The Hybrid Ant Lion Optimization Flow Shop Scheduling Problem for Minimizing Completion Time

Dian Setiya Widodo¹ and Dana Marsetiya Utama²

¹Department Manufacturing Technology University of 17 Agustus 1945 Surabaya, Jl. Semolowaru No. 45, 60118 Surabaya, East Java, Indonesia
²Department Industrial Engineering University of Muhammadiyah Malang, Jl. Tlogomas No. 246, 65144 Malang, East Java, Indonesia

Abstract. This article aims to develop the Ant Lion Optimization (ALO) algorithm for the Permutation Flow Shop Scheduling Problem (PFSP). We use the objective function to minimize completion time. We propose the Hybrid Ant Lion Optimization (HALO) algorithm to minimize completion time in PFSP. We offer one of the total search agents on HALO using the NEH algorithm. Determination of the position of the search agent using a Large Rating Value (LRV). How it works from LRV is sorting from the largest value to the value. To improve the solution for each iteration, we also use swap, flip, and slide exchange procedures. The performance of the HALO algorithm is measured by Efficiency Index Percentage (EIP). HALO algorithm will be compared with several other algorithms, namely ALO algorithm and Hybrid Whale Optimization Algorithm (HWOA). we conducted numerical experiments with Ten variations of experimental data are used to show the performance of the HALO algorithm. Based on numerical experiments, the HALO algorithm is useful for minimizing completion time.

Keyword: ALO; HALO; Flow shop; Scheduling;

1. Introduction

Scheduling is an essential problem in the manufacturing industry [1]. Scheduling is the allocation of resources for completion of work on several machines [2]. One of the problems with scheduling is the permutation flow shop (PFSP). In PFSP, n jobs are processed on m machines in the same order [3, 4]. In recent years, the problem of PFSP has attracted the attention of new researchers. PFSP is categorized as an NP-hard problem [5]. Generally, scheduling criteria are minimizing completion time. Completion time shows the time to complete all jobs on the machine [6].

Several algorithms for PFSP have been extensively studied. Furthermore, several studies have also been published on this issue. PFSP was first introduced by Johnson [7] for optimal scheduling problems with two machines and three machines (particular case). He is a pioneer figure in the scheduling field. Furthermore, Nawaz, et al. [8] introduced a heuristic algorithm to handle n-jobs and m-machines in PFSP. They compared 15 other algorithms with the proposed algorithm. After NEH proved its effectiveness, several other researchers have used NEH to develop new algorithms. Marichelvam, et al. [9] combined NEH with the Improved Cuckoo Search (ICS) algorithm. Furthermore, Marichelvam, et al. [10] proposed the Hybrid Monkey Search Algorithm (MSA). They integrated NEH with the MSA algorithm. Liu, et al. [11] proposed a useful Differential Evolution (DE) algorithm with a combination of NEH.

Some other metaheuristic algorithms for solving PFSP include Gravitational Emulation Local Search algorithm [12], Chemical Reaction Optimization algorithm [13], Iterated Greedy algorithm [14], Differential Evolution algorithm [15]. Several new metaheuristic hybrid algorithms have also been
proposed to solve PFSP. Some of these algorithms are Hybrid Backtracking Search Algorithm [16], Hybrid Multiobjective Evolutionary Algorithm [17], Hybrid Crow Search Algorithm [18], Hybrid Firefly Variable Neighborhood Search Algorithm [19] Hybrid Whale Optimization Algorithm (HWOA) [20]. Many new metaheuristic algorithms, because previous studies have been successful in completing PFSP. Therefore, this study was inspired by successful hybrid strategies and applications in previous studies.

Recently, nature-based optimization algorithms have been widely proposed in the literature. Most of the proposed algorithms are effective for solving problems. There are many methods available to resolve PSFP problems. However, we propose the Hybrid Ant Lion Optimization (HALO) algorithm. Ant Lion Optimization (ALO) was introduced by Mirjalili [21]. This algorithm is inspired by antlion behavior in food hunting. At present, some ALO research is still minimal. Dubey, et al. [22] used ALO to complete scheduling hydroelectric power. Furthermore, Petrović, et al. [23] uses the ALO algorithm in the case of planning and scheduling. Our proposed algorithm (HALO) is a development of the ALO algorithm. As far as we know, there has been no previous research that developed the ALO algorithm in PFSP. In this article, the NEH algorithm replaces one search agent from the ALO algorithm. To comparison performance of the algorithm, the HALO algorithm is tested through computational experiments with several other algorithms.

2. Methods

2.1 Assumptions Problems and notations

In the flow shop scheduling, the order processing system for each job is fully determined, and each job visits the work station in the same order. The goal that can be achieved through the flow shop scheduling is to minimize completion time. Some assumptions in the flow shop scheduling are (1) All machines are ready to do the work at \( t = 0 \); (2) Removal time is considered zero; (3) There is no interruption to do other work in the middle of production; (4) Every machine that is used is always ready to do the work without interruption such as breakdown; (5) Each machine can only process 1 job; (6) Setup time is included in the processing time.

In this article, the mathematical model used is taken from the Mixed Integer Programming (MIP) model that has been developed [24]. The objective function of this PFSP problem is to minimize completion time. Following is the formula from PFSP. Parameter; \( Pri \) = Processing time of job \( i \) on machine \( r \)

Objective function:

\[
\begin{align*}
\text{Minimize } F_{\text{max}} \\
\text{constraints:} \\
\sum_{i=1}^{n} Z_{i} = 1 & \quad i = 1, \ldots, n \quad (2) \\
\sum_{i=1}^{n} Z_{i} = 1 & \quad j = 1, \ldots, n \quad (3) \\
\sum_{i=1}^{n} P_{r} z_{j+i} + y_{j+1} + x_{j+1} = y_{j} + \sum_{i=1}^{n} P_{r} z_{i} + x_{j+1} & \quad (4) \\
\sum_{i=1}^{n} z_{i} + y_{j} = f_{j} & \quad (5) \\
\sum_{j=1}^{m} z_{j} = x_{j} & \quad k = 2, \ldots, m \quad (6) \\
y_{k} & \quad k = 1, \ldots, m - 1 \quad (7)
\end{align*}
\]

The notation used in this article is:

- \( i, j : 1, ..., n \) Job index
- \( k, r : 1, ..., m \) Engine index

Decision variable:

- \( Z_{ij} : \) \( 1 \), where 1, if job \( i \) is located at position \( j \) in permutation and 0, vice versa.
- \( F_{\text{max}} : \) The objective function minimizes completion time.
- \( x_{j} : \) idle time on machine \( k \) before the job in the \( j \)-position starts in the permutation.
- \( y_{j} : \) waiting time for job \( j \)-position in permutation, after the process on machine \( k \) is complete, and waiting for machine \( k + 1 \) is free to do the next job.

Equations 2 and 3 are used to place jobs and determine the permutations of existing jobs. Equation 4 explains the calculation of the Gantt chart. Equation 5 is used to calculate the completion time. Equation 6 explains the calculation of the idle time of the second-order machine and the next machine. Equation 7 is used to ensure that the first-order job is done on the next machine.
2.2 Hybrid Ant Lion Optimization Algorithm (HALO)

We agreed to improve the Ant Lion Optimization Algorithm (ALO) for PFSP approval. The ALO algorithm also has a feature for completing flow shop scheduling. The algorithm we call Hybrid Ant Lion Optimization (HALO) We added the NEH algorithm and local search to improve ALO performance. In this article, the HALO algorithm is a combination of ALO algorithms with local search strategies such as swap, flip, and slide. In addition, in this article, the NEH algorithm replaces one search agent on ALO. There are five steps in the HALO algorithm: initialization of position for search agents, using LRV for the position, search for search agents using NEH algorithm, evaluating ALO, using swap, flip, and slide. Pseudo-code algorithm presented is presented in algorithm 1.

2.3 Initialization of search agent

Initialization of the position of the search agent generated randomly. It must be ensured that there is no similar loop in the initialization of the search agent position. Positioning in the form of a matrix is a line indicating the number for a search agent, while a column indicates a dimension. The dimension referred to in the position initialization matrix is the number of jobs.

| Algorithm 1 Pseudo-code Algorithm Hybrid Ant Lion Optimization (HALO): |
|---|
| 1. Initialize the first population of ants and antlion randomly |
| 2. Apply LRV on each search agent to be mapped into job permutation |
| 3. Solve the PFSSP using NEH |
| 4. Choose about one search agents from the population and replace them with NEH |
| 5. Calculate the fitness function values for ants and ant lions |
| 6. Select the fittest antlion as the elite |
| 7. While the iteration number is smaller than the maximum iteration number |
| 8. For each ant |
| 9. Select an antlion by utilizing the roulette wheel operator |
| 10. Update the lower and upper bounds |
| 11. Update γ and δ using |
| 12. Generate two random walks around the roulette selected antlion and the elite and normalize them |
| 13. Update the ant positions |
| 14. End for |
| 15. Apply LRV on each search agent to be mapped into job permutation |
| 16. Calculate the fitness function values for all ants and replace them in the corresponding matrix |
| 17. Replace an antlion with its corresponding ant |
| 18. Update the elite value if an antlion’s fitness function value becomes better than the elite |
| 19. if $W^* < W$ |
| 20. $W = W^*$ |
| 21. end if |
| 22. Apply Local Search |
| 23. For $i = 1 : n(n - 1)/2$ |
| 24. Perform swap on the $W^*$. Ensure no repeated swap in the $W^*$ |
| 25. if $W^* < W$ |
| 26. $W^* = W$ |
| 27. end if |
| 28. End for |
| 29. For $j = 1 : n(n - 1)/2$ |
| 30. Perform flip on the $W^*$. Ensure no repeated swap in the $W^*$ |
| 31. if $W^* < W$ |
| 32. $W = W^*$ |
| 33. end if |
| 34. End for |
| 35. For $k = 1 : n^2$ |
| 36. Perform slide on the $W^*$. Ensure no repeated swap in the $W^*$ |
| 37. if $W^* < W$ |
| 38. $W^* = W^*$ |
| 39. end if |
| 40. End for |
| 41. End while |
| 42. Report the best solution (elite) |
2.4 Application Large Rank Value (LRV)
In this article, determining the position of search agents uses Large Rank Value (LRV). How it works from LRV is sorting from the largest to the smallest value.

2.5 Nawaz Enscore Ham (NEH) Algorithm
Nawaz Enscore Ham (NEH) is a heuristic algorithm developed by Muhammad Nawaz, E. Emory Enscore Jr., and Inyong Ham in 1983. The workings of the NEH method is jobs with the largest total processing time are given top priority to be done [8]. Initialize NEH job sequences descending based on the total processing time of each job. After that, determine the best sequence for each job position (see algorithm 2). NEH outperformed 15 other heuristic methods significantly in the field of flow shop scheduling [25].

2.6 The local search
Local search is a combinatorial optimization method that changes the order of the initial solution set by iteration until the objective function is found [26]. Regarding each iteration, the solution is repaired with no further improvement. An effective metaheuristic algorithm is obtained by continuing the systematic change in local search [27]. The local search steps used swap, flip, and do slide. Swap by exchanging two genes randomly. The swap operation is repeated \( n (n - 1) / 2 \). Flip by reversing the order of the selected genes. The flip operation is repeated \( n (n - 1) / 2 \). Perform the slide done by shifting genes. The slide operation is repeated \( n^2 \).

2.7 Ant Lion Optimization Algorithm (ALO)
The antlion algorithm is a metaheuristic algorithm that is inspired by the intelligence of antlion behavior in food hunting in nature [21]. In the ALO algorithm, there are 2 variables, ant and antlion. Ants are search agents in the decision room. Antlions hide in the decision room. Antlion can hunt them down and catch their position to be fit. To evaluate antlions and ants, the fitness function of them is calculated during the optimization process and stored in the matrix. In this section, the best antlion is the antlion which has the best fitness in accordance with the objective function which is then selected as an elite. The next step, the ALO algorithm needs to use the roulette wheel to select the antlion according to their fitness during optimization. This mechanism provides a high chance for fitter antlions to catch ants by building traps. It should be noted that in the ALO algorithm, each ant can fall into only one antlion trap in each iteration. The roulette wheel chosen as the antlion for each ant is the antlion that has trapped the ant. By using the roulette wheel, a good fitness function solution has more chance to choose, because antlions with bigger traps can hunt more ants. When the ant falls into a trap, the antlion starts shooting sand out to launch the ant down while the ant is trying to escape. This behavior is modeled mathematically by shrinking the radius of the ant road at random. So the limit range for all decision variables is reduced and updated.

2.8 Experimental procedure
The proposed HALO algorithm needs to be evaluated by conducting several randomized experiments. The data used in this experiment are the number of machines, number of jobs, operating time. The operating time data used in this study uses the processing time data from Carlier [28], Heller [29], and Reeves [30]. The total number of job variations used is 10. In metaheuristic algorithms parameters affect the objective function. In this article, the parameters used are 3 population numbers including populations 10, 50, and 100. Moreover, 3 number of iterations include iterations of 10, 50, and 100. Each number of jobs and the number of machines in the group is tried on each parameter. Experiments were conducted to determine the effect of population and iteration on minimum completion time.

To find out the effectiveness of the HALO algorithm, this algorithm was compared with several other algorithms such as the ALO Mirjalili algorithm [21] and the Hybrid Whale Optimization Algorithm (HWOA) [20]. Numerical test trial of the hybrid ant lion optimization algorithm using Matlab R16 software. The performance of the HALO algorithm is measured by the Efficiency Index Percentage (EIP). The EIP formula is presented in equation (8). If the EIP value> 100%, the proposed algorithm is worse, EIP = 100% the performance of the proposed algorithm with the same comparison, and EIP <100%, the proposed algorithm is better.

\[
EIP = \frac{p}{a} \times \frac{a}{al} \times 100\% \tag{8}
\]
3. Results and discussion

3.1 Experiment Parameters and Comparison of the Hybrid Ant Lion Optimization Algorithm with Other Algorithms.

Table 2 shows the experimental parameters of the population and iteration used. From the results of the comparison, a large population and iteration can improve the performance of the objective function. In the case of small jobs, using population numbers and large iterations does not effectively solve the problem. However, for the case of large numbers of jobs, completion using a population and significant iteration is useful.

3.2 Efficiency Index Percentage (EIP) Hybrid Ant Lion Optimization (HALO) Algorithm with Other Algorithms

Table 1 shows a comparison of the Efficiency Index Percentage (EIP). EIP value is a comparison between the proposed algorithm and the comparison algorithm, from the test results obtained the average percentage of EIP on the comparison algorithm <100%. It was concluded that the proposed HALO algorithm is better than the HWOA and ALO algorithms.

Table 1 Experiments for Population & Iteration Parameters

| Case       | n job | m machine | Population 10 | Population 50 | Population 100 |
|------------|-------|-----------|---------------|----------------|----------------|
|            |       |           | Iteratio n 10 | Iteratio n 50  | Iteratio n 100  |
|            |       |           | n 10          | n 50           | n 100          |
| Carlie r [28] | 11    | 5         | 7038          | 7038           | 7038           |
| 13         | 4     | 7166      | 7166          | 7166           | 7166           |
| 12         | 5     | 7312      | 7312          | 7312           | 7312           |
| 14         | 4     | 8003      | 8003          | 8003           | 8003           |
| 10         | 6     | 7767      | 7767          | 7821           | 7821           |
| 8          | 9     | 8813      | 8754          | 8813           | 8868           |
| 7          | 7     | 6590      | 6645          | 6643           | 6590           |
| Heller [29] | 20    | 10        | 141           | 141            | 141            |
| Reeve s [30]| 20    | 5         | 1279          | 1249           | 1249           |

Table 2 Comparison of Efficiency Index Percentage (EIP)

| n job | m machine | HWOA[20] | ALO   |
|-------|-----------|----------|-------|
| 11    | 5         | 100%     | 100%  |
| 13    | 4         | 100%     | 100%  |
| 12    | 5         | 100%     | 96.9% |
| 14    | 4         | 100%     | 100%  |
| 10    | 6         | 99.5%    | 98.8% |
| 8     | 9         | 100%     | 99.2% |
| 7     | 7         | 100%     | 100%  |
| 8     | 8         | 100%     | 100%  |
| 20    | 10        | 98.6%    | 96.5% |
| 20    | 5         | 98.3%    | 99%   |
| Average Percentage | 99.64% | 99.04% |

4. Conclusion

In this article, the Hybrid Ant Lion Optimization (HALO) algorithm is combined with the NEH stages, and local search strategies such as swap, flip, and slide. LRV is used for assigning search agent position values from the largest to the smallest and converted into job sequences. The HALO algorithm is studied to minimize completion time. Furthermore, the HALO algorithm is compared with several other algorithms. From the comparison results, the HALO algorithm is proven to be better than other algorithms in some cases, the number of jobs and the number of machines. We propose that in the future work, the HALO algorithm can be tried to be combined with other metaheuristic algorithms to obtain optimal results.
5. References

[1] 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," International Journal of Technology, vol. 10, pp. 320-331, 2019.

[2] R. Ginting, Penjadwalan Mesin. Yogyakarta: Graha Ilmu, 2009.

[3] D. Marsetiya Utama, "An Effective Hybrid Sine Cosine Algorithm to Minimize Carbon Emission on Flow-shop Scheduling Sequence Dependent Setup," 2019, vol. 20, p. 10, 2019-02-26 2019.

[4] D. M. Utama, T. Baroto, D. Maharani, F. R. Jannah, and R. A. Octaria, "Algoritma ant-lion optimizer untuk meminimasi emisi karbon pada penjadwalan flow shop dependen sequence set-up," 2019, vol. 9, p. 10, 2019-06-28 2019.

[5] D. S. Johnson and M. R. Garey, Computers and intractability: A guide to the theory of NP-completeness vol. 1: WH Freeman San Francisco, 1979.

[6] K. R. Baker, Introduction to sequencing and scheduling; John Wiley & Sons, 1974.

[7] S. M. Johnson, "Optimal two-and three-stage production schedules with setup times included," Naval research logistics quarterly, vol. 1, pp. 61-68, 1954.

[8] M. Nawaz, E. E. Enscore Jr, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," Omega, vol. 11, pp. 91-95, 1983.

[9] M. Marichelvam, T. Prabaharan, and X.-S. Yang, "Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan," Applied Soft Computing, vol. 19, pp. 93-101, 2014.

[10] M. Marichelvam, Ö. Tosun, and M. Geetha, "Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time," Applied Soft Computing, vol. 55, pp. 82-92, 2017.

[11] Y. Liu, M. Yin, and W. Gu, "An effective differential evolution algorithm for permutation flow shop scheduling problem," Applied Mathematics Computation, vol. 248, pp. 143-159, 2014.

[12] R. Sanjeev Kumar, K. Padmanaban, and M. Rajkumar, "Minimizing makespan and total flow time in permutation flow shop scheduling problems using modified gravitational emulation local search algorithm," Journal of Engineering Manufacture, vol. 232, pp. 534-545, 2018.

[13] H. Bargaoui, O. B. Driss, and K. Ghédira, "A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion," Computers Industrial Engineering, vol. 111, pp. 239-250, 2017.

[14] J. Dubois-Lacoste, F. Pagnozzi, and T. Stützle, "An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem," Computers Operations Research, vol. 81, pp. 160-166, 2017.

[15] W. Shao and D. Pi, "A self-guided differential evolution with neighborhood search for permutation flow shop scheduling," Expert Systems with Applications, vol. 51, pp. 161-176, 2016.

[16] P. Chen, L. Wen, R. Li, and X. Li, "A hybrid backtracking search algorithm for permutation flow-shop scheduling problem minimizing makespan and energy consumption," in 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2017, pp. 1611-1615.

[17] W. Zhang, Y. Wang, Y. Yang, and M. Gen, "Hybrid multiobjective evolutionary algorithm based on differential evolution for flow shop scheduling problems," Computers Industrial Engineering, vol. 130, pp. 661-670, 2019.

[18] K.-W. Huang, A. S. Girsang, Z.-X. Wu, and Y.-W. Chuang, "A Hybrid Crow Search Algorithm for Solving Permutation Flow Shop Scheduling Problems," Applied Sciences, vol. 9, p. 1353, 2019.

[19] A. Migdalas, "A Hybrid Firefly-VNS Algorithm for the Permutation Flowshop Scheduling Problem," in Lecture Notes in Computer Science, 2019, pp. 274-286.

[20] M. Abdel-Basset, G. Manogaran, D. El-Shahat, and S. Mirjalili, "A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem," Future Generation Computer Systems, vol. 85, pp. 129-145, 2018.
[21] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80-98, 2015.

[22] H. M. Dubey, M. Pandit, and B. K. Panigrahi, "Ant lion optimization for short-term wind integrated hydrothermal power generation scheduling," *International Journal of Electrical Power & Energy Systems*, vol. 83, pp. 158-174, 2016/12/01/ 2016.

[23] M. Petrović, J. Petronijević, M. Mitić, N. Vuković, Z. Miljković, and B. Babić, "The ant lion optimization algorithm for integrated process planning and scheduling," in *Applied Mechanics and Materials*, 2016, pp. 187-192.

[24] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz, *Scheduling computer and manufacturing processes*: springer science & Business media, 2013.

[25] Y. B. Park, C. D. Pegden, and E. E. R. Enscore, "A survey and evaluation of static flowshop scheduling heuristics," *The International Journal of Production* vol. 22, pp. 127-141, 1984.

[26] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers operations research*, vol. 24, pp. 1097-1100, 1997.

[27] G. Laporte, M. Gendreau, J. Y. Potvin, and F. Semet, "Classical and modern heuristics for the vehicle routing problem," *International transactions in operational research*, vol. 7, pp. 285-300, 2000.

[28] J. Carlier, "Ordonnancements a contraintes disjonctives," *RAIRO-Operations Research*, vol. 12, pp. 333-350, 1978.

[29] J. Heller, "Some numerical experiments for an Mx J flow shop and its decision-theoretical aspects," *Operations Research*, vol. 8, pp. 178-184, 1960.

[30] C. R. Reeves, "A genetic algorithm for flowshop sequencing," *Computers Operations Research*, vol. 22, pp. 5-13, 1995.