Minimizing Number of Tardy Jobs in Flow Shop Scheduling Using A Hybrid Whale Optimization Algorithm

D M Utama

Department Industrial Engineering, University of Muhammadiyah Malang, Jl. Tlogomas No. 246, 65144 Malang, East Java, Indonesia

Email: dana@umm.ac.id

Abstract. The number of Tardy Jobs is a critical performance in scheduling because it can increase customer trust in the company. If the number of tardy jobs is large, trust in the company decreases. This article aims to minimize the number of tardy jobs in Permutation Flow Shop Scheduling Problems (PFSSP) using Hybrid Whale Optimization Algorithm (HWOA). We propose the HWOA method for minimizing the number of tardy jobs in PFSSP. There are 5 phases of the proposed HWOA algorithm. Phase 1 is replacing 1of the initial search agent population replaced by the NEH-EDD heuristic procedure. The Random Search Agent position's initialization and the Large Rank Value (LRV) is the second phase. Phase 3 is the whale exploration phase. Phase 4 is the whale exploitation phase. The last phase (5) is the local search exchange procedure. Numerical experiments were conducted to test the performance of HWOA, and it was compared with the previous research algorithm. Algorithm experiment results show that the HWOA algorithm has better performance than Particle Swarm Optimization and Genetic Algorithms.

Keywords: Flow shop, Scheduling, tardy jobs, Metaheuristic, Whale Algorithm

1. Introduction

Scheduling has an essential role in the manufacturing industry sector [1]. It is an activity of allocating resources to complete a set of jobs [2] [3]. One of the scheduling problems is the Permutation Flow Shop Scheduling Problem (PFSSP). In this problem, n jobs have the same sequence process to complete on m machines [4, 5]. Recently, the manufacturing industry must manage production properly in order to work. Therefore, companies need to carry out scheduling appropriately. The objective of the scheduling function is to optimize one or more company performance. The number of tardy jobs is one of the critical scheduling performances because it affects costs, profitability, and customer satisfaction. In addition, the number of tardy jobs influences a company's long-term financial performance [6].

Several researchers have conducted PFSSP research to complete the number of tardy jobs. Generally, researchers have proposed heuristic procedures such as EDD [7-11], Hermelin Algorithm [12], Branch and Bound [13], and Moore’s Algorithm [6]. Currently, some metaheuristic procedures have been proposed to solve PFSSP problems such as GRASP and PSO [14], MBIP [15], Ant Lion Optimization (ALO) [16], Differential Heuristic Algorithm [17], Genetic Algorithm (GA) [18, 19], Greedy Algorithm [20, 21], NEH [22], and Particle Swarm Optimization (PSO) [23]. The metaheuristic procedure is chosen because the PFSSP problem is categorized as an NP-hard problem [5]. Moreover, the solution to this problem is difficult to find in polynomial time [24]. Because of this, several PFSSP studies use of metaheuristic algorithms.
Research on the number of tardy jobs in PFSSP is mostly limited to the heuristics procedure. As far as our knowledge, very little attention has been paid to the PFSSP problem of minimizing the number of tardy jobs using the metaheuristics algorithm. One sophisticated algorithm to solve scheduling problems is the Hybrid Whale optimization Algorithm (HWOA) algorithm. Based on previous research, we have not found research on PFSSP using the HWOA algorithm to solve the problem of minimizing the number of tardy jobs. HWOA has been used to solve the PFSSP problem by researchers. Utama, Widodo, Ibrahim, Hidayat, Baroto and Yurifah [25] used HWOA to minimize energy consumption, and Abdel-Basset, Manogaran, El-Shahat and Mirjalili [26] proposed HWOA reduce completion time. This algorithm is an algorithm inspired by whales’ behavior in looking for prey proposed by Mirjalili and Lewis [27]. Therefore, this study aims to develop a Whale Algorithm algorithm to minimize the number of tardy jobs in PFSSP problems. It is hoped that this study will contribute to the problem of reducing the number of tardy jobs in PFSSP by proposing a new metaheuristic algorithm.

2. Methods

2.1. Assumptions and Problem Definitions

Several assumptions in PFSSP with the minimization of the number of tardy jobs; (1) each job can be processed in each machine only once; (2) Each machine can only process one job (3) each job j takes time to be processed in each machine i, (4) the arrival of jobs at the same time (t = 0); (5) start time is 0; (6) travel time is considered zero; (7) There are no interruptions to work on jobs; (8) Every machine used is always ready; (9) Each job has a due date; (10) The setup time is included in the processing time.

The notation used in this PFSSP problem is as follows:

\[ n \] : number of jobs
\[ m \] : number of machines
\[ j \] : index of jobs
\[ i \] : index of machines
\[ j_k \] : jobs in sequence \( k \)
\[ n_{j_k}\] : number of tardy jobs

\[ \text{pf} \] : processing time
\[ \text{NEH} \] : NEH Heuristic Algorithm
\[ \text{WOA} \] : Whale Optimization Algorithm

\[ c(j_1, i_1) = pt_{j_1i_1}, k = 1, h = 1 \] (1)
\[ c(j_1, i_h) = c(j_1, i_{h-1}) + pt_{j_1i_1}, h = 2, ... m \] (2)
\[ c(j_k, i_1) = c(j_{k-1}, i_1) + pt_{j_{k-1}i_1}, k = 2, ... n \] (3)
\[ c(j_k, i_h) = \max(c(j_{k-1}, i_h), (j_1, i_{h-1})))) + pt_{j_{k-1}i_1}, k = 2, ... n, h = 2, ... m \] (4)
\[ n_{j_k}\] : number of tardy jobs

Equation 1 describes the completion time of job sequence one on machine 1; Equation 2 describes the completion time for job sequence 1 on machines 2 to m; Equation 3 describes the time for completing job sequence k on machine 1; Equation 4 shows the time for completing job sequence k from machine h; Equation 5 shows the formula for the number of tardy jobs.

2.2. The Proposed Method of Hybrid Whale Optimization Algorithm (HWOA)

This study proposes HWOA to minimize the number of tardy jobs. HWOA is an algorithm that combines the Whale Optimization Algorithm (WOA) and the NEH Heuristic Algorithm and local search. The pseudocode algorithm is presented to algorithm 1. There are 5 phases of the HWOA algorithm. In phase 1, 1 of the initial population of Whale search agents is replaced with NEH-EDD.

\[ n_{j_k}\] : number of tardy jobs

Equation 1 describes the completion time of job sequence one on machine 1; Equation 2 describes the completion time for job sequence 1 on machines 2 to m; Equation 3 describes the time for completing job sequence k on machine 1; Equation 4 shows the time for completing job sequence k from machine h; Equation 5 shows the formula for the number of tardy jobs.

2.2. The Proposed Method of Hybrid Whale Optimization Algorithm (HWOA)

This study proposes HWOA to minimize the number of tardy jobs. HWOA is an algorithm that combines the Whale Optimization Algorithm (WOA) and the NEH Heuristic Algorithm and local search. The pseudocode algorithm is presented to algorithm 1. There are 5 phases of the HWOA algorithm. In phase 1, 1 of the initial population of Whale search agents is replaced with NEH-EDD.
Phase 2 is Random Search Agent Initialization and Large Rank Value (LRV). Phase 3 is the whale exploration phase. Phase 4 is the whale exploitation phase. The last phase (5) is the local search exchange procedure. These 5 phases are adjusted as follows:

**a) NEH-EDD Algorithm**
NEH-EDD is an efficient heuristic algorithm for solving scheduling problems [22]. We propose that one of the initial population of whale search agents be initialized with NEH-EDD. The proposed NEH-EDD stages are as follows: (1) Sort the jobs based on the smallest due date to largest; (2) Choose the first two jobs and choose the best partial permutation to minimize the number of tardy jobs. If there is no number of late jobs late, the partial permutation is chosen based on the shortest completion time. (3) Add the job in the third rank to the partial permutation in step 2. Insert the three possible positions. Evaluate all possible partial permutations and select the best permutation of the number of tardy jobs. (4). Add the next job in the next rank to the last best partial permutation. Evaluate the job in all possible positions from partial permutations and select the best partial permutations. (5). Repeat step 5 until all jobs are scheduled in the job permutation.

