusing on makespan minimization. Therefore, this study's Research Goals (GR) are as follows: (RG 1). Developing the latest algorithms, namely FHO and BWO in solving NWPFSP problems with the goal of makespan minimization; (RG 2). Analyzing the effectiveness of using FHO and BWO algorithms in NWPFSP with the goal of makespan minimization. Based on the RGs, it is clear that this research contributes to science by providing a new reference or alternative solution in solving NWPFSP that focuses on makespan minimization. Furthermore, this research also encourages researchers and practitioners to expand their research scope in similar areas.

The structure of this article is described as follows: Section 1 is the background of this research. The method and proposed algorithm are presented in section 2. Section 3 will present the research data and the results of the research that has been carried out. Finally, section 5 presents the conclusions obtained from the research results.

II. Research Method

This section describes the notation and mathematical model of the NWPFSP problem. Figures 1 and 2 show differences related to the permutation flowshop scheduling problem (PFSP) and NWPFSP. PFSSP is a generalization of the flowshop scheduling problem (FSP) in which jobs are allowed and processed in the same job order on all machines to minimize the amount of completion time (Iqbal, 2014). NWPFSP is a case where n jobs are processed in the same order on m machines, and no job is allowed to wait between consecutive operations until the entire process is completed. Thus, the start of a job on the first machine may be delayed to satisfy the no-wait constraint.(Guevara-Guevara, Gómez-Fuentes, Posos-Rodríguez, Remolina-Gómez, & González-Neira, 2022)

The difference between FSSP and NWPFSP in Figures 1 and 2, illustrated by Makuchowski (2015).

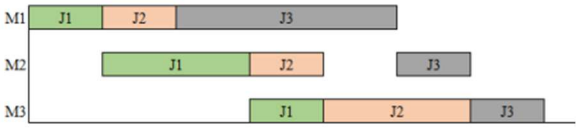

Figure 1. Permutation Flowshop

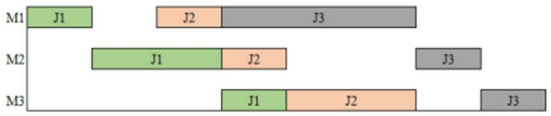

Figure 2. No-wait Permutation Flowshop

Notations and Formulation of Mathematical models

The problem in this case has the following assumptions: (1) sequential jobs or no pre-emption; (2) all jobs and machines are ready at time t=0; (3) machines are ready to use or in prime condition; (4) each machine only performs one task at a time; (5) the next task can be done if the machine has completed the previous task; (6) setup and removal times are included in the processing time.

Notations and mathematical models in solving NWPFSP to minimize makespan are as follows (Ye et al., 2016):

π Sequence for n jobs, π = [J1, J2, ..., Jj-1, ...,Jn];

n Number of jobs;

m Number of machines;

$P_{j,I}$ Process time for job j on machine I, where j-1, ..., n and i-1, ..., m;

$ST_{j,I}$ start time of job j on machine i;;

$CT_{j,i}$ job completion time for job j on machine i;

$d^{i}_{j-1\,j}$ The potential distance between the job completion time of job j-1 and the job start time of job j on machine i;

$D_{j\text{-}1j}$ the distance between the completion times of two adjacent jobs on the last machine.

$$ST_{j,1} = CT_{j-1,m} \text{ where } j = 1, 2, \dots n \quad (1)$$

$$ST_{j,1} = ST_{j-1} + \sum_{k=1}^{i-1} P_{j,k} \text{ where } j = 1, 2, \dots, n \text{ and } i = 1, 2, \dots, \quad (2)$$

$$CT_{j,1} = CT_{j,m} - \sum_{k=i+1}^{m} P_{j,k} \text{ where } j = 1, 2, \dots, n \text{ and } i = 1, 2, \dots, m \quad (3)$$

$$d^{i}_{j-1,j} = ST_{j,i} - CT_{j-1,i} = ST_{j,1} - CT_{j-1,m} + \sum_{k=1}^{i-1} P_{j,k} + \sum_{k=i+1}^{m} P_{j-1,k} = \sum_{k=1}^{i-1} P_{j,k} + \sum_{k=i+1}^{m} P_{j-1,k} \quad (4)$$

$$D_{j-1,j} = \sum_{k=1}^{m} P_{j,k} - \min d^{i}_{j-1,j} = \sum_{k=1}^{m} P_{j,k} - \min_{\{i\}}\left(\sum_{k=1}^{i-1} P_{j,k} + \sum_{k=1}^{m} P_{j-1,k}\right) = \max_{\{i\}}\left(\sum_{k=i}^{m} P_{j,k} + \sum_{k=i+1}^{m} P_{j-1,k}\right) \quad (5)$$

