African Vultures Optimization Algorithm for Solving Flowshop Scheduling with Tardiness Objective

Muhammad Hisyam Taqiyyuddin1,a), Annisa Kesy Garside 1,b), Ikhlasul Amallynda 1,c)

1Industrial Engineering Department, Universitas Muhammadiyah Malang, Malang, Indonesia

a) hisyam.csk30@gmail.com

b) Corresponding author: annisa@umm.ac.id

c) ikhlasulamallynda@umm.ac.id

**Abstract.** Production scheduling is one of the most crucial aspects in manufacturing systems, enabling the on-time completion of jobs and maximizing resource utilization. This study addresses the Pure Flowshop Scheduling Problem (PFSP) to minimize total tardiness. A real case study from an Indonesian paint manufacturer is presented, in which the current scheduling rule—First Come First Served (FCFS)—frequently results in suboptimal performance. To counteract this, the Africa Vultures Optimization Algorithm (AVOA), a recent nature-inspired metaheuristic based on the foraging behavior of African vultures, is applied. The algorithm was evaluated with different population and iteration configurations. The results indicate that the AVOA is superior to the current company’s scheduling method, with the former resulting in a 37.2% reduction in total tardiness. This demonstrates the good performance of the algorithm and shows that the approach may be used as a practical tool for flowshop production scheduling.

**Keywords:** Production Scheduling, Pure Flowshop, African Vultures Optimization, Tardiness.

# INTRODUCTION

Production scheduling plays a crucial role in efficient operations, determining the optimal sequence of jobs to be scheduled on existing machines. Among the scheduling types, the Pure Flowshop Scheduling Problem (PFSP) has been extensively investigated, mainly when all jobs are processed in the same order on all machines [1]. An important measure of performance in PFSP is total tardiness, which is the sum of the deviations of each job from its due date. High levels of tardiness can lead to low customer satisfaction and incur expensive penalties [2].

However, in practice, many companies still use the simple rule-based scheduling method (e.g., FCFS), as it may only guarantee the priority of jobs placed first, rather than the optimal sequence [3]. Such methods may result in higher makespan and total tardiness, which has led researchers to use metaheuristic algorithms for better scheduling solutions. Recent advances have introduced techniques such as whale swarm algorithm [4], hybrid shuffle frog leaping algorithm based on cuckoo search [5], and hybrid whale optimization algorithm (WOA) with local search heuristics [6], which have been developed to address makespan and energy consumption objectives. Additionally, hybrid and multi-objective methods have emerged to balance multiple production objectives [7], [8].

**LITERATURE REVIEW**

Table 1 presents flowshop scheduling studies from 2018 to 2022, focusing on the objectives of makespan, total tardiness, and energy consumption. Remarkably, few papers have total tardiness as their primary objective, such as [9], which proposes several matheuristic algorithms, [10], which utilizes a Hybrid Discrete Water Wave Optimization Algorithm, or [11], [12], which employs an Iterated Greedy Algorithm. Other relevant approaches include Hybrid Discrete Harris Hawks Optimization [12] and Crossbreed Discrete Artificial Bee Colony [13].

The African Vultures Optimization Algorithm (AVOA) is a relatively new metaheuristic that simulates the cooperative and competitive foraging behavior of African vultures [14]. It has demonstrated competitive performance for a wide range of problem applications, such as parameter fitting in SOFC models [15] and the design of heat exchangers [16]. Additionally, comparative analyses indicate that AVOA yields superior results to other algorithms (e.g., PSO, GWO, and WOA) in various optimization problems [14].

**TABLE 1**. Previous studies on flowshop scheduling

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Author & Year** | **Objective Function** | | | | | | | **Solution Approach** | **Algorithm** |
| **Make-span** | **Tardi-ness** | **Energi Consu-mption** | **Noise polu-tion** | **Flow time** | **Adjus-ment time** | **Earli-ness** |
| Ta et al. (2018) [9] | - | ✓ | - | - | - | - | - | Matheuristics | Hybridization of a local search and an exact resolution method |
| Lu et al. (2018) [7] | ✓ | - | ✓ | ✓ | - | - | - | Hybrid | Hybrid multi-objective grey wolf algorithm |
| Zhong et al.(2018) [5] | ✓ | - | ✓ | - | - | - | - | Hybrid | Hybrid shuffle frog leaping algorithm based on cuckoo search |
| Fu et al. (2019) [8] | ✓ | ✓ | ✓ | - | - | - | - | Metahe-uristics | Multi-objective brain storm Optimization |
| Wang et al. (2019) [4] | ✓ | - | ✓ | - | - | - | - | Metahe-uristics | Multi-objective whale swarm |
| Ribas et al. (2019) [11] | - | ✓ | - | - | - | - | - | Metahe-uristics | Iterated greedy |
| Oztop et al. (2020) [17] | - | - | ✓ | - | ✓ | - | - | Metahe-uristics | Multi-Objective Iterated Greedy |
| Zhao et al. (2020) [10] | - | ✓ | - | - | - | - | - | Hybrid | Hybrid Discrete Water Wave Optimization |
| Wang et al. (2020) [18] | ✓ | - | ✓ | - | - | - | - | Exact | Epsilon-constraint algorithm integrated with L-shaped method |
| Lu et al. (2021) [19] | ✓ | - | ✓ | - | - | - | - | Hybrid | Hybridization of Iterated Greedy and an efficient local search |
| Khare & Agrawal (2021) [12] | - | ✓ | - | - | - | - | - | Heuris-tics, Hybrid, and Metahe-uristics | NEHedd, ESL, Hybrid discrete Harris Hawks Optimization (HHO) and Iterated Greedy (IG) Algorithm |
| Li et al. (2021) [6] | ✓ | - | ✓ | - | - | - | - | Hybrid | Hybridization of Whale optimization algorithm and local search |
| Mou et al. (2022) [20] | - | - | ✓ | - | - | ✓ | - | Hybrid | Effective Hybrid Collaborative Algorithm |
| Li et al. (2022) [21] | - | ✓ | - | - | - | - | - | Heuris-tics | Iterated Greedy Algorithms |
| Garside & Amallynda (2022) [13] | - | ✓ | - | - | - | - | ✓ | Metahe-uristics | Crossbreed Discrete Artificial Bee Colony |

Despite these advancements, to our knowledge, the AVOA has not yet been applied to PFSPs to minimize total tardiness. This study aims to fill this gap by adapting AVOA for solving a real-world case of pure flowshop scheduling in an Indonesian paint manufacturing company, which currently applies the FCFS rule. The main objective is to determine whether the AVOA can significantly reduce total tardiness and provide a practical alternative to the existing scheduling rule.

# METHODS

This study addresses the problem of minimizing the total tardiness in a pure flowshop scheduling with the African Vultures Optimization Algorithm (AVOA). The following sections describe the AVOA algorithm, the job permutation LRV discretization method, and the modified AVOA version, which is customized for solving scheduling problems.

**African Vultures Optimization Algorithm**

The African Vultures Optimization Algorithm (AVOA) is a nature-inspired metaheuristic introduced by [14], which emulates the cooperative and competitive searching behavior of African vultures. AVOA proceeds by splitting the population into two subpopulations, one governed by a dominant vulture chosen according to its fitness. The algorithm iteratively switches between exploration (global search) and exploitation (local search) based on the satiety level of the vultures. This satiety regulates the vulture's curiosity to discover new territories or return to rewarding ones.