**b) Initialize the Position of the Random Search Agent and the Large Rank Value (LRV)**
This study proposes that the search agent whale position is not repeated for the same search agent. This position illustration is shown in Figure 1. Mirjalili and Lewis [27] proposed the WOA algorithm to solve continuous problems. This algorithm cannot be applied to the PFSSP problem because the PFSSP problem is a combinatorial discrete space problem. This study proposes the conversion of search agents to job permutations by applying the Large Rank Value (LRV). LRV is an effective way to map continuous values into job permutations [28].

\[
\begin{bmatrix}
0.52 & 0.79 & 0.61 & 0.91 \\
0.71 & 0.53 & 0.98 & 0.86 \\
0.55 & 0.72 & 0.94 & 0.84 \\
0.12 & 0.33 & 0.71 & 0.71 \\
0.61 & 0.82 & 0.53 & 0.53 \\
0.55 & 0.76 & 0.72 & 0.94 \\
\end{bmatrix}
\]

(1) Accepted  (2) Rejected population

![Fig. 1. Search Agent Position](image)

| 0.41 | 0.65 | 0.34 | 0.93 | 0.43 | 0.61 | 0.85 | 0.85 |
|------|------|------|------|------|------|------|------|

Apply LRV

J1  J2  J3  J4

(1) Correct Job Permutation

| 3  | 2  | 4  | 1  |

J1  J2  J3  J4

(2) Incorrect Job Permutation

| 2  | 3  | 1  | 1  |

**c) Exploration Phase**
The behavior encircles of prey are presented in equations (6) and (7). Where \(\overline{D}\) indicates the position of the whale’s distance to the prey, \(t\) denotes iteration, \(\overline{A}\) and \(\overline{C}\) are coefficient vectors. \(X^*\) is the position vector of the best solution obtained. \(\overline{X}\) is the position vector, and || \(s\) an absolute value. \(X^*\) should be updated in every iteration if a better solution exists. Vectors \(\overline{A}\) and \(\overline{C}\) are calculated based on equations (8) and (9). Where \(\overline{d}\) is decreases linearly from 2 to 0 during the experiment (in the exploration and exploitation phase), and \(\overline{r}\) is a random vector with the range [0,1].

\[\overline{D} = |\overline{C} \cdot \overline{X}^\ast(t) - \overline{X}(t)|\]  
(6)

\[\overline{X}(t + 1) = \overline{X}^\ast(t) - \overline{A} \cdot \overline{D}\]  
(7)
\[ \vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \]  \hspace{1cm} (8)

\[ \vec{C} = 2 \cdot \vec{r} \]  \hspace{1cm} (9)

---

**Algorithm 1 Pseudo-code Hybrid Whale Optimization Algorithm**

1. Initialize a population of \( n \) random whales or search agents
2. Apply LRV on each search agent to be mapped into a sequence
3. Solve the Scheduling problem with the number of the tardy job using NEH
4. Choose about 1 search agents from the population and replace them with NEH
5. Evaluate each search agent
   - \( X^* = \text{the best search agent} \)
6. \( t = 0 \)
7. **while** (\( t < \) maximum number of iterations)
   - for each search agent in the population
     - Update \( a, A, C, l, \) and \( p \)
     - If (\( p < 0.5 \))
       - If (\( |A| < 1 \))
         - Update the position of the current search agent by the equation (6)
       - else if (\( |A| \geq 1 \))
         - Select a random search agent (\( X_{rand} \))
         - Update the position of the current search agent by the equation (11)
     - end if
     - else if (\( p \geq 0.5 \))
       - Update the position of the current search by the equation (10)
     - end if
   - end for
   - Ensure no repeated values in the same search agent \( x^{t+1} \)
   - Apply LRV on each search agent \( x^{t+1} \)
   - for \( i = 0 : 0.01 \times n \)
     - Perform swap mutation on the search agent \( x^{t+1} \)
     - if (evaluate (\( x^{t+1} \)) \(<\) evaluate (\( X^* \)))
       - \( X^* = x^{t+1} \)
     - end if
   - end for
   - Perform flip operation on a random search agent \( x_r \)
   - for \( i = 0 : 0.01 \times n \)
     - if (evaluate (\( x_r \)) \(<\) evaluate (\( X^* \)))
       - \( X^* = x_r \)
     - end if
   - end for
   - \( X^* = \text{perform local search strategy on the best search agent} X^* \)
   - Evaluate the search agent
   - Update \( X^* \) if there is a better solution in the population
   - \( t = t + 1 \)
8. **end while**
9. return*
d) Exploitation Phase

The whale's behavior in updating the spiral position is formulated in Equation (10). Where $\vec{D} = [\bar{X}^*(t) - \vec{X}(t)]$ which indicates the distance of the whale to prey (the best solution obtained). $b$ is a constant spiral, and $l$ is a random number with the range $[-1, 1]$. The whale's behavior to update the position is presented in Equation (11). Where $p$ is a random number with the range $[0, 1]$.

$$\vec{X}(t + 1) = \vec{D} \cdot e^{bl} \cdot \cos (2\pi l) + \vec{X}^2(t)$$  \hspace{1cm} (10)$$

$$\vec{X}(t + 1) = \begin{cases} \vec{X}^2(t) - \vec{A} \cdot \vec{D} \\ \vec{D} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^2(t) \end{cases}$$ \quad \text{if } p < 0.5 \\
\begin{cases} \vec{X}^2(t) - \vec{A} \cdot \vec{D} \\ \vec{D} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^2(t) \end{cases} \quad \text{if } p \geq 0.5 \hspace{1cm} (11)$$

e) Local Search Procedure

This study proposes a Local Search procedure to improve WOA performance. The two proposed local search rules are swap and flip. In a swap operation, two positions are randomly selected, and they are swapped. Furthermore, a flip is performed by reversing the order of randomly selected jobs.

2.3. Data Collection

This study used processing time data from the research by Carlier [29] and Heller [30]. There were three problem variants used in this experiment. We generated due date times from uniform distributions. The data used in this study are presented in Table 1.

| Problem | Data from | Job and machine | Generate uniform due date time |
|---------|-----------|----------------|-------------------------------|
| Case 1  | Carlier [29] | 10 job 6 machine | 1000 – 3000                    |
| Case 2  | Heller [30]  | 20 job 10 machine | 20 – 50                        |
| Case 3  | Heller [30]  | 100 job 10 machine | 30 - 60                       |

2.4. Experimental Procedure

This study used a combination of different iteration and population parameters. The population used 3 population levels, namely 10, 50, and 100. Iteration applied 4 different iteration levels, such as 10, 50, 100, and 200. Therefore, this experiment was carried out as many as 36 experiments. The HWOA algorithm was compared with other algorithms, such as particle swarm optimization (PSO) and Genetic Algorithm (GA). The PSO and GA algorithms used 200 iterations. 100 particles were applied to PSO, and 100 chromosomes were applied to GA. Experiments were carried out with Matlab R2014a software's help on Windows 10 Intel (R) Core (TM) i3-2348M CPU RAM 2 Gb. To measure the HWOA algorithm's performance, this study implemented the Efficiency Index (EI), which is presented in Equation (12).

$$\text{EIP} = \frac{\text{proposed algorithm}}{\text{another algorithm}} \times 100\%$$  \hspace{1cm} (12)$$

3. Results and Discussion

The results of the parameter experiment of several problem cases are shown in Table 2. The most apparent findings from the analysis are iterations and large populations, minimizing the number of tardy jobs. However, in the case of a small number of jobs, the number of population and the large iteration did not significantly affect the minimization of the number of tardy jobs. Further analysis shows that the uses a large population and iteration for the case of a large number of jobs because it is very significant to minimize the number of tardy jobs.
Table 2. Results of number of tardy jobs and computation time

| Problem | population | Number of Tardy Job | Computation Time |
|---------|------------|---------------------|------------------|
|         |            | 10  50  100  200  10  50  100  200 |
| Case 1  |            | 10 8  8  8  8  1.46  4.9  10.75  20.73 |
|         |            | 50 8  8  8  8  3.25  15.87  34.42  68.6 |
|         |            | 100 8  8  8  8  6.09  33.15  61.34  126.32 |
| Case 2  |            | 10 18 17 16 15  1.9  9.2  18.3  37.3 |
|         |            | 50 17 16 15 14  4.6  23.3  46.4  95.6 |
|         |            | 100 17 16 15 14  8.3  41.3  83.1  221.5 |
| Case 3  |            | 10 73 73 72 71  91.2  987.5  995.1  1500.6 |
|         |            | 50 73 73 72 71  203.5  1060.0  2140.0  3250.0 |
|         |            | 100 73 71 70 70  214.7  1080.0  2340.0  4240.0 |

The computation time to solve the problem is also presented in Table 2. The computation time obtained by the population and the number of iterations fluctuate from the experimental results. The experimental results show that the greater the population and the number of iterations desired, the optimal search solution increases.

Table 3 shows the comparison of the Efficiency Index (EI) with the PSO and GA algorithms. Based on the experimental results, PSO has an EI value of 94.4.63%. Furthermore, GA has an EI value of 96.8%. These results show that the HWOA algorithm has better performance than PSO and GA.

Table 3. Comparison of Algorithms

| Problem case | Job and Machine | Number Of Tardy Job | Efficiency Index (EI) % |
|--------------|-----------------|---------------------|-------------------------|
|              |                 | HWOA | PSO | GA | HWOA | PSO | GA |
| Case 1       | 10 job 6 machine | 8     | 8   | 8   | 100   | 100   | 100   |
| Case 2       | 20 job 10 machine | 14    | 16  | 15  | 100   | 87.5  | 93.3  |
| Case 3       | 100 job 10 machine | 70    | 73  | 72  | 100   | 95.8  | 97.2  |
| Average      |                 | 100   | 94.4 | 96.8 |

4. Conclusion

This study aimed to develop the HWOA algorithm to minimize the number of tardy jobs on the PFSSP problem. The proposed algorithm can solve the minimization of the number of tardy jobs. The results of the algorithm experiment show that the HWOA algorithm has better performance than PSO and GA. This study used three different cases. Therefore, future work is proposed to try several cases in some medium and large jobs.

References

[1] Utama D M, Widodo D S, Wicaksono W and Ardiansyah L R 2019 A New Hybrid Metaheuristics Algorithm for Minimizing Energy Consumption in the Flow Shop Scheduling Problem *International Journal of Technology* **10** 320-31
[2] Widodo D S and Utama D M 2020 The Hybrid Ant Lion Optimization Flow Shop Scheduling Problem for Minimizing Completion Time. In: *Journal of Physics: Conference Series*: IOP Publishing) p 022097
[3] Utama D M, Baroto T and Widodo D S 2020 Energy-Efficient Flow Shop Scheduling Using Hybrid Grasshopper Algorithm Optimization *Jurnal Ilmiah Teknik Industri* **19** 30-8
[4] Utama D M 2019 An Effective Hybrid Sine Cosine Algorithm to Minimize Carbon Emission on Flow-shop Scheduling Sequence Dependent Setup 2019 **20** 62-72
[5] Utama D M, Baroto T, Maharani D, Jannah F R and Octaria R A 2019 Algoritma ant-lion optimizer untuk meminimasi emisi karbon pada penjadwalan flow shop dependent sequence set-up 2019 **9** 69-78
[6] Aydilek A, Aydilek H and Allahverdi A 2017 Algorithms for minimizing the number of tardy jobs for reducing production cost with uncertain processing times Applied Mathematical Modelling 45 982-96

[7] Allahverdi A, Aydilek A and Aydilek H 2016 Minimizing the number of tardy jobs on a two-stage assembly flowshop Journal of Industrial and Production Engineering 33 391-403

[8] Chen W-J 2009 Minimizing number of tardy jobs on a single machine subject to periodic maintenance Omega 37 591-9

[9] Santos D and Roa I 2018 Minimizing The Number of Tardy Jobs in Hybrid Flow Shops with Non-Identical Multiple Processors Industrial and Systems Engineering Review 6 62-73

[10] Chen W-Y and Sheen G-J 2007 Single-machine scheduling with multiple performance measures: minimizing job-dependent earliness and tardiness subject to the number of tardy jobs International Journal of Production Economics 109 214-29

[11] Cheng T E, Ng C and Yuan J 2006 Multi-agent scheduling on a single machine to minimize total weighted number of tardy jobs Theoretical Computer Science 362 273-81

[12] Hermelin D, Karhi S, Pinedo M and Shabtay D 2018 New algorithms for minimizing the weighted number of tardy jobs on a single machine Annals of Operations Research 1-17

[13] Liu M, Wang S, Chu C and Chu F 2016 An improved exact algorithm for single-machine scheduling to minimise the number of tardy jobs with periodic maintenance International Journal of Production Research 54 3591-602

[14] Bajwa N, Melouk S and Bryant P 2019 A hybrid heuristic approach to minimize number of tardy jobs in group technology systems International Transactions in Operational Research 26 1847-67

[15] Su L-H, Hsiao M-C, Zhou H and Chou F-D 2018 Minimizing the number of tardy jobs on unrelated parallel machines with dirt consideration Journal of Industrial and Production Engineering 35 383-93

[16] Rajendran C and Ziegler H 2004 Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs European Journal of Operational Research 155 426-38

[17] Pan Q-K, Tasgetiren M F and Liang Y-C 2008 A discrete differential evolution algorithm for the permutation flowshop scheduling problem Computers & Industrial Engineering 55 795-816

[18] Vallada E and Ruiz R 2010 Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem Omega 38 57-67

[19] Iyer S K and Saxena B 2004 Improved genetic algorithm for the permutation flowshop scheduling problem Computers & Operations Research 31 593-606

[20] Ruiz R and Stützle T 2007 A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem European Journal of Operational Research 177 2033-49

[21] Fernandez-Viagas V and Framinan J M 2015 A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem International Journal of Production Research 53 1111-23

[22] Fernandez-Viagas V and Framinan J M 2015 NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness Computers & Operations Research 60 27-36

[23] Lian Z, Gu X and Jiao B 2006 A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan Applied mathematics and computation 175 773-85

[24] Utama D M, Ardiansyah L R and Garside A K 2019 Penjadwalan Flow Shop untuk Meminimasi Total Tardiness Menggunakan Algoritma Cross Entropy–Algoritma Genetika Journal Optimasi Sistem Industri 18 133-41

[25] Utama D M, Widodo D S, Ibrahim M F, Hidayat K, Baroto T and Yurifah A 2020 The hybrid whale optimization algorithm: A new metaheuristic algorithm for energy-efficient on flow
shop with dependent sequence setup. In: Journal of Physics: Conference Series: IOP Publishing) p 022094

[26] Abdel-Baset M, Manogaran G, El-Shahat D and Mirjalili S 2018 A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem Future Generation Computer Systems 85 129-45

[27] Mirjalili S and Lewis A 2016 The Whale Optimization Algorithm Advances in Engineering Software 95 51-67

[28] Li X and 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

[29] Carlier J 1978 Ordonnancements a contraintes disjonctives RAIRO-Operations Research 12 333-50

[30] Heller J 1960 Some numerical experiments for an M× J flow shop and its decision-theoretical aspects Operations Research 8 178-84