The formula in equation (1) assumes that the start time of job j on the last machine is the same as the finish time of job j-1 on the previous machine. Since there is no waiting time on each machine for each job, the starting time of job j on machine i and the finishing time of job j-1 on machine i are formulated in equations 2 and 3. The distance between the starting time of job j and the finishing time of job j-1 is formulated in equation (4). The potential distance between the starting time on job j and the completion time on job j-1 can be reduced by shifting job j to the left, which can be formulated in equation (5).

So that to find the completion time or makespan value can be formulated in equation (6):

$$C_{max}(\pi) = \sum_{i=1}^{m} P_{1,i} + \sum_{j=2}^{n} D_{j-1,j} \quad (6)$$

Proposed Algorithm

This section will explain the proposed FHO and BWO procedures in solving the makespan minimization problem in the NWPFSP case.

Fire Hawk Optimizer

FHO is one of the new metaheuristic algorithms proposed by Azizi et al. (2022). Hawks' behavior inspires this algorithm to hunt prey by spreading fire in the hunting area. The Hawk will take a burning stick and drop it in another unburned place to start a small fire. This small fire scares the prey and forces them to flee hastily and nervously so that the Hawk can catch them more easily. This research will solve the makespan minimization problem in the NWPFSP case using FHO.

The first stage of Fire Hawk Optimizer is initialization. Several candidate solutions (X) are initially defined as the position vectors of the Fire Hawk and its prey. A random initialization process is used to identify the initial positions of these vectors in space. The formulation of the position vectors is as in equations 7 and 8.

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} X_2^1 X_1^2 \cdots X_j^i \cdots X_1^d \\ X_2^1 X_2^2 \cdots X_2^j \cdots X_2^d \\ \vdots\vdots\vdots \ddots \vdots \\ X_i^1 X_i^2 \cdots X_i^j \cdots X_i^d \\ \vdots\vdots\vdots \ddots \vdots \\ X_N^1 X_N^2 \cdots X_N^j \cdots X_N^d \end{bmatrix} \quad (7)$$

$$X_i^j(0) = X_{i,\min}^j + \text{rand}.\left(X_{i,\max}^j - X_{i,\min}^j\right), \begin{cases} i = 1, 2, \ldots, N \\ j = 1, 2, \ldots, d \end{cases} \quad (8)$$

Where d represents the dimension of the problem under consideration; then Xi represents the i-th solution candidate in the search space; N is the total number of solution candidates in the search space; X_i^j (0) represents the initial position of the solution candidate; X_i^j is the j-th decision variable of the i-th solution candidate; X_i^j min and X_i^j max are the minimum and maximum limits of the j-th decision variable for the i-th solution candidate; and rand is a uniformly distributed random number in the range [0,1].

Next is to determine the location of the Fire Hawk in the area by evaluating the objective function for the candidate solution considering the selected optimization problem formulated in equations 9 and 10. Where F_1^H is the 1-th Fire Hawk considering the total number of n Fire Hawks in the search space, and P_k^R is the k-th prey in the search space considering the total number of m prey.

$$PR = \begin{bmatrix} PR_1 \\ PR_2 \\ \vdots \\ PR_k \\ \vdots \\ PR_m \end{bmatrix}, k = 1, 2, \ldots, m, \quad (9)$$

$$FH = \begin{bmatrix} PH_1 \\ PH_2 \\ \vdots \\ PH_k \\ \vdots \\ PH_n \end{bmatrix}, k = 1, 2, \ldots, n, \quad (10)$$

The next phase calculates the distance between the Fire Hawk and the intended prey. The calculation of the total distance between the Fire Hawk and the intended prey is formulated in equation 11.

$$D_K^I \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}, \begin{cases} I = 1, 2, \ldots, n. \\ k = 1, 2, \ldots, m.' \end{cases} \quad (11)$$

Where (x1, y1) and (x2, y2) represent the coordinates of the Fire Hawk and prey in the search space; m is the total number of prey in the search space; n is the total number of Fire Hawks in the search space; and D_K^I is the total distance between the 1st Fire Hawk and the k-th prey. In the next phase of the algorithm is the position update procedure in the main search loop of FHO, as shown in equation 12.

$$FH_I^{new} = FH_I + (r_I \times GB - r_2\, FH_{near}), I = 1, 2, \ldots, n, \quad (12)$$

Where FH_I^{new} is the new position vector of the 1-th Fire Hawk (FHl); GB is the global best solution in the search space considered as the main fire; FH_{near} is one of the other Fire Hawk in the search space; and r1 and r2 are uniformly distributed random numbers in the range of (0,1) to determine the Fire Hawk movement towards the main fire and the other Fire Hawk territory.

In the next phase of the algorithm, the movement of prey within each Fire Hawk territory is considered a key aspect of animal behavior for the position update process. This action can be considered in the position update process using equation 13.

$$PR_q^{new} = PR_q + (r_3 \times FH_I - r_4 \times SP_I), \begin{cases} I = 1, 2, \ldots, n. \\ q = 1, 2, \ldots, r. \end{cases} \quad (13)$$

Where SP_I is the safe place under the lth Fire Hawk territory; r3 and r4 are uniformly distributed random numbers in the range of (0, 1) to determine the prey's movement towards the Fire Hawk and the safe place. While PR_q^{new} is the new position vector of the q-th prey (PR_q) surrounded by the 1-th Fire Hawk (FH_I); GB is the best solution in the search space which is considered as the main fire.

In addition, the prey can move toward another Fire Hawk territory, and it is possible that the prey

Figure 3. Fire Hawk Optimizer

```
procedure Fire Hawk Optimizer (FHO)
    Determine initial positions of solution candidates (X;) in the search space with N candidates
    Apply LRV
    Evaluate fitness values for initial solution candidates
    Determine the Global Best (GB) solution as the main fire
    while Iteration <  Maximum number of iterations
        Generate n as a random integer number for determining the number of Fire Hawks
        Determine Fire Hawks (FH) and Preys (PR) in the search space
        Calculate the total distance between the Fire Hawks and the preys
        Determine the territory of the Fire Hawks by dispersing the preys
        for I=1:n
            Determine the new position of the Fire Hawks by Eq. 12.
            for q=1:r
                Calculate the safe place under Ith Fire Hawk territory by Eq. 15.
                Determine the new position of the preys by Eq. 13.
                Calculate the safe place outside the Ith Fire Hawk territory by Eq. 16.
                Determine the new position of the preys by Eq. 14.
            end
        end
        Apply LRV
        Evaluate fitness values for the newly created Fire Hawks and preys
        Determine the Global Best (GB) solution as the main fire
    end while
    return GB
end procedure
```

can get closer to the Fire Hawk near the ambush or even try to hide in a safer place outside the Fire Hawk territory where they are trapped. These actions can be considered in the position update process using equation 14.

$$PR_q^{new} = PR_q + (r_5 \times FH_{Alter} - r_6 \times SP), \begin{cases} I = 1, 2, \dots, n. \\ q = 1, 2, \dots, r. \end{cases} \quad (14)$$

Where FH_{Alter} is one of the other Fire Hawks in the search space; PR_q^{new} is the new position vector of the q-th prey (PRq) surrounded by the 1-th Fire Hawk (FH_I); SP is a safe place outside the territory of the Ith Fire Hawk; r5 and r6 are uniformly distributed random numbers in the range of (0,1) to determine the movement of prey towards other Fire Hawks and safe places outside the territory.

Based on the fact that safe havens in nature are places where most animals congregate to stay safe and healthy during danger, the mathematical presentation of SPI and SP are formulated in equations 15 and 16.

$$SP_I = \frac{\sum_{q=1}^{r} PR_q}{r}, \begin{cases} q = 1, 2, \dots, r. \\ I = 1, 2, \dots, n. \end{cases} \quad (15)$$

$$SP = \frac{\sum_{k=1}^{m} PR_K}{m}, \; k = 1, 2, \dots, m. \quad (16)$$

PRq is the qth prey surrounded by the Ith Fire Hawk (FHI); PRk is the kth prey in the search space.

Beluga Whale Optimization

The BWO algorithm is inspired by the behavior of beluga whales during beluga whale swimming, preying, and whale fall proposed by Zhong et al. (2022). This research will solve the makespan minimization problem in the NWPFSP case with

Figure 4. The pseudo code of BWO algorithm

Algorithm : The pseudo code of BWO algorithm
Input: Algorithmic parameters (population size, maximum iteration)
Output: The best solution

Initialize the population, Apply LRV, and evaluate fitness values, obtain the best solution (P*)
While $T \leq Tmax$ **Do**
  Obtain probability of whale fall Wf by Eq. (26) and balance factor Bf by Eq. (19)
  For each beluga whale (Xi) **Do**
    If $Bf(i) > 0.5$
      // In the exploration phase
      Generate pj (j = 1,2,...,d) randomly from dimension
      Choose a beluga whale Xr randomly
      Update new position of i-th beluga whale using Eq. (20)
    Else If $Bf(i) \leq 0.5$
      // In the exploitation phase
      Update the random jump strength $C1$ and calculate the Levy flight function
      Update new position of i-th beluga whale using Eq. (21)
    End If
    Check the boundaries of new positions, Apply LRV, and evaluate the fitness values
  End For
  For each beluga whale (Xi) **Do**
    // the whale fall phase
    If $Bf(i) \leq Wf$
      Update the step factor $C2$
      Calculate the step size $Xstep$
      Update new position of i-th beluga whale using Eq. (24)
      Check the boundaries of new position, Apply LRV, and calculate fitness value
    End If
  End For
  Find the current best solution P^*
  $T = T+1$
End While
Output the best solution

three stages in BWO. The stages in BWO are described as follows: (1) Initiation Phase, (2) Exploration Phase, and (3) Whale Fall.

Since BWO is population-based, beluga whales are considered search agents, and each is also considered a candidate solution. So the position matrix is modeled in equation 17.

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{bmatrix} \quad (17)$$

Where n is the population size of beluga whales, d represents the dimension of the design variable. The corresponding fitness values are modeled for all beluga whales in equation 18.

$$f_x = \begin{bmatrix} f(x_{1,1}, x_{1,2}, \ldots, x_{1,d}) \\ f(x_{2,1}, x_{2,2}, \ldots, x_{3,d}) \\ \vdots \\ f(x_{n,1}, x_{n,2}, \ldots, x_{n,d}) \end{bmatrix} \quad (18)$$

The algorithm can change from exploration to exploitation depending on the equilibrium factor B_f modeled by equation 19.

$$B_f = B_0(1 - \frac{T}{2T_{max}}) \quad (19)$$

Where B_0 changes randomly between (0,1) at each iteration, T is the current iteration, and Tmax is the maximum number of iterations,. The exploration stage occurs when the equilibrium factor B_f > 0.5 while the exploitation stage occurs when B_f = 0.5. As the iteration T increases, the fluctuation range of B_f decreases from (0, 1) to (0, 0.5), illustrating a significant change in probability for the exploitation and exploration phases, while the probability of the exploitation phase increases as the iteration T increases.

Exploration Phase

In the exploration phase, BWO is performed by considering the swimming behavior of beluga whales. Therefore, the position of the search agent is determined by the swimming pair of the beluga whale, and the position of the beluga whale is updated by modeling in equation 20.

$$\begin{cases} X_{1,j}^{T+1} = X_{1,p}^{T} + \left(X_{r,p1}^{T} + X_{1,pj}^{T}\right)(1 + r_1)\sin(2\pi r_2), \ j = even \\ X_{1,j}^{T+1} = X_{1,p}^{T} + \left(X_{r,p1}^{T} + X_{1,pj}^{T}\right)(1 + r_1)\sin(2\pi r_2), \ j = odd \end{cases} \quad (20)$$

Where T is the current iteration, $X_{1,pj}^{T}$ is the position of the i-th beluga whale at dimension pj, $X_{1,j}^{T+1}$is the new position of the i-th beluga whale at dimension j, pj (j = 1, 2, ... , d) is a random number selected from the d-dimension, $X_{1,pj}^{T}$ and $X_{r,p1}^{T}$ are the current positions for the i-th and r-th beluga whales (r is a randomly selected belug`a whale), r1 and r2 are random numbers between (0, 1), sin (2πr2) and cos (2πr2) means the beluga whale fins are mirrored towards the surface. Based on the dimensions chosen by the odd and even numbers, the updated position reflects synchronous behavior when swimming or diving. Two random numbers r1 and r2 are used to enhance the random operator in the exploration phase.