*Phase 1: Leader Selection*

At the beginning of each iteration, the fitness of each vulture (solution candidate) is evaluated. The two top-scoring vultures are subsequently chosen as leaders (BestVulture1 and BestVulture2). Each individual references one of these leaders for guidance. The selection mechanism is governed by:

|  |  |
| --- | --- |
|  | (1) |

Where L1 and L2 are leadership probabilities such as L1 + L2= 1. The selection is based on roulette wheel probabilities calculated as:

|  |  |
| --- | --- |
| *pi =* | (2) |

*Phase 2: Satiety Level and Transition Mechanism*

The transition between exploration and exploitation phases is based on satiety indicator *F,* influenced by iteration count and random behavior. The intermediate variable *t* is first computed as:

|  |  |
| --- | --- |
| *t = h.* | (3) |
| *F = (*2 *. +* 1*). + t* | (4) |

Where *z* ∈[-1, 1], *h* ∈[-1, 1], and *rand*1 ∈[0, 1]. If , the algorithm performs exploration. Otherwise, it moves into the exploitation phase.

*Phase 3: Exploration Phase*

When , exploration is guided by two strategies chosen based on a probability *P*1:

|  |  |
| --- | --- |
| *P (i+*1*) =* | (5) |
| *P (i +* 1*) = R(i) – D(i). F* | (6) |
| *D(i) = X =* 2 *× rand* | (7) |
| *P (i +* 1*) = R(i) – F + . ((ub – lb) . + lb)* | (8) |

Where *lb* and *ub* are variable bounds.

*Phase 4: Exploitation Phase*

If < 1, exploitation is divided into two sub-phases depending on whether > 0.5 or ≤ 0.5. The first sub-phase switches between two strategies based on:

|  |  |
| --- | --- |
| *P (i +* 1*) =* | (9) |
| *P (i +* 1*) = D(i) . (F +* | (10) |
| *d(t) = R(i) – P(i)* | (11) |
| *= R(i) . . cos (P(i))*  *= R(i) . . sin (P(i))* | (12) |
| *P (i +* 1*) = R(i) – ()* | (13) |

If ≤ 0.5, the second sub-phase is executed:

|  |  |
| --- | --- |
| *P(i +* 1*) =* | (14) |
|  | (15) |
| *P (i +* 1) *=* | (16) |
| *P (i +* 1*) = R(i) – . F. LF(d)* | (17) |
| *LF (x) =* 0.01 *. , =* | (18) |

Where *β* = 1.5, and *u, v* ∈ [0, 1] are random values.

**Discretization using the Largest Rank Value**

The AVOA continuous solution vectors are transformed into job permutations for scheduling using the Largest Rank Value (LRV) procedure. The components of a vulture's position vector are sorted in descending order after every generation. The job index corresponding to the highest value is scheduled first, followed by the second-highest value, and so on [22]. This ranking mechanism ensures that a valid permutation exists and guarantees its uniqueness.

|  |  |
| --- | --- |
| A diagram of a number of objects  AI-generated content may be incorrect. | A diagram of a diagram  AI-generated content may be incorrect. |
| **FIGURE 1.** LRV Correct Job Permutation | **FIGURE 2.** LRV Incorrect Job Permutation |

Fig. 1 illustrates an example of a correctly generated mapping job permutation using the LRV method. In contrast, Fig. 2 presents an inappropriate mapping job permutation that occurs when the values are not distinct or properly ranked.

**AVOA for Flowshop Scheduling**

For minimization of total tardiness in the Pure Flowshop Scheduling Problem (PFSP), the AVOA approach is applied with LRV-based discretization. Every vulture has encoded into it a continuous vector, which is then converted into a job sequence through LRV. The total tardiness of the solution is used as a measure of fitness.

|  |
| --- |
| 1: Inputs: the population size N and maximum number of iterations T  2: Outputs: The location of Vulture and its fitness value  3: Initialize the random population (***i*** = 1,2, …, *N*)  4: **while** (stopping condition is not met) **do**  5: Apply LRV for changes best potition to job squences  6: Total Tardiness values of Vulture  7: Set , as the location of Vulture (First best location Best Vulture Category 1)  8: Set , as the location of Vulture (Second best location Best Vulture Category 2)  9: **for** (each Vulture ()) **do**  10: Select R(i) using equation (1)  11: Update the F using equation (4)  12: **if** **1**) **then**  13:  **if** ( ) **then**  14: Update the location Vulture using equation (6)  15: **else**  16: Update the location Vulture using equation (8)  17:  **if** **1**) **then**  18: **if** **0.5**) **then**  19:  **if** ( ) **then**  20: Update the location Vulture using equation (10)  21:  **else**  22: Update the location Vulture using equation (13)  23: **else**  24: **if** ( ) **then**  25: Update the location Vulture using equation (16)  26:  **else**  27: Update the location Vulture using equation (17)  Return |

