Advances in Industrial Engineering, June 2024, 58(1): 13-36
DOI: 10.22059/aie.2024.345213.1844

RESEARCH PAPER

Flexible Flowshop Scheduling Problem Considering Manpower Skill-based Processing Times

Seyed Mohammad Hassan Hosseini[1], Hossein Amoozad Khalili[2*], Moujan Shirali[3]

[1]*Associate Professor, Department of Industrial Engineering and Management, Shahrood University of Technology, Shahrood, Iran.*
[2]*Assistant Professor, Department of Industrial Engineering, Sari Branch, Islamic Azad University, Sari, Iran.*
[3]*M.Sc., Department of Management and Economics, Science and Research Branch, Islamic Azad University, Tehran, Iran.*

Received: 30 June 2022, Revised: 04 February 2024, Accepted: 09 February 2024

Abstract

Scheduling for flexible flowshop environments is generally limited by resources such as manpower and machines. However, the majority of efforts tackle machines as the only constrained resource. This paper aims to investigate the problem of scheduling in flexible flowshop environments considering different skills as human resource constraints to minimize the total completion time. In this way, a mathematical model of complex integer linear programming is presented for solving small-sized problems in a reasonable computational time. In addition, due to the NP-hard nature of the problem, a whale hybrid optimization algorithm is tuned to solve the problem in large-sized dimensions. In order to evaluate the performance of the proposed optimization algorithm, the results are compared with five known optimization algorithms in the research background. All evaluations and results show the good performance of the whale hybrid algorithm. Especially, the final solution of the proposed algorithm shows a 0.75% deviation of the best solution in solving different instances on large-scale sizes. However, the genetic algorithm, memetic global and local search algorithm, and hybrid salp swarm algorithm are in the next ranks with 3.31, 3.52, and 4.02 percent respectively. In addition, proper discussions and managerial insights are provided for the relevant managers.

Keywords:
flexible flowshop, scheduling, meta-heuristic algorithm, manpower skills

Introduction

Job scheduling is one of the most important activities in production systems. Finding the best schedule can be very easy or very difficult depending on the production environment, performance indicators, and process constraints (Ramezanian and Hallaji, 2021). One of the most famous scheduling problems is the flexible flowshop scheduling problem (Ruiz and Vazquez-Rodriguez, 2010), which has many applications in real-world industries, such as label companies (Lin and Liao, 2003), Semiconductor industry (Quad and Cohen, 2005), tile production (Ruiz and Maruto, 2006), steel manufacturing process (Nakhaeinejad, 2019), etc.

* Corresponding author: (Hossein Amoozad Khalili)
Email: amoozad92@yahoo.com

In most scheduling problems, machines are supposed as the only constrained source. However, other resources such as manpower are limited in the real world of production, and consequently, it is illogical to consider sufficient manpower for the job process (Hasani and Hosseini, 2020). In the world of practical production, many manufacturing firms face rising wage costs and better use of manpower is essential (Lee and Goo, 2014). In many resources, the actual workforce in the flexible flowshop environment is multi-skill, that is, each workforce specializes in one or more skills and can be sent to different production stages. To achieve a feasible schedule with optimal performance conditions and efficiency, the manpower must be trained in different skills and sent to the source of different machines in accordance with their skills (Lee, Huang, and Niu, 2016). Therefore, it is necessary to examine the limited dual resources of manpower and machinery by job scheduling in the flexible flowshop environment, in which the manpower source has different functional skills and efficiencies and according to their skills to different production stages. Most studies have been conducted to minimize the maximum completion time, which includes 60% of the articles published in the combined flowshop literature (Hasani and Hosseini, 2022). In this paper, we study the job schedule in the flexible flowshop environment to minimize the total completion time considering limitations for machines and human resources so that workers have different skills and mobility between stations.

In the proposed problem, we present a mathematical model for job scheduling taking into account manpower skills as well as moving manpower between stations. Are considered. The proposed mathematical model in Gomez software solves the small size of the problem with the Cplex server.

Given that Gupta (1988) has proven that the problem of flexible flowshop scheduling is an NP-hard problem. Therefore, the more complex issue of scheduling jobs in the production environment with flexible flowshop technology, given the innovation in human resource skills, remains NP-hard. When faced with optimization of such complex problems, it is very difficult to use precise methods to optimize at computational time. Many optimization algorithms have been studied in the problem of scheduling jobs in the production environment with flexible flowshop technology. Khaloli et al., (2010) used the ACO optimization algorithm to solve the problem of flexible flowshop with acceleration of weighted delay. Lea Oe et al., (2012), have developed a PSO optimization algorithm regarding the maximum completion time in the flexible flowshop problem. Marichelmam et al. (2013) investigated a BA optimization algorithm for the flexible flowshop problem with the aim of minimizing the maximum completion time. Wang (2013) proposed an EDA optimization algorithm for the flexible flowshop problem with the aim of minimizing the maximum completion time. An SFLA optimization algorithm was proposed by Zhou et al. (2013) to minimize the maximum completion time in the scheduling problem in a flexible flowshop generation environment. Chong and Liu (2013) proposed an IAIS optimization algorithm for the flexible flowshop problem to minimize the maximum completion time. Pan and Dong (2014) propose an MBO optimization algorithm for the hybrid flowshop problem with the aim of minimizing time. Lee et al. (2014) proposed the HVNS optimization algorithm for the flexible flowshop problem with the aim of minimizing the maximum completion time. Marie Chelwam et al. (2014) used the CS optimization Collaborators (2014) provided flexibility to solve the flowshop problem with the aim of minimizing the maximum completion time. Zang et al. (2017) proposed an EMBO optimization algorithm to solve the flexible flowshop problem with the aim of minimizing the total time. Pan et al. (2017) proposed an IFFO optimization algorithm to minimize the maximum completion time for a flexible flowshop problem with a workflow-dependent startup time. Yu et al. (2018) have proposed the GA algorithm to minimize job latency in the flexible flowshop problem. Khareh and Agrawal (2019) expressed the HSSA algorithm to minimize the minimum latency and speed of the job in the case of a flexible flowshop with sequence-

dependent start-up time. Engine B. and Engine (2020) designed the MGLS algorithm for the flexible multi-stage flowshop problem to minimize the maximum completion time. As evident in the related literature, different metaheuristics have been used to solve the flexible flowshop problem to minimize the maximum completion time. In this regard we can cite the discrete artificial bee optimization algorithm (Pan et al., 2014), the extended migratory bird optimization algorithm (Zang et al., 2017), the genetic algorithm (Yu et al., 2018), the hybrid squirrel search algorithm (Khare and Agraval, 2019), and local and global search algorithm (Engine B. and Engine O, 2020). These efforts have shown to have functions and operators that are easily adapted to our proposed problem. Therefore, to solve large-size problems, we adapt those five meta-heuristic optimization algorithms to the proposed problem and present a hybrid whale optimization algorithm.