Exploitation Phase

The preying behavior of beluga whales inspires BWO's exploitation phase. Beluga whales can cooperatively forage and move according to the location of nearby beluga whales. Therefore, beluga whales prey by sharing position information, considering the best candidate and others. This phase also uses the Levy Flight strategy (Mantegna, 1994) to improve convergence. Mathematical modeling in the exploitation phase is formulated in equation 21.

$$X_i^{T+1} = r_3 X_{best}^{T} - r_4 X_i^{T} + C_1 \times L_F \times (X_r^{T} - X_i^{T}) \quad (21)$$

Where T is the current iteration, X_r^T and X_i^T are the current positions for the i-th beluga whale and the random beluga whale, X is the position of the new position of the i-th beluga whale, X_{best}^T is the best position among the beluga whales, r3 and r4 are random numbers between (0,1), C1 = 2r4(1-T/Tmax) is the random jump strength that measures the intensity of Levy Flight.

L_F is the Levy Flight function (Mantegna, 1994) formulated by the mathematical model in equations 22 and 23.

$$L_F = 0{,}05 \times \frac{\mu \times \sigma}{|\nu|^{1/\beta}} \quad (22)$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma((1+\frac{\beta}{2}) \times \beta \times 2^{(\beta-1)/2}}\right)^{1/\beta} \quad (23)$$

Where u and v are normally distributed random numbers, β is a default constant equal to 1.5.

Whale Fall

The Whale Fall phase is an inspired phase when beluga whales die. During migration and foraging, beluga whales are threatened by killer whales, polar bears, and humans. Most beluga whales are intelligent and can escape threats by sharing information. However, many beluga whales do not survive and fall to the bottom of the deep sea. This phenomenon, called "whale fall", feeds many creatures.

To ensure the sum of the population size is constant, the position of the beluga whales and the size of the Whale Fall step are used to establish the latest position. The mathematical model is expressed in equation 24.

$$X_i^{T+1} = r_5 X_i^{T} - r_6 X_r^{T} + r_7 X_{step} \quad (24)$$

X_{step} is the Whale Fall step size modeled in equation 25. While r_5, r_6, and r_7 represent random numbers between (0.1).

$$X_{step} = (\mu_b - 1_b)\exp(\frac{-C_2 T}{T_{max}}) \quad (25)$$

ub and lb are the upper and lower limits of the variable, respectively. The variable C2 is a step factor related to the probability of whale fall and population size (C2=2Wf×n). The step size is affected by the boundaries of the design variables, iterations and the maximum iterative number. The probability of Whale Fall (W_f) can be calculated as a linear function in equation 26.

$$W_f = 0.1 - 0.05T/T_{max} \quad (26)$$

The probability of Whale Fall decreases from 0.1 at the beginning of the iteration to 0.05 at the last iteration, indicating that as the beluga whale gets closer to the food source during the optimization process, the danger of the beluga whale decreases.

Largest Rank Value

According to research by Li and Yin (2013) LRV is an effective way of mapping job permutations. Where the first largest value is selected as the first order of job permutation. After that, select the second largest value as the second order in the work. This experiment will apply LRV in sorting FHO and BWO positions in determining a job sequence. An illustration of the use of LRV is shown in Figure 5, which refers to research conducted by D. M. Utama.

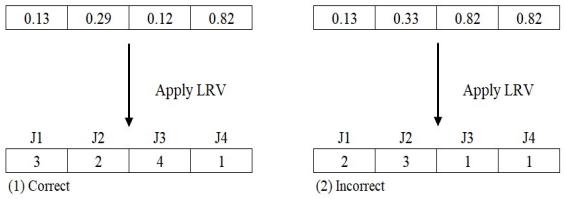

Figure 5. Largest Rank Value

Experimental data and procedures

The first research data use the data in table 1 published by Nailwal et al. (2016) with a problem of 5 jobs with 3 machines, referred to as Case 1 and belongs to the small data category. The second research data use Table 2 published by Carlier (1978) Car 06 with a problem of 8 jobs with 9 machines called Case 2 and belongs to the medium data category. The third research data use data in table 3 published by Reeves (1995) Rec 09 with a problem of 20 jobs with 10 machines called Case 3 and belongs to the big data category.

III. RESULT AND DISCUSSION

Algorithm Test Results

Table 1. Reasearch data case 1

Job	Machine		
	Machine 1	Machine 2	Machine 3
1	3	2	4
2	4	5	3
3	1	4	5
4	1	3	2
5	4	3	7

Table 2. Reasearch data case 2

Job	Machine								
	1	2	3	4	5	6	7	8	9
1	887	447	234	159	201	555	463	456	753
2	799	779	567	267	478	444	123	789	21
3	999	999	852	483	520	120	456	630	427
4	666	666	140	753	145	142	789	258	520
5	663	25	222	420	699	578	876	741	142
6	333	558	558	159	875	965	543	36	534
7	222	886	965	25	633	412	210	985	157
8	114	541	412	863	222	25	123	214	896

Table 3. Reasearch data case

job	Machine									
	1	2	3	4	5	6	7	8	9	10
1	77	95	41	97	47	45	10	41	72	8
2	99	28	42	4	7	30	65	45	51	94
3	74	25	92	29	4	21	47	36	61	9
4	4	21	40	80	66	85	1	33	1	4
5	49	95	96	74	96	63	59	84	70	29
6	53	59	75	19	13	50	82	60	9	13
7	88	47	28	11	86	90	93	38	33	59
8	92	99	84	13	73	55	19	93	74	25
9	2	49	86	46	58	42	24	79	12	17
10	97	18	28	77	92	54	49	24	19	71
11	28	93	93	7	25	89	49	11	93	45
12	64	22	91	56	46	27	32	70	94	5
13	25	96	98	51	21	20	93	64	86	11
14	19	41	87	15	31	78	54	74	71	6
15	81	1	74	56	8	55	3	92	28	5
16	9	29	49	48	72	38	26	3	49	80
17	5	74	19	27	71	35	52	76	79	47
18	8	66	40	71	17	61	84	49	52	56
19	34	7	58	94	22	27	40	19	26	77
20	13	56	45	27	40	26	90	28	27	88

This section describe and explain the results of the study by considering the effect of different population numbers and the number of iterations on Makespan. In addition, this section will compare the effectiveness of the FHO algorithm with the CDS algorithm shown in tables 4 and 5 in determining the makespan value.

Table. 4 Makespan calculation results with CDS method

Reaserach data	Number of Machines and Jobs	Makespan
Case 1	5 job and 3 machine	30
Case 2	8 job and 9 machine	12334
Case 3	20 job and 10 machine	2628

The test results in Table 4 and Table 5 show that the FHO and BWO algorithms get a better makespan value when compared to the CDS algorithm. In addition, after testing, it is known that the number of populations and the number of iterations will affect the calculation results of the FHO and BWO algorithms. The higher the population and iterations used, the better the makespan results will be compared to the population size and the small number of iterations, especially if the data tested is large data.

Comparison of Algortima Testing Results

This section describes the results of testing the FHO and BWO algorithms and the results of the independent sample t-test.

After testing the data 30 times in Table 6, the independent sample t-test was continued. According to McMillan and Schumacher (2010) the independent sample t-test is an inferential statistical procedure to determine the possibility of rejecting the null hypothesis that the two results are the same. So, in this study, the independent sample t-test test was used to

Table 5. Makespan calculation results with FHO and BWO algorithm

Reaserach data	Number of Machines and Jobs	Iteration	Population	Makespan FHO	Makespan BWO
Case 1	5 job and 3 machine	100	100	25	25
			300	25	25
			500	25	25
		300	100	25	25
			300	25	25
			500	25	25
		500	100	25	25
			300	25	25
			500	25	25
Case 2	8 job and 9 machine	100	100	9690	9690
			300	9690	9690
			500	9690	9690
		300	100	9690	9690
			300	9690	9690
			500	9690	9690
		500	100	9690	9690
			300	9690	9690
			500	9690	9690
Case 3	20 job and 10 machine	100	100	2137	2128
			300	2137	2100
			500	2101	2063
		300	100	2099	2132
			300	2091	2058
			500	2084	2047
		500	100	2103	2094
			300	2071	2051
			500	2071	2047

determine whether there was a difference between the FHO and BWO test results. In addition, it aims to find out which one is more effective in solving NWPFSP in reducing makespan. The results of the independent sample t-test are shown in Table 7.

IV. CONCLUSION

In this study, researchers proposed the FHO and BWO algorithms to minimize makespan in the No-Wait Flowshop Scheduling Problem. Test results show where the FHO and BWO algorithms are better when compared to the CDS algorithm. The FHO and BWO algorithms, when used in small data case studies, provide the same makespan value, so the FHO and BWO algorithms are recommended for use in solving NWPFSP. However, the test results on big data case studies are different. The results of the analysis on the independent sample t-test test show that the two algorithms have different results. In this test, the mean result of the BWO algorithm is lower than the FHO, so it can be concluded that BWO is more recommended in solving NWPFSP in large data case studies. In addition, the test results analysis shows that the number of populations

Table 6. FHO and BWO test result

Replications	Case 1		Case 2		Case 3	
	FHO	BWO	FHO	BWO	FHO	BWO
1	25	25	9690	9690	2122	2057
2	25	25	9690	9690	2116	2174
3	25	25	9690	9690	2122	2056
4	25	25	9690	9690	2153	2135
5	25	25	9690	9690	2104	2133
6	25	25	9690	9690	2105	2079
7	25	25	9690	9690	2155	2088
8	25	25	9690	9690	2157	2082
9	25	25	9690	9690	2148	2090
10	25	25	9690	9690	2123	2126
11	25	25	9690	9690	2135	2075
12	25	25	9690	9690	2153	2101
13	25	25	9690	9690	2159	2086
14	25	25	9690	9690	2154	2143
15	25	25	9690	9690	2152	2081
16	25	25	9690	9690	2117	2079
17	25	25	9690	9690	2147	2075
18	25	25	9690	9690	2112	2055
19	25	25	9690	9690	2112	2108
20	25	25	9690	9690	2120	2091
21	25	25	9690	9690	2122	2083
22	25	25	9690	9690	2109	2094
23	25	25	9690	9690	2136	2084
24	25	25	9690	9690	2121	2101
25	25	25	9690	9690	2103	2047
26	25	25	9690	9690	2149	2053
27	25	25	9690	9690	2123	2090
28	25	25	9690	9690	2126	2105
29	25	25	9690	9690	2122	2088
30	25	25	9690	9690	2071	2082

Table 7. independent sample t-test result

	Case 1		Case 2		Case 3	
	FHO	BWO	FHO	BWO	FHO	BWO
Mean	25	25	9690	9690	2128,2667	2091,3667
Standard Deviation	0	0	0	0	20,98757	28,45745
t	-		-		5,716	
Sig. (2-tailed)	-		-		0	

and the number of iterations tested will affect the calculation results, especially on big data. The greater the number of populations and iterations, the smaller the makespan results. Suggestions for future research are to compare with other algorithms and provide different approaches or goals in using the FHO and BWO algorithms.

REFERENCES

Azizi, M., Talatahari, S., & Gandomi, A. H. (2022). Fire

Hawk Optimizer: a novel metaheuristic algorithm. *Artificial Intelligence Review*, 1-77.

Baker, K. R., & Trietsch, D. (2009). Safe scheduling: Setting due dates in single-machine problems. *European Journal of Operational Research, 196* (1), 69-77.

Baker, K. R., & Trietsch, D. (2013). *Principles of sequencing and scheduling*: John Wiley & Sons.

Bertolissi, E. (2000). Heuristic algorithm for scheduling in the no-wait flow-shop. *Journal of Materials Processing Technology, 107* (1-3), 459-465.

Carlier, J. (1978). Ordonnancements à contraintes disjonctives. *RAIRO-Oper. Res., 12* (4), 333-350.

Ding, J., Song, S., Zhang, R., Gupta, J. N. D., & Wu, C. (2015). Accelerated methods for total tardiness minimisation in no-wait flowshops. *International Journal of Production Research, 53* (4), 1002-1018.

Engin, O., & Güçlü, A. (2018). A new hybrid ant colony optimization algorithm for solving the no-wait flow shop scheduling problems. *Applied Soft Computing, 72*, 166-176.

Firmansyah, A., Utomo, D., & Irawan, M. (2016). Algoritma Genetika Ddengan Modifikasi Kromosom Untuk Penyelesaian Masalah Penjadwalan Flowshop. *J. Sain dan Seni, 1* (1).

Grabowski, J., & Pempera, J. (2005). Some local search algorithms for no-wait flow-shop problem with makespan criterion. *Computers & Operations Research, 32* (8), 2197-2212.

Guevara-Guevara, A., Gómez-Fuentes, V., Posos-Rodrígu*ez, L., Remolina-Gómez, N., & González-Neira, E. (2022). Earliness/tardiness minimization in a no-wait flow shop with sequence-dependent setup* times. *Journal of Project Management, 7* (3), 177-190.

Hernanda, D. A., & Hariastuti, N. L. P. (2022). *Usulan Penjadwalan Produksi Pada Departemen Produksi PT. Preshion Engineering Plastec*.

Iqbal, P. (2014). Genetic Algorithm for Permutation Flowshop Scheduling Problem to Minimize the Makespan. *International Journal of Computing Algorithm, 3*, 1086-1091. doi:10.20894/IJCOA.101.003.002.021

Li, X., & Yin, M. (2013). An opposition-based differential evolution algorithm for permutation flow shop scheduling based on diversity measure. *Advances in Engineering Software, 55*, 10-31.

Makuchowski, M. (2015). Permutation, no-wait, no-idle flow shop problems. *Archives of Control Sciences* (2).

Mantegna, R. N. (1994). Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. *Physical Review E, 49* (5), 4677.

McMillan, J. H., & Schumacher, S. (2010). *Research in Education: Evidence-Based Inquiry*, MyEducationLab Series. Pearson.

Nadia, V., Dewi, D. R. S., & Sianto, M. E. (2017). Penjadwalan Produksi dan Perancangan Persediaan Bahan Baku di PT. Wahana Lentera Raya. *Widya Teknik, 9* (2), 179-192.

Nailwal, K., Gupta, D., & Jeet, K. (2016). Heuristics for no-wait flow shop scheduling problem. *International Journal of Industrial Engineering Computations, 7* (4), 671-680.

Pan, Q.-K., Wang, L., & Qian, B. (2009). A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems. *Computers & Operations Research, 36* (8), 2498-2511.

Pan, Q.-K., Wang, L., & Zhao, B.-H. (2008). An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion. *The International Journal of Advanced Manufacturing Technology, 38* (7), 778-786.

Pinedo, M. L. (2012). *Scheduling* (Vol. 29): Springer.

Pinedo, M. L. (2016). *Scheduling: Theory, Algorithms, and Systems*, Springer International Publishing.

Raaymakers, W. H. M., & Hoogeveen, J. A. (2000). Scheduling multipurpose batch process industries with no-wait restrictions by simulated annealing. *European Journal of Operational Research, 126* (1), 131-151.

Rajendran, C. (1994). A no-wait flowshop scheduling heuristic to minimize makespan. *Journal of the Operational Research Society, 45* (4), 472-478.

Reeves, C. R. (1995). A genetic algorithm for flowshop sequencing. *Computers & Operations Research, 22* (1), 5-13. doi:https://doi.org/10.1016/0305-0548(93)E0014-K

Shishehgarkhaneh, M. B., Azizi, M., Basiri, M., & Moehler, R. C. (2022). BIM-Based Resource Tradeoff in Project Scheduling Using Fire Hawk Optimizer (FHO). *Buildings, 12* (9). doi:10.3390/buildings12091472

Utama, D. M. (2021). *Minimizing Number of Tardy Jobs in Flow Shop Scheduling Using A Hybrid Whale Optimization Algorithm*.

Utama, D. M., Widodo, D. S., Ibrahim, M. F., & Dewi, S. K. (2020). A new hybrid butterfly optimization algorithm for green vehicle routing problem. *Journal of Advanced Transportation*, 2020.

Wismer, D. A. (1972). Solution of the flowshop-scheduling problem with no intermediate queues. *Operations Research, 20* (3), 689-697.

Ye, H., Li, W., & Miao, E. (2016). An effective heuristic for no-wait flow shop production to minimize

makespan. *Journal of Manufacturing Systems, 40*, 2-7.

Zhong, C., Li, G., & Meng, Z. (2022). Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. *Knowledge-Based Systems*, 109215.