**FIGURE 3**. Modification pseudo code African Vultures Optimization Algorithm

Fig. 3 presents the modified pseudocode of AVOA for scheduling purposes. In terms of satiety-based phase transition, the structure behaves in the same manner as the original algorithm, with the addition of a step for LRV transformation and tardiness calculation at each iteration. This adjustment makes the search dynamics of AVOA compatible with the discrete setting of job scheduling, strikes a good balance between global exploration and local exploitation, and maintains feasible solution permutations

# RESULTS AND DISCUSSION

**Scheduling Results Using AVOA**

The African Vultures Optimisation Algorithm (AVOA) was implemented to enhance the production scheduling system on PT. X, an Indonesian paint company. The company employs the First Come First Served (FCFS) rule, which often results in suboptimal job sequences and high tardiness due to poor scheduling decisions. To adress this, the AVOA algorithm was implemented using different population sizes (100 and 200) and iterations (50, 100, 200, 500, and 1000).

The numerical experiments were implemented using MATLAB R2018a. The optimal values of the parameters were obtained by conducting experiments for each population and iteration. The results are reported in Table 2, which shows the job sequence and total tardiness for each AVOA configuration. The best solutions were obtained using a population of 100 and 1000 iterations, which produced a total tardiness of 546.9 minutes. This setup generated the optimal job sequence among all considered configurations.

**TABLE 2**. Best Result from AVOA Algorithm

|  |  |  |  |
| --- | --- | --- | --- |
| **Population** | **Iteration** | **Job Sequence** | **Total Tardiness (minutes)** |
| 100 | 50 | JO4-JO1-JO2-JO3-JO5-JO6-JO7-JO8-JO9-JO10-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS2-JS3-JS5-JS7-JS8-JS9-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS12-JS14-JS15-JS16-JS17-JS18-JS19-JS20-JS10-JS4-JS11-JS6-JS13 | 585.9 |
| 100 | J02-JO8-JO1-JO3-JO4-JO5-JO6-JO7-JO9-JO10-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS2-JS3-JS5-JS7-JS8-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS11-JS12-JS13-JS14-JS15-JS16-JS17-JS18-JS19-JS20-JS9-JS6-JS4 | 571.9 |
| 200 | JO10-JO7-JO22-JO1-JO2-JO3-JO4-JO5-JO6-JO8-JO9-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO23-JO24-JS3-JS4-JS5-JS6-JS7-JS9-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JO37-JO38-JO39-JO40-JO41-JO42-JS18-JS19-JS20-JS1-JS17-JS8-JS2 | 618.1 |
| 500 | JO8-JO12-JO11-JO1-JO2-JO3-JO4-JO5-JO6-JO7-JO9-JO10-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS2-JS3-JS7-JS8-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JO37-JO38-JO39-JS14-JS15-JS17-JS18-JS19-JS20-JS4-JS5-JS1-JS6-JS16-JS9 | 597.1 |
| 1000 | JO2-JO11-JO7-JO10-JO1-JO3-JO4-JO5-JO6-JO8-JO9-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS3-JS4-JS5-JS6-JS7-JS9-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS11-JS12-JS13-JS14-JS15-JS16-JS17-JS18-JS19-JS20-JS8-JS2 | **546.9** |
| 200 | 50 | JO8-JO2-JO1-JO3-JO4-JO5-JO6-JO7-JO9-JO10-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS2-JS3-JS4-JS5-JS8-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS11-JS12-JS13-JS14-JS15-JS16-JS17-JS18-JS19-JS20-JS9-JS7-JS6 | 571.9 |
| 100 | JO16-JO11-JO6-JO3-JO1-JO2-JO4-JO5-JO7-JO8-JO9-JO10-JO12-JO13-JO14-JO15-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS2-JS3-JS4-JS6-JS7-JS9-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS11-JS12-JS13-JS14-JS15-JS17-JS18-JS19-JS20-JS8-JS16-JS5 | 655.9 |
| 200 | JO22-JO1-JO2-JO3-JO4-JO5-JO6-JO7-JO8-JO9-JO10-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO23-JO24-JS4-JS5-JS6-JS7-JS8-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO5-JO36-JS11-JS12-JS13-JS15-JS16-JS18-JS19-JS20-JS3-JS1-JS14-JS17-JS9-JS2 | 657.8 |
| 500 | JO10-JO11-JO1-JO2-JO3-JO4-JO5-JO6-JO7-JO8-JO9-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS1-JS2-JS3-JS4-JS5-JS6-JS7-JS8-JS9-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS11-JS12-JS13-JS14-JS15-JS16-JS17-JS18-JS19-JS20 | 638.7 |
| 1000 | JO2-JO4-JO3-JO6-JO1-JO5-JO7-JO8-JO9-JO10-JO11-JO12-JO13-JO14-JO15-JO16-JO17-JO18-JO19-JO20-JO21-JO22-JO23-JO24-JS3-JS5-JS6-JS7-JS8-JS9-JS10-JO25-JO26-JO27-JO28-JO29-JO30-JO31-JO32-JO33-JO34-JO35-JO36-JS12-JS13-JS14-JS15-JS16-JS17-JS18-JS19-JS20-JS2-JS1-JS4-JS11-JS17 | 571.9 |

**Scheduling Performance Comparison**

To evaluate the performance of the AVOA, it is compared to the company's scheduling method (FCFS). Table 3 shows that the toal of tardiness using the FCFS is 871.3 minutes, but it is reduced to 546.9 minutes using the AVOA. This represents a decrease of 324.4 minutes (a 37.2% increase in scheduling efficiency). These results demonstrate that scheduling sequences has a significant impact on total tardiness. With the new schedule generated with AVOA, PT. X will experience more efficient operations and better delivery time performance. Consequently, the AVOA shows excellent potential for use as decision support in reducing tardiness in practical flow shop scheduling problems.

**TABLE 3.** Total Tardiness Comparison

|  |  |  |
| --- | --- | --- |
| **Method** | **Job Sequence** | **Total Tardiness** |
| FCFS (Company) | JO1–JO2–JO3–JO4–JO5–JO6–JO7–JO8–JO9–JO10–JO11–JO12–JO13–JO14–JO15–JO16–JO17–JO18–JO19–JO20–JO21–JO22–JO23–JO24–JS1–JS2–JS3–JS4–JS5–JS6–JS7–JS8–JS9–JS10–JO25–JO26–JO27–JO28–JO29–JO30–JO31–JO32–JO33–JO34–JO35–JO36–JS11–JS12–JS13–JS14–JS15–JS16–JS17–JS18–JS19–JS20 | 871.3 |
| African Vultures Optimization Algorithm | JO2–JO11–JO7–JO10–JO1–JO3–JO4–JO5–JO6–JO8–JO9–JO12–JO13–JO14–JO15–JO16–JO17–JO18–JO19–JO20–JO21–JO22–JO23–JO24–JS1–JS3–JS4–JS5–JS6–JS7–JS9–JS10–JO25–JO26–JO27–JO28–JO29–JO30–JO31–JO32–JO33–JO34–JO35–JO36–JS11–JS12–JS13–JS14–JS15–JS16–JS17–JS18–JS19–JS20–JS8–JS2 | 546.9 |
| Difference | | 324.4 |
| Efficiency | | 37.2% |

# CONCLUSIONS

This study proposed the application of the African Vultures Optimisation Algorithm (AVOA) for solving the PFSP to minimize total tardiness. A case study was conducted at a manufacturing company, PT. X, which employs a First Come, First Served (FCFS) policy in production scheduling. The experimental results showed that AVOA-based scheduling significantly outperformed the existing method. The best configurations-i.e., population of 100 and 1000 iterations-resulted in a total tardiness of 546.9 minutes as opposed to 871.3 minutes with the company’s FCFS-based schedule. It results in a 37.2% decrease in total tardiness, verifying the potential of AVOA as a useful decision-assisting tool in scheduling for flowshop problems. The proposed algorithm can also be extended to address multi-objective scheduling problems and a dynamic environment with stochastic job arrivals, as well as machine availability in future. Additionally, hybridizing AVOA with local search methods could further enhance solution quality and convergence speed.

# References

[1] K. R. Baker and D. Trietsch, *Principles of Sequencing and Scheduling*, 2nd ed. John Wiley & Sons, Inc., 2019.

[2] D. M. Utama, L. R. Ardiansyah, and A. K. Garside, “Penjadwalan Flow Shop untuk Meminimasi Total Tardiness Menggunakan Algoritma Cross Entropy–Algoritma Genetika,” *J. Optimasi Sist. Ind.*, vol. 18, no. 2 SE-Articles, pp. 133–141, Oct. 2019, doi: 10.25077/josi.v18.n2.p133-141.2019.

[3] M. R. Fadli and W. Sulistiyowati, “Optimization of Pipe Production Scheduling in Line 18 Using First Come First Serve (Fcfs), Earlier Due Date (Edd), Short Process Time (Spt) Methods (Case Study: Pt Wtur),” *PROZIMA (Productivity, Optim. Manuf. Syst. Eng.*, vol. 3, no. 2 SE-Articles, Mar. 2021, doi: 10.21070/prozima.v3i2.1268.

[4] G. Wang, X. Li, L. Gao, and P. Li, “A Multi-Objective Whale Swarm Algorithm for Energy-Efficient Distributed Permutation Flow shop Scheduling Problem with Sequence Dependent Setup Times,” *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 235–240, 2019, doi: https://doi.org/10.1016/j.ifacol.2019.11.142.

[5] L.-C. Zhong, B. Qian, R. Hu, and C.-S. Zhang, “The Hybrid Shuffle Frog Leaping Algorithm Based on Cuckoo Search for Flow Shop Scheduling with the Consideration of Energy Consumption BT - Intelligent Computing Theories and Application,” D.-S. Huang, V. Bevilacqua, P. Premaratne, and P. Gupta, Eds., Cham: Springer International Publishing, 2018, pp. 649–658.

[6] Q. Li, J. Li, X. Zhang, and B. Zhang, “A wale optimization algorithm for distributed flow shop with batch delivery,” *Soft Comput.*, vol. 25, no. 21, pp. 13181–13194, 2021, doi: 10.1007/s00500-021-06099-0.

[7] C. Lu, L. Gao, X. Li, J. Zheng, and W. Gong, “A multi-objective approach to welding shop scheduling for makespan, noise pollution and energy consumption,” *J. Clean. Prod.*, vol. 196, pp. 773–787, 2018, doi: https://doi.org/10.1016/j.jclepro.2018.06.137.

[8] Y. Fu, G. Tian, A. M. Fathollahi-Fard, A. Ahmadi, and C. Zhang, “Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint,” *J. Clean. Prod.*, vol. 226, pp. 515–525, 2019, doi: https://doi.org/10.1016/j.jclepro.2019.04.046.

[9] Q. C. Ta, J.-C. Billaut, and J.-L. Bouquard, “Matheuristic algorithms for minimizing total tardiness in the m-machine flow-shop scheduling problem,” *J. Intell. Manuf.*, vol. 29, no. 3, pp. 617–628, 2018, doi: 10.1007/s10845-015-1046-4.

[10] F. Zhao, L. Zhang, Y. Zhang, W. Ma, C. Zhang, and H. Song, “A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion,” *Expert Syst. Appl.*, vol. 146, p. 113166, 2020, doi: https://doi.org/10.1016/j.eswa.2019.113166.

[11] I. Ribas, R. Companys, and X. Tort-Martorell, “An iterated greedy algorithm for solving the total tardiness parallel blocking flow shop scheduling problem,” *Expert Syst. Appl.*, vol. 121, pp. 347–361, 2019, doi: https://doi.org/10.1016/j.eswa.2018.12.039.

[12] A. Khare and S. Agrawal, “Effective heuristics and metaheuristics to minimise total tardiness for the distributed permutation flowshop scheduling problem,” *Int. J. Prod. Res.*, vol. 59, no. 23, pp. 7266–7282, Dec. 2021, doi: 10.1080/00207543.2020.1837982.

[13] A. K. Garside and I. Amallynda, “A Crossbreed Discrete Artificial Bee Colony for Permutation Flow Shop Scheduling Problem to Minimize Total Earliness and Tardiness,” *Int. J. Intell. Eng. Syst.*, vol. 15, no. 1, pp. 441–452, 2022, doi: 10.22266/IJIES2022.0228.40.

[14] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, “African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems,” *Comput. Ind. Eng.*, vol. 158, p. 107408, 2021, doi: https://doi.org/10.1016/j.cie.2021.107408.

[15] H. A. Bagal, Y. N. Soltanabad, M. Dadjuo, K. Wakil, M. Zare, and A. S. Mohammed, “SOFC model parameter identification by means of Modified African Vulture Optimization algorithm,” *Energy Reports*, vol. 7, pp. 7251–7260, 2021, doi: https://doi.org/10.1016/j.egyr.2021.10.073.

[16] D. Gürses, P. Mehta, S. M. Sait, and A. R. Yildiz, “African vultures optimization algorithm for optimization of shell and tube heat exchangers,” vol. 64, no. 8, pp. 1234–1241, 2022, doi: doi:10.1515/mt-2022-0050.

[17] 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, 2020, doi: https://doi.org/10.1016/j.eswa.2020.113279.

[18] S. Wang, S. J. Mason, and H. Gangammanavar, “Stochastic optimization for flow-shop scheduling with on-site renewable energy generation using a case in the United States,” *Comput. Ind. Eng.*, vol. 149, p. 106812, 2020, doi: https://doi.org/10.1016/j.cie.2020.106812.

[19] C. Lu, L. Gao, J. Yi, and X. Li, “Energy-Efficient Scheduling of Distributed Flow Shop With Heterogeneous Factories: A Real-World Case From Automobile Industry in China,” *IEEE Trans. Ind. Informatics*, vol. 17, no. 10, pp. 6687–6696, 2021, doi: 10.1109/TII.2020.3043734.

[20] J. Mou, P. Duan, L. Gao, X. Liu, and J. Li, “An effective hybrid collaborative algorithm for energy-efficient distributed permutation flow-shop inverse scheduling,” *Futur. Gener. Comput. Syst.*, vol. 128, pp. 521–537, 2022, doi: https://doi.org/10.1016/j.future.2021.10.003.

[21] Y.-Z. Li, Q.-K. Pan, R. Ruiz, and H.-Y. Sang, “A referenced iterated greedy algorithm for the distributed assembly mixed no-idle permutation flowshop scheduling problem with the total tardiness criterion,” *Knowledge-Based Syst.*, vol. 239, p. 108036, 2022, doi: https://doi.org/10.1016/j.knosys.2021.108036.

[22] M. Abdel-Basset, G. Manogaran, D. El-Shahat, and S. Mirjalili, “RETRACTED: A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem,” *Futur. Gener. Comput. Syst.*, vol. 85, pp. 129–145, 2018, doi: https://doi.org/10.1016/j.future.2018.03.020.