Regarding research related to the scheduling of jobs with limited machine and manpower resources, Mehravaran and Logendran (2013) have presented a linear mixed integer programming model as well as three metaheuristic algorithms to meet the dual criteria of non-permutation flowshop technology with limited resources. Solve man-machine and sequence-dependent start-up times to minimize the sum of weight delays and the sum of weight completion times. Shahvari and Logandaran (2017) address a two-objective batch process problem with limited human resources and unrelated parallel machines to minimize maximum completion time plus the total cost of accelerating and delaying jobs in parallel with the total cost of the batch process. They have studied. Pinda et al. (2019) have developed a repetitively greedy algorithm to solve a scheduling problem with flexible work-flowshop technology with limited machine and human resources with a goal that simultaneously considers time and total time criteria. Yazdani et al. (2014) have considered the issue of scheduling with flexible work-flowshop technology with limited resources of human and machine resources. They propose a refrigeration simulation algorithm and a hybrid algorithm developed from variable neighborhood search algorithms to minimize maximum completion time. Li and Guo (2014), Zhang and Wang (2016), Gao and Pen (2016), and Zhang et al. (2017) The problem of scheduling with flexible work-flowshop technology with dual human-machine resources to maximize Study completion time. Zhang et al. (2019) two mathematical models of hybrid integer linear programming based on different modeling ideas and an effective variable local search algorithm to solve a scheduling problem with flexible work-technology with limited human resources and provided energy to reduce total energy consumption. Yazdani et al. (2019) presented the genetic algorithm of faulty ranking and the genetic algorithm of faulty sorting and a scheduling problem with flexible work-flowshop technology with limited dual resources, multiple objectives, and goals of minimizing maximum completion time, workload the total number of machines, and the working volume of the critical machine were presented simultaneously. Gong et al. (2018) developed a memetic optimization algorithm to solve the scheduling problem with flexible work-technology with limited dual human-machine resources and multi-objective machines to minimize maximum completion time, maximum total workload of all machines, and suggest the workload of the machines. Wu et al. (2020) have studied the two-objective problem of flexible work-schedule scheduling with limited dual human-machine resources with loading and unloading time with the aim of minimizing the maximum completion time and minimizing start-up time. They have solved a problem by designing a new multi-objective genetic algorithm. Figileska (2014), Walder and Nast (2017), and Figileska (2018) examined the problem of job scheduling in a flowshop production environment with limited dual human-machine resources to minimize maximum completion time. Gong, Chiong, Deng, Han, Zhang, Lin, and Li (2019) focus on the issue of flexible hybrid flowshop scheduling with limited dual human-machine resources. The existing research focuses on the resource flexibility of machines and workers along with process time, factors related to energy consumption as well as the cost of wage labor at the same time. Torres et al. (2021) have

studied the problem of two-station combined flowshop with limited human resources with the aim of minimizing the average delay. They have used an innovative algorithm to solve the problem. Han et al. (2021) addressed a bi-objective problem of combined flowshop with human resource constraints to minimize the maximum completion time and minimize the sum of tardiness simultaneously. They proposed an innovative two-objective evolutionary algorithm to solve the problem on a large scale.

As evident in the related literature, researchers have considered only one unique executive skill for job processing, and no effort has addressed the issue of scheduling in the flexible flowshop considering manpower skills constraints and its effect on the processing times. In this way, considering multi-skilled manpower and the possibility of movement between production stages is discussed in this study as a novel innovation.

The different parts of the article are as follows. In the second part, the research problem is described in detail and the linear programming formulation of a compound integer is stated. In Section 3, the whale hybrid optimization algorithm will be studied to solve the proposed problem. Section 4 provides the computational result and analysis. Finally, Section 5 is devoted to conclusions and future suggestions.

Problem Description

A flexible flowshop scheduling problem considering manpower skill-based processing times is tackled in this work. All jobs are processed on a flexible flowshop consists of a set of M production stations and each job j (j={1,…, n}) must follow the whole sequence process of production stations, (so that first production station 1, then production station 2 and up to production station m). There are several identical parallel machines at each i∈M of the production station, and each machine can perform processing operations by the number (p> 1) of workers. Labors have different executive skills as well as performance. Therefore, the processing time of each job is not only affected by the capacity of the production station but also by the efficiency of labor. Labors can move or dispatche between different production stations according to their skills.

So that the source of parallel machines at a production station is the same, a job can be processed by any source of machine l∈m_i at the production station i. A machine resource can process only one job at a time, a job can be processed by only one machine resource, each manpower can execute only one machine resource, and one machine resource can Processed only by one manpower. All sources of machinery, manpower, and jobs are available in zero time, and cutting jobs is not allowed. The intermediate buffers between the production stations have unlimited capacity. Travel time between the production station and start-up time is part of the processing time, and man-walking times between the production station and between the sources of parallel machines are not considered.

In the proposed problem, the machine resource must be allocated to process the jobs and determine which labor must be allocated to process the work. In other words, the sequence of jobs in each machine source as well as the sequence of jobs in each labor must be decided. So that the maximum time to complete the job is minimized. The problem data is assumed definite and predetermined.

Indices and sets

$\boldsymbol{i}$	Index of stages {1,…, m}
$\boldsymbol{j, k}$	Indices of jobs {1,…, n }
$\boldsymbol{m_i}$	The source set of machines in the production stage i, $i \in M$
$\boldsymbol{P}$	The manpower set {1,…, p }

$\boldsymbol{o_i}$	The set of manpower skilled in the job process at the ith stage, $i \in M$
$\boldsymbol{v_w}$	The set of production stages under the manpower w, $w \in P$

Parameters

$\boldsymbol{p_{jiw}}$	The processing time of job $j \in N$ in stage $i \in M$ by the manpower $w \in P$
Q	A very large number

Decision variables

$\boldsymbol{x_{jik}}$	A binary variable such that if job j is processed in stage i after job k is equal to 1, otherwise 0.
$\boldsymbol{y_{jil}}$	A binary variable such that if job j is processed in stage i on the machine source l is equal to 1, otherwise 0.
$\boldsymbol{c_{ji}}$	Continuous variable of completion time j in stage i
$\boldsymbol{h_{jiw}}$	A binary variable such that if job j is done in stage i by manpower w is equal to 1, otherwise 0.
$\boldsymbol{e_{jwk}}$	A binary variable such that if job j is done by manpower w after job k is equal to 1, otherwise 0.

Based on the abovementioned notations, the considered problem is formulated as a mixed-integer linear programming (MIP) model as below:

$$\text{Minimize} = C_{max} \quad (1)$$

Subject to

$$\sum_{l \in m_i} y_{jil} = 1 \qquad \forall\, i \in M, j \in N \quad (2)$$

$$\sum_{w \in o_i} h_{jiw} = 1 \qquad \forall\, i \in M, j \in N \quad (3)$$

$$c_{ji} - c_{j(i-1)} - p_{jiw}.h_{jiw} \geq 0 \qquad \forall\, i \in M, j \in N, w \in o_i \quad (4)$$

$$c_{ji} - c_{ki} - p_{jiw}.h_{jiw} + \mathrm{Q}(3 - x_{jik} - y_{jil} - y_{kil}) \geq 0 \qquad \forall\, j,k \in N: j < k, i \in M, l \in m_i, w \in o_i \quad (5)$$

$$c_{ki} - c_{ji} - p_{kiw}.h_{kiw} + (\mathrm{Q}.x_{jik}) + \mathrm{Q}(2 - y_{jil} - y_{kil}) \geq 0 \qquad \forall\, j,k \in N: j < k, i \in M, l \in m_i, w \in o_i \quad (6)$$

$$c_{ji} - c_{ki} - p_{jiw}.h_{jiw} + \mathrm{Q}(3 - e_{jwk} - h_{jiw} - h_{kiw}) \geq 0 \qquad \forall\, j,k \in N: j < k, i \in M, w \in o_i \quad (7)$$

$$c_{ki} - c_{ji} - p_{kiw}.h_{kiw} + (\mathrm{Q}.e_{jwk}) + \mathrm{Q}(2 - h_{jiw} - h_{kiw}) \geq 0 \qquad \forall\, j,k \in N: j < k, i \in M, w \in o_i \quad (8)$$

$$c_{ji} - c_{ks} - p_{jiw}.h_{jiw} + Q(2 - h_{jiw} - h_{ksw}) \geq 0 \qquad \forall\, j,k \in N, i,s \in M: i > s, w \in v_w \quad (9)$$

$$x_{jik}, y_{jil}, h_{jiw}, e_{jwk} \in \{0,1\} \qquad \forall\, j,k \in N: j < k, i \in M, w \in P \quad (10)$$

$$c_{ji} \geq 0 \qquad \forall\, i \in M, j \in \quad (11)$$

Relation (1) minimizes the maximum completion time of jobs as the considered objective function. Constraints (2) and (3) ensure that each job passes through the entire production stage and is processed exactly by a qualified machine source and a qualified manpower at each production stage. Be. The completion time of each job is specified in the constraints (4) to (9). On the other hand, relation (4) ensures that the operation of the i-m stage of a job starts processing after the operation of its previous stage. Constraints (5) and (6) indicate that it is not

possible to process two jobs at the same time with the same machine source, and one job cannot be processed before the end of each previous job in the source. The same machine is to be processed. Q is a large number. Constraints (7) and (8) state that it is not possible to process two jobs with the same manpower in the same operation at the same time, and one job cannot be completed before the end time of each job. The former is processed in the same operation with the same manpower. The constraint set number (9) determines that it is not possible to process two jobs in different operations with the same manpower at the same time, and one job cannot be performed in different operations with the same force before the time of completion of each previous job. The same human being is to be processed. Finally, the constraints (10) and (11) determine the model variables.

Problem Solving Approaches

In this section, a whale hybrid optimization algorithm is presented to solve the problem at hand. This section is divided into four sub-sections. In the first part, we present the background of the whale optimization algorithm. The second section details the discrete whale optimization algorithm, local search method, and diversity control method. In the third part, we will explain the process of the whale hybrid optimization algorithm, and in the fourth part, we will discuss how to encrypt and decrypt the answers.

Background of Whale Optimization Algorithm

The whale optimization algorithm is in the category of meta-heuristic algorithms expressed by Mirjalili in year 3 (Mirjalili and Lewis, 1). Just as whalers can position prey well and surround it, this algorithm is inspired by the mechanism and natural nature of whaling. In other words, the whale algorithm simulates the spatial position of bypassing and spiraling and the mechanisms of random hunting of the whale. The parts of the algorithm consist of three stages: hunting siege, bubble attack, and hunting search.

Recently, scientists in various fields have paid much attention to the features of the whale algorithm and have successfully used it to solve various optimization problems. Issues such as photonic crystal filters (Mirjalili et al., 2020) and other flowshop scheduling issues (Jiang et al., 2018; Abdul Basit, 2018; Liu, 2020).

Hunting siege:

Since the optimal position in the search space is not known at first, the whale algorithm assumes on this basis that the best answer of the current candidate is the target prey or close to the optimal state. Once the best search factor has been identified, other search agents try to update their positions relative to the best search agent, so that the behavior is shown by the equations (12).

$$X_{ij}^{t+1} \coloneqq X_{best,j}^{t}\text{-}A_{ij}.C_{ij}.X_{best,j}^{t} - X_{ij}^{t} \tag{12}$$

So t recent repetition, $A_{ij}$and C_{ij}coefficients i am the whale in the j dimension, $X_{best,j}^{t}$ the best answer ever obtained in the j dimension, $X_{ij}^{t}$the position of the i whale in the j dimension. It should be noted that if there is a better answer, $X_{best,j}^{t}$should be updated in each iteration. The coefficients $A_{ij}$and $C_{ij}$are calculated as (13) to (14).

$$A_{ij} \coloneqq 2a_{ij}.r_{ij} - a_{ij} \tag{13}$$

$$C_{ij} \coloneqq 2.r_{ij} \tag{14}$$

So that a_{ij}decreases linearly from 2 to 0 during each iteration (in both exploration and exploitation stages) and r_{ij} is a random value between [0,1].

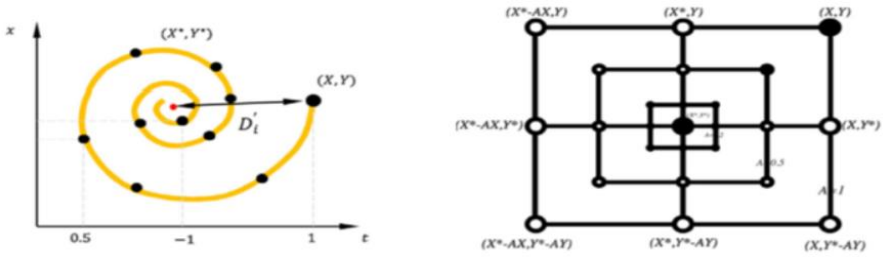
Bubble attack:

The bubble attack phase is the operation phase of the optimization algorithm. For mathematical modeling of whale bubble attack behavior, two approaches are expressed as follows:

Shrinkage blocking mechanism:

This behavior is formed by reducing the value of a_{ij} in relation to number (13). It should be noted that the oscillation ranges of A_{ij} is also reduced by a_{ij}Alternatively, $A_{ij}$is a random value in the interval [a_{ij}, a_{ij}] in which a_{ij}decreases from 2 to 0 during repetitions. By randomly assigning $A_{ij}$to [-1,1], it can create a new position of a search agent anywhere between the main agent position and the current best agent position. Figure 1 shows the possible positions from (x, y) to (x *, y *) that can be obtained by 0≤A≤1 in two-dimensional space.

Update the snail position:



B) Search in three-dimensional space A) Search in two-dimensional space

Figure 1. Searching the answer space

As shown in Figure 1, we first calculate the distance between the whale in the coordinates (X, Y) and the prey in (X *, Y *). A spiral equation is then generated between the position of the existing whale and the prey to generate the whale-shaped cochlear motion as shown in the equation (15)

$$X_{ij}^{t+1} \coloneqq \left|X_{best,j}^{t} - X_{ij}^{t}\right|.e^{bl}.\cos(2\pi l) + X_{best,j}^{t} \tag{15}$$

So that $\left|X_{best,j}^{t} - X_{ij}^{t}\right|$ The distance i from the whale to the prey (best answer ever obtained), b denotes a constant value to form a logarithmic helix, l a random number in the interval [0,1].

It is necessary to explain that whales swim along a contraction circle and in a spiral path around the prey. In order to simultaneously model the above behavior, it is assumed that the whales with a probability of 1% choose one of the mechanisms of the helical model or the contraction siege to update their position during the optimization. The mathematical model is as (16), where P is a random number:

$$X_{ij}^{t+1} = \begin{cases} X_{best,j}^{t} - A_{ij}.D_{ij}, & p < 0.5 \\ X_{best,j}^{t} - X_{ij}^{t}.e^{bl}.\cos(2\pi l) + X_{best,j}^{t}, & p \geq 0.5 \end{cases} \tag{16}$$

Hunting search:

Apart from the bubble attack strategy, the whale randomly searches for prey so that it can be used to search for prey (exploration stage). Whale based on A_{ij}The same strategy is based on changing the position vector of each other, the search is done randomly. Therefore, we use $A_{ij}$with random values greater than 1 or less than 1- to urge the search agent to move away from the reference whale. In order to update the position of the search agent in the exploratory stage, in comparison with the exploitation stage, instead of finding the best search agent so far, we use a randomly selected search agent. This method emphasizes exploration and allows the algorithm to perform a global search.

$$D_{ij} \coloneqq \left|C_{ij}.X_{rand_{ij}} - X_{ij}\right| \quad (17)$$

$$X_{ij}^{t+1} \coloneqq X_{rand_{ij}} - A_{ij}.D_{ij} \quad (18)$$

$X_{rand_{ij}}$ is a random whale in the jth dimension selected from the current population. Figure 2 shows the pseudo-code of the whale optimization algorithm.

Figure 2. Pseudo-code whale optimization algorithm

Input: Parameters (PS population size, maximum number of Max-iterations, number of variables D)
Output: Best Whale (X_{best})
// Population quantification //
For i from 1 to PS
For j from 1 to D.
$X_{ij} \coloneqq x_{min_j} + \left(x_{max_j} - x_{min_j}\right) * rand( )$
End
For each whale, $f_i \coloneqq fitnessfunction(X_i)$ // Calculate the target function for each whale
$i^* = \arg\left(min_{i=1,..,PS}(f_i)\right)$
$X_{best} \coloneqq X_{i^*}$// Get the best search agent
t:=0
As long as $t \leq Max_{iter}$
For i from 1 to PS
For j from 1 to D.
Update a_{ij}, A_{ij}, C_{ij}, l, p
If prob <0.5
If A≥l
// Exploration method
Set rn to a random integer between 0 and PS
$X_{rand,j} \coloneqq X_{rn,j}$// Random whale selection
Update position X_{ij}^{t+1} (current whale) by Equation (18)
A <l otherwise if
// Extraction method
Update position X_{ij}^{t+1} (current whale) by Equation No. (12)
End
Otherwise if prob≥0.5
Update position X_{ij}^{t+1} (current whale) by Equation (15)
End
End

End
Update X_{best}
t +t + 1
End
Return $X_{best}$and fitnessfunction (X_{best})
End

Whale hybrid optimization algorithm

In order to improve the quality of the whale optimization algorithm, we design a combined whale optimization algorithm to solve the proposed problem by combining the discrete whale optimization algorithm with the local search method and the diversity control method.

Application of the whale hybrid optimization algorithm to the proposed problem

Primary population

The population size of the whale hybrid optimization algorithm is fixed in each iteration, which is specified as PS. PS The answer is generated randomly between (0,1) for the initial population. To make the PS answer, a random number between 0 and 1 is generated for each job. To convert continuous solutions to discrete permutation solutions, the law of least value of position is used. For example, encoding and displaying the answer to a problem with 5 jobs is shown in Table 1. Suppose a vector answer obtained from continuous numbers in an iterative of the discrete whale optimization algorithm is denoted by p and p` is a discrete permutation answer.

Table 1. A sample of encoding solution

	Job				
	1	2	3	4	5
p	0.22	0.15	0.78	0.36	0.91
$\grave{p}$	2	1	4	3	5

The smallest value in row p is 0.15, which is related to job 2, so according to the law of the smallest position value, job 2 is in the first position of row p. The second smallest value in row p is 0.22, which is related to job 1, so job 1 is in the second position of row p. Similarly, all jobs are rotated in row p. Therefore, the sequence of jobs is p`= (2,1,4,3,5). It is clear that by presenting such an answer, there is no doubt that a reasonable timeline for the problem of scheduling jobs in the production environment with flexible flowshop technology, taking into account the innovation in human resource skills.

Discrete Whale Optimization Algorithm is a biologically inspired meta-heuristic algorithm designed to solve continuous problems. In order to use the whale optimization algorithm to solve scheduling problems, the structure of the algorithm must be adapted to the discrete environment, so we use the law of least value position to convert search agents to permutations. It should be noted that there is no evidence that a discrete whale optimization pattern has been explored to solve flexible flowshop scheduling problems. The pseudo-code of the discrete whale optimization algorithm is shown in Figure 3. It should be noted that the position of each search factor is updated by Equations (18), (12), and (15) and the maximum number of neighborhoods searched for each answer in each iteration of the proposed algorithm in the computational time. Specified by the max-iteration symbol.

Figure 3. Pseudo-code of discrete whale optimization algorithm

Input: Parameters (PS population size, maximum number of Max-iterations, number of variables D)
Output: Best Whale X_{best}
// Population quantification //
For i from 1 to PS
For j from 1 to D.
$X_{ij} \coloneqq x_{min_j} + \left(x_{max_j} - x_{min_j}\right) * rand( )$
End
Use the least value position (SPV) rule to convert search agents to discrete permutations
For each whale, $f_i \coloneqq fitnessfunction(X_i)$ // Calculate the target function for each whale
$X_{best} \coloneqq X_{i^*}$
$X_{best} \coloneqq X_{i^*}$// Get the best search agent
t:=0
As long as $t \leq Max_{iter}$
For i from 1 to PS
For j from 1 to D.
Update $a_{ij}, A_{ij}, C_{ij}, l, p$
If prob <0.5
If A≥l
// Exploration method
Set rn to a random integer between 0 and PS
$X_{rand,j} \coloneqq X_{rn,j}$// Random whale selection
Update positionX_{ij}^{t+1} (current whale) by Equation (18)
A <l otherwise if
// Extraction method
Update position X_{ij}^{t+1} (current whale) by Equation No. (12)
End
Otherwise if prob≥0.5
Update position X_{ij}^{t+1} (current whale) by Equation (15)
End
End
End
Use the least value position (SPV) rule to convert search agents to discrete permutations
Update X_{best}
t +t + 1
End
Return $X_{best}$and $fitnessfunction(X_{best})$
End

We use a combination of the focus method and diversity method to construct the proposed local search method. In the variation method, the combination of inverse and exchange operators is used so that α is a control parameter. In the focus method, our strategy is based on the fact that the job with the highest completion time is selected as the most problematic job and is greedily inserted by the operator. In this method, the motion operators are a combination of inverse operators and exchanges in the variation method and the operator is a greedy insert in the concentration method. Define π as an answer. Figure 4 shows the local search method.

Figure 4. Local search method

Consider π an answer
Generate two integers randomly between 1 and n, then place one in w and the other in a, so that w> a
/ Diversity Strategy /
If α is greater than a random number between 0 and 1, then using w and a, set π equal to the application of the exchange operator on π
Otherwise, set π equal to the application of the inverse operator in the range a to w on π
End
Calculate the objective functions π and π.
If the objective function $\tilde{\pi}$ <the objective function π
Then set π equal to π.
End
/ Focus strategy /
Set k to zero
As long as | π | +1> K
Select the job with the most completion time as the most problematic job, exit it from π, include it in the best position with the least objective function, and put the result in $\tilde{\pi}$
If the objective function $\tilde{\pi}$ <the objective function π
Then put π π in π.
End
k = k + 1
End
Return p
End

With the evolution of the whale optimization algorithm, most answers tend to converge, so this is inconsistent with population variability. This causes the search to stop at the local optimization. To solve this problem, we propose a variation control method. If the best answer obtained, which has not been improved so far, is larger than the γmax counter, the proposed method is performed.

In each run, the best answer is retained in the current population, and some other bad answers will be reconstructed by a random method. The diversity control method is shown in Figure 5.

Figure 5. Diversity control method process

Arrange the answers in the population incrementally based on the value of the objective function
For each $\pi \epsilon P_s$
If πϵ [1, ..,./ 4. P_s]
Then hold π
otherwise
if πϵ [./ 4. P_s, .., P_s]
Randomly generate a new answer and put it in π
End
End
End

After detailing the proposed algorithm in the previous sections, in this section the complete process of the proposed algorithm is set out in Figure 6.

Figure 6. Pseudo-code of the whale hybrid optimization algorithm

Input: Parameters (PS population size, maximum number of Max-iterations, number of variables D)
Output: Best Whale (X_{best})
/ Setting parameters /
Quantify the PS parameters, Max-iteration D
Set to zero ϑ
// Population quantification //
$P_s := \{P_s(1), P_s(2), \dots, P_s(PS)\}$ The answer is generated randomly, while another answer is generated by the earliest delivery time.
As long as t<t_{max}
// Discrete Whale Optimization Algorithm Process //
Perform the whale optimization algorithm for P_s
// Local Search Method Process //
For j from 1 to PS
Put $P_s(j)$ in π
Run the local search method for π and set it to $P_s(j)$
End
Use the least value position (SPV) rule to convert search agents to discrete permutations
Update X_{best}
If X_{best}has not changed, ϑ = ϑ + 1
// Diversity control method process //
As long as y_{max} = ϑ
For j from 1 to PS
Put $P_s(j)$ in π
Execute the variation control method for π and set it to $P_s(j)$
Set ϑ to zero
End
End
End
Return $X_{best}$and $fitnessfunction(X_{best})$
End

Five available optimization algorithms, which have shown acceptable efficiency and effectiveness in the research literature on scheduling problems, are tuned and applied to the problem at hand. These algorithms include the discrete artificial bee optimization algorithm (Pan et al., 2014), the extended optimization algorithm for migratory birds (Zang et al., 2017), the genetic algorithm (Yu et al., 2018), the hybrid squirrel search algorithm (Khare and Agraval, 2019), and local and global search algorithm (Engine B. and Engine O, 2020). All of these procedures have functions and operators that are easily adapted to our proposed problem. In order to be able to solve the proposed problem by the mentioned optimization algorithms, we adapt the mentioned algorithms to the proposed problem with the following changes: (1) Substitute the objective function of minimizing the maximum time of completion of jobs. (2) Replace the method of encoding and decryption of the answers described in the following section.

As explained in the previous sections, the purpose of scheduling jobs in the production environment with flexible flowshop technology is to decide on the sequence of jobs on machines and manpower, taking into account innovation in human resource skills. Therefore, in the method of encoding and decryption of the answer to the proposed problem, both decisions must be considered. Therefore, in this research, we express a new answer to the proposed

problem. Initially, the shift-based answer is encoded and displayed based on π = {π (1), π (2),…, π (n) ﻉ, and in each iteration g, $Time_g$ shows the scheduling time in each iteration. Give. Each job j has a priority value obtained by the relation $[\![pw_j := \{j : \pi(j) = j\}$. The first stage where job j should be processed is indicated by fg_j. The set of jobs available at $Time_g$ is specified by the LJ list. The process completion time on the M_{il}machine is indicated by repeating gm with FM_{il}^g. The process completion time is indicated by the manpower $P_w$in the repetition of gm with $[\![FP_w^g$. The pseudocode of the encoding and decryption method is shown in Figure 7. The main steps of the encoding and decryption method of the designed answers are summarized as follows:

Schedule all available jobs to be processed. Select jobs to be processed according to their priorities. All sources of machinery and manpower are selected for the available job according to the minimum workload at the production stage so that at one time a job is processed by only one machine and one machine is processed by only one manpower. Job must be scheduled throughout the production phase.

Figure 7. Pseudo-code method of encoding and decryption of the answer

Input: Answer p
Output: Maximum completion time
Put $g := 1$ ،$Time_g := 0$ ؛$iter := 1$؛;
For each j∈π
$pw_j := \{j : \pi(j) = j\}$ ؛$LS_j := \{M | j \in \pi\}$ ؛$C_j^g := \{0 | j \in \pi\}$ ؛$FP_w^g := \{0 | w \in P\}$ ؛
$FM_{il}^g := \{0 | i \in P, l \in m_i\}$ ؛$LJ := \emptyset$ ؛$\hat{\pi} := \pi$ ؛

End
As long as iter = 1
For each job jϵπˆ,
$fg_j := \min\{LS_j\}$
$LJ := \{j | C_j^g \leq Time_g\}$ // When scheduling $Time_g$, specify all available jobs that need to be processed
End
$j^* = Arg(min_{j \in LJ}(pw_j))$; // Decide on the selection of jobs to be processed according to their priorities
$(l^*, w^*) = Arg(min_{l \in m_{fg_{j^*}}, w \in p_{fg_{j^*}}}(\max(FM_{(fg_{j^*},l)}^g, FP_w^g) + p_{(j^*,fg_{j^*},w)}))$; // Adjust all machines and workers for available job according to the minimum workload in the production stage
Assign work$j^*$to machine $l^*$and worker w^* with minimum load time.
$FM_{(fg_{j^*},l^*)}^{g+1} := \max(FM_{(fg_{j^*},l^*)}^g, FP_{w^*}^g, Time_g) + p_{(j^*,fg_{j^*},w^*)}$; // Time to complete the operation on M_{il}
$C_{j^*}^{g+1} := FM_{(fg_{j^*},l^*)}^{g+1}$; // Complete job j^* in repeat gm
$FP_{w^*}^{g+1} := C_{j^*}^{g+1}$); // Time of completion of the operation by the worker w^*
g:=g + 1;
If $LS_{j^*} = \emptyset$
$\hat{\pi} := \hat{\pi} - \{j^*\}$;
End
If $|\hat{\pi}| = \emptyset$

iter:=2;
$itLS_{j^*} \coloneqq LS_{j^*} - \{fg_{j^*}\} er \coloneqq 2;$
otherwise
If $min_{j \in \hat{\pi}}\left(fg_j\right) = 1$
$Time_g \coloneqq min_{l \in m_1}\left(AM_{1l}^{g+1}\right);$
otherwise
$Time_g \coloneqq min_{l \in m_1}\left(AM_{1l}^{g+1}\right);$
End
End
End
The target function is equal to $\boldsymbol{C_{max}}$
End

Computational Results

Data Design

Since there is no benchmark data for scheduling in the production environment with a flexible flowshop considering human resource skills, two sets of experimental production examples are designed. The first contains 5 random examples generated for a combination of job sets n = {4, 6, 8, 10} and the production stage set m = {2,5} in the category of small and medium size examples (40 examples in total). The second one contains 10 random examples generated for a combination of job sets n = {15,20,25} and production stage set m = {10,15} in the category of large-size examples (60 examples in total) and the number of machine sources in each production stage at random Intervals [1,3] such as research (Zohli et al., 2019) are used. The structure of the output data for the process time of each job on each production step is generated by manpower w (p_{jiw}) in the interval [1,100]. The number of manpower 2 2 m and the number of skills of each manpower are randomly generated between [1,3]. The mathematical model Designed by GAMS software is checked. The optimized algorithms tested are encoded by MATLAB software and the tests are performed on a Core i7 processor system with 16.0 GB of memory.

Computational Results

This section numerically examines the performance of the proposed whale hybrid optimization algorithm against the 5 known optimization algorithms in the research background that were previously described in the previous section and adapted to the proposed problem. To implement the optimization algorithms, we have used the encoding, decryption, and objective function methods described in the previous sections. First, the parameters of the proposed optimization algorithm are adjusted, and then we examine the overall performance of the tested metaheuristic algorithms against the optimal solution of the proposed mathematical model. Finally, the proposed optimization algorithm is compared with the 5 optimization algorithms adapted to the proposed problem. The stop condition is equal to the required computational time, which will be specified as 3 × n × m based on the dimensions of each problem.

We use the percentage of relative deviation as a performance evaluation criterion, which is calculated as follows.

$$RPD(F) = \frac{(F - F^*)}{F^*} \times 100$$

In the above relation, F is the answer related to the known optimization algorithm in a given example and F^* is the best answer among the optimization algorithms or in other words the optimal answer in that particular example.

Parameter Tuning

The performance of optimization algorithms is highly dependent on the adjustment of their parameters. Taguchi method is used to adjust the parameters of the whale hybrid optimization algorithm and also to obtain the best effective parameters. Taguchi method uses the signal-to-noise ratio (S / N) to measure:

$$S/N\ ratio = -10 \times \log\,(RPD)^2$$

The proposed algorithm consists of four main parameters, and each of these parameters has four levels, which are specified in Table 2.

Table 2. Parameter levels for the proposed algorithm

Parameter level				parameters
4	3	2	1	
100	75	50	25	PS
10	8	4	2	Max-iteration
0.95	0.80	0.50	0.20	α
20	15	10	5	γmax

The 16 combinations of parameters, or in other words the orthogonal array of 16 lines L_16, are used to determine the best parameter combination. For each combination of parameters, the algorithm is run 5 times independently and the average percentage deviation of 5 runs is applied to the response variable. The Taguchi strategy is applied to the large size problem. The highest S / N ratio for the respective levels is selected as the highest level. Figure 8 shows the S / N ratio of the different levels of the parameters. As it is clear, PS is the most valuable parameter of the proposed algorithm. The specified combination of parameter values for the proposed algorithm is shown in Table 3.

Table 3. The result of parameter tuning

Parameters	Final value
PS	25
Max-iteration	10
α	0.95
γmax	10

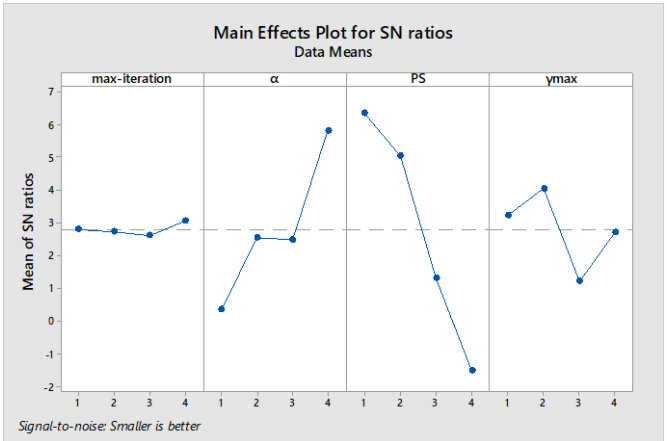


Figure 8. Signal-to-noise ratio analysis for parameter levels

Result of Solving the Small and Medium-Sized Instances

This section examines the optimized algorithms tested for the performance of small and medium-sized examples. To do this, we used 30 small and medium-sized examples that are optimally solved by the proposed mathematical model over a time of 1800 seconds, so that the proposed mathematical model in Examples 31 to 40 at the specified time has not been able to solve. Table 4 shows the results of the proposed mathematical model on small and medium-sized examples, the "time" column for the average time taken to solve the example optimally, and the "gap" column for the relative gap reported by the software. GAMS displays after the computational time of 1800 seconds.

Table 4: Computational results of mathematical model

Gap (percentage)	Time(seconds)	m	n	Example	Gap (percentage)	Time(seconds)	m	n	Example
0	3.7	5	4	6	0	4.3	2	4	1
0	8.1			7	0	8			2
0	10.3			8	0	8.1			3
0	6.8			9	0	4.1			4
0	7.7			10	0	9.3			5
0	90.5	5	6	16	0	22.3	2	6	11
0	31			17	0	38.6			12
0	57.6			18	0	65.6			13
0	20.5			19	0	17.2			14
0	74.6			20	0	34.4			15
0	387.2	5	8	26	0	622.3	2	8	21
0	1086			27	0	1033.8			22
0	1288			28	0	9.6			23
0	1641			29	0	986.7			24
0	1750			30	0	306			25
31.3	1800	5	10	36	13.2	1800	2	10	31
15.2	1800			37	9.5	1800			32
24.5	1800			38	25.6	1800			33
17.2	1800			39	18.4	1800			34
10.5	1800			40	29.2	1800			35

In this section, the results of adapted algorithms of Discrete Artificial Bee Optimization Algorithm (DABC), Extended Migratory Bird Optimization Algorithm (EMBO), Genetic Algorithm (GA), Hybrid Salp Swarm Algorithm (HSSA), Memetic Global and Local Search (MGLS), and the proposed Whale Combined Optimization Algorithm (HWOA) are compared with the optimal results obtained from the proposed mathematical model. Table 5 shows the average optimality gap of the algorithms implemented in each problem.

Table 5. Performance comparison of algorithms in solving small and medium instances

Average percentage of deviation						m	n
HWOA	EMBO	GA	MGLS	HSSA	DABC		
0.00	0.00	0.00	0.00	0.00	0.00	2	4
0.00	0.00	0.00	0.00	0.00	0.00	5	
0.00	0.01	0.06	0.02	0.03	0.02	2	6
0.00	0.17	0.04	0.01	0.01	1.02	5	
0.04	0.90	0.05	0.13	0.57	1.50	2	8
0.08	0.37	0.80	1.01	0.77	0.63	5	
0.02	0.24	0.16	0.19	0.23	0.53	Average	

The whale hybrid optimization algorithm optimally solves 25 examples (about 85%) out of 30 small and medium examples, leading to an optimization gap of 0.02%. Other algorithms solve between 18 and 23 examples optimally. Due to the optimization gap in these small and medium samples, all tested optimization algorithms are acceptable in the optimization gap of less than 1%.

Result of Solving the Large Size Instances

As shown in Table 5, there is no significant difference between optimization algorithms for solving small and medium-sized examples. In the following, large-size examples including 60 examples are solved by optimization algorithms to evaluate the performance. Each example is solved 10 times by each optimization algorithm. Table 6 shows the results of the best, average, and worst relative deviation of the optimization algorithms in examples of different sizes. In terms of the average relative deviation percentage, the Whale Hybrid Optimization Algorithm (HWOA) with a relative deviation percentage of 0.75% is the best. GA algorithm, MGLS algorithm, and HSSA algorithm are in the next ranks with 3.31, 3.52, and 4.02 percent, respectively. The worst performance is related to the DABC algorithm with a relative deviation of 6.26%. The HWOA ranks first in terms of the best and worst percentage deviations, and this reflects the strong performance of the HWOA. Figure 9 shows the convergence diagram (objective function) obtained from the HWOA algorithm for the problem size of 20 jobs and 15 machines. To evaluate the statistical performance of algorithms, the results of the "average" column are used. The results show a statistically significant difference between the performances of the optimized algorithms tested with zero p-value. For a more accurate comparison of the performance of the algorithms, Figure 10 shows the grouping of the algorithms and Figure 11 shows the percentage of relative deviation and the test is the least significant difference. Based on Figures 11 and 12, the HWOA algorithm performs statistically better than other algorithms. The MGLS algorithm and the GA algorithm have similar performance and their performance is better than the three algorithms DABC, EMBO, and HSSA. Figures 13 and 14 also show the average percentage of relative deviations of the algorithms for n and m, respectively. As is evident in these figures, the proposed whale hybrid optimization algorithm is reliable in all aspects of the problem in terms of optimality. The good performance of the whale hybrid optimization algorithm is achieved by the behaviors related to hunting siege, bubble attack, and hunting search, as well as the features related to the local search method and the diversity control method, a proper balance between general search response space and focus.

Table 6. Performance comparison of algorithms in solving large instances

Percentage of deviation										
Worst			Average			Best			m	n
MGLS	DABC	HSSA	MGLS	DABC	HSSA	MGLS	DABC	HSSA		
7.18	9.63	6.83	2.88	3.79	2.72	0.85	0.20	0.96	10	15
4.17	8.57	5.36	2.80	4.60	2.57	0.22	2.41	0.11		20
4.63	10.35	5.27	2.88	5.99	3.62	0.69	2.88	0.86		25
5.13	10.03	6.12	2.88	6.90	2.77	0.12	3.38	0.10	15	15
9.52	14.18	10.90	3.86	7.56	4.77	0.22	0.29	0.18		20
10.41	16.59	10.54	5.85	8.72	7.65	0.27	0.29	0.07		25
6.84	11.55	7.50	3.52	6.26	4.02	0.39	1.57	0.38	Average	
HWOA	EMBO	GA	HWOA	EMBO	GA	HWOA	EMBO	GA		
0.87	5.66	6.00	0.51	3.57	2.76	0.05	0.61	0.52	10	15
1.42	5.38	4.57	0.73	3.67	2.66	0.20	0.47	0.52		20
1.39	5.57	4.11	0.83	4.36	2.92	0.23	0.94	0.90		25
0.66	6.79	6.02	0.46	3.58	2.73	0.17	0.28	0.20	15	15
1.24	9.18	8.16	0.76	4.60	3.64	0.20	0.35	0.33		20
2.49	12.20	9.32	1.24	8.01	5.17	0.07	0.40	0.45		25
1.36	7.46	6.36	0.75	4.64	3.31	0.15	0.51	0.48	Average	

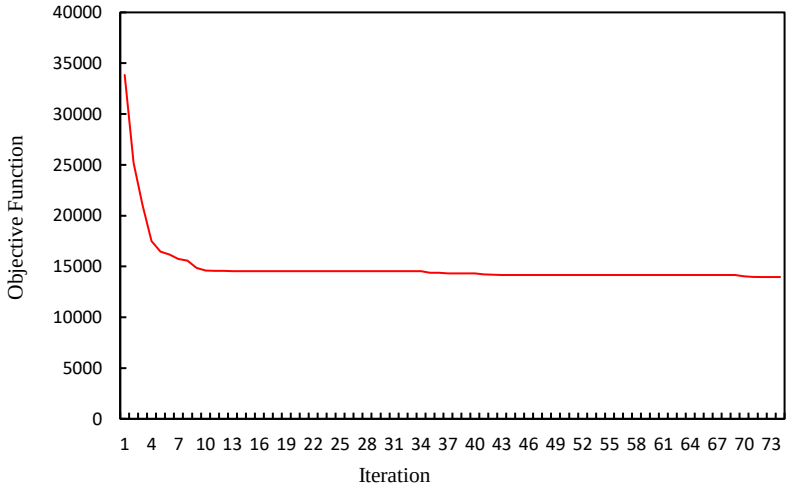


Figure 9. Convergence diagram for the problem size n=20 and m=15

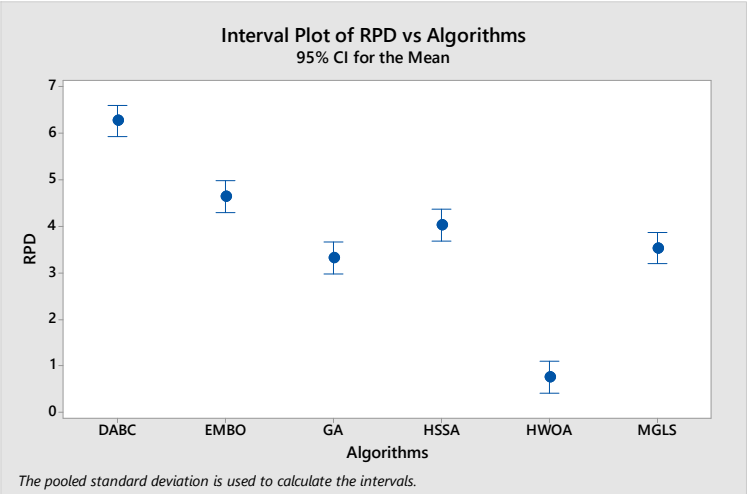


Figure 10. Grouping Algorithms

Algorithms	N	Mean	Grouping
DABC	60	6.260	A
EMBO	60	4.632	B
HSSA	60	4.017	C
MGLS	60	3.525	D
GA	60	3.313	D
HWOA	60	0.7550	E

Figure 11. Mean percentage of relative deviation and distance Percentage of relative deviation and test of least significant difference for optimization algorithms

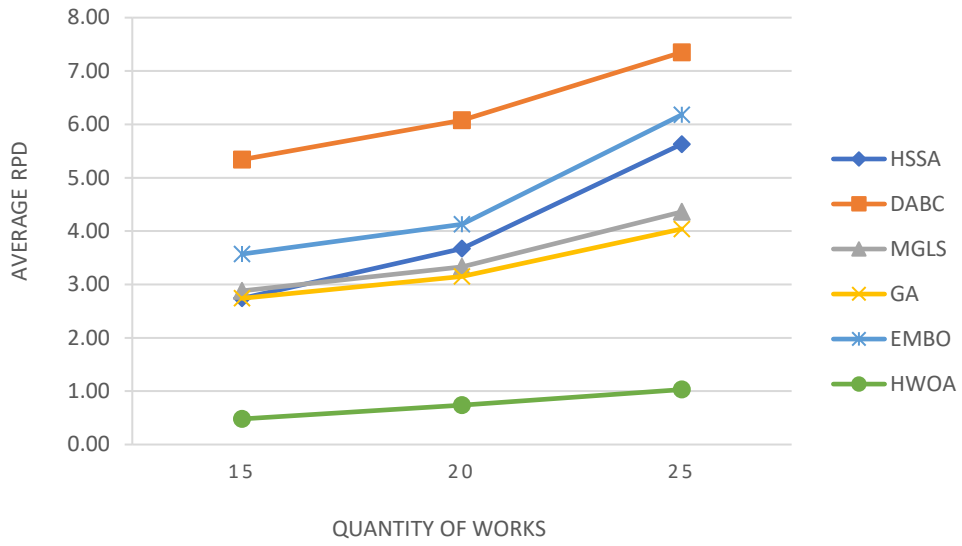


Figure 12. Average percentage of deviation of algorithms related to the number of jobs

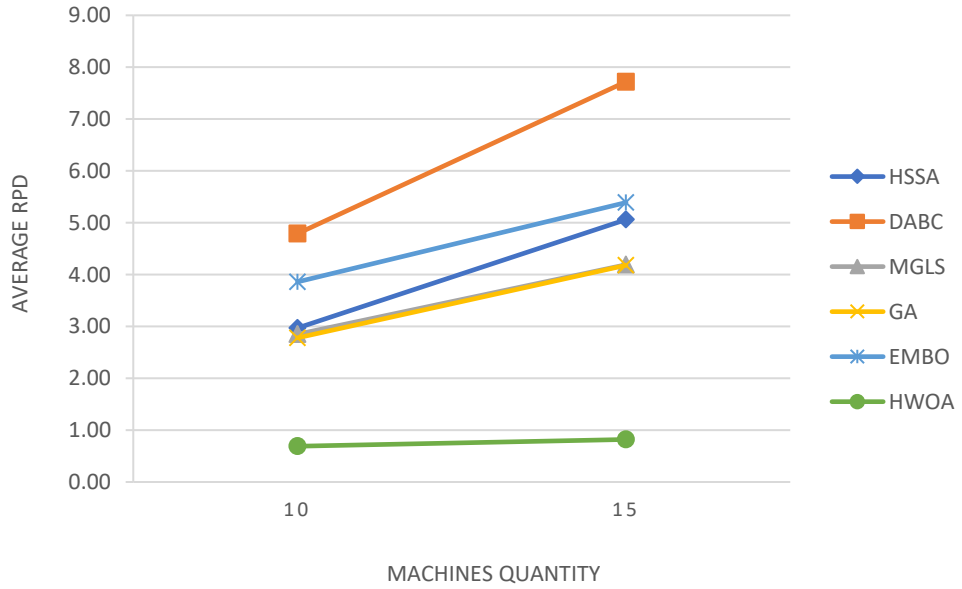


Figure 13. Mean percentage of deviation of algorithms related to the number of production steps

The most significant managerial implication of this study is related to manufacturing industries with flexible flowshop environments. The outcomes can be used by relevant managers and decision-makers to increase the total efficiency of their resources and reduce their total expected costs by appropriate planning to use multi-skill manpower. Using a mobile

workforce will lead to achieving more overall equipment efficiency (OEE) as a sustainable competitive advantage in the current competitive business environment and better planning for the future. The findings of this study can also assist managers of related industries in their decision-making processes to reduce job tardiness. Such industries could use the proposed model to optimally plan and manage their multiple resources and increase their customer satisfaction. At the same time, upstream and supervisory organizations that are responsible for economic, social, and environmental matters can use sensitivity analysis to introduce suitable and robust legislation on business objectively when dealing with multiple limited resources.

Conclusion

This study aimed to propose and investigate a flexible flowshop scheduling problem (FFSP) considering manpower skill-dependent process times. A mixed-integer linear programming (MIP) model was developed for the problem at hand to minimize the total completion time (Makespan) as the classic objective function in the scheduling field. Due to the Np-Hard nature of the considered problem, a hybrid whale optimization algorithm and a new response representation method from a cryptographic and decryption perspective were proposed to solve the problem on practical scales. In order to evaluate the performance of the proposed algorithm, two sets of examples were generated. Small and medium size examples were solved by the proposed mathematical model. The results show that the proposed mathematical model can find the optimal solution for most of the examples at the time of logical calculations. To evaluate the performance of the proposed algorithm in solving large-scale instances, 5 well-known approximation algorithms were matched with the proposed problem, and the performance of the proposed algorithm against the adapted algorithms was compared. In small examples, experiments showed that algorithms are usually very efficient. However, in large-size examples, the computational results showed that the proposed optimization algorithm performed much better than the adapted algorithms. From a managerial point of view, this paper presents a planning tool (solution method and mathematical model) to reduce the maximum completion time for factories with production environments with flexible flowshop with different human resource skills. For small factories with less than 10 jobs, mathematical modeling and optimization software are good planning tools. And large companies need more appropriate optimization techniques. Our proposed optimization algorithm has discovered an answer that reduces the maximum completion time compared to the adapted optimization algorithms by an average of 4% and in the worst-case scenario, has discovered an answer that is 7% better than adapted optimization algorithms. This indicates that the proposed optimization algorithm is strong even in the worst case.

Like every research, this study suffers some limitations. One of the most important limitations of this study is that the machine's breakdown is not considered in the proposed problem. Therefore, in order to improve the efficiency of the proposed models, new issues such as machine breakdown and maintenance operations could be added to the problem in the future. Moreover, the processing time is assumed to be a fixed value. However, we usually face uncertainty in the release time and processing time of jobs in real-world conditions.

There are several suggestions for future research related to this study. First, the current model is a deterministic model, which may not reflect reality accurately. Therefore, future studies can include uncertainty in the main parameters such as processing times. Moreover, adopting and applying other meta-heuristic algorithms for solving this problem and comparing results can be another interesting topic for future research. Realistic and functional assumptions such as energy savings, variable process speeds, and human resource constraints can also be considered. Also, the development of encoding and decryption methods can be considered.

References

Andrade-Pineda, J. L., Canca, D., Gonzalez-R, P. L., & Calle, M. Scheduling a dual-resource flexible job shop with makespan and due date-related criteria. Annals of Operations Research, 1-31.

Asgari, T. M., & Zandieh, M. (2014). A cloud-based simulated annealing algorithm for order acceptance problem with weighted tardiness penalties in permutation flow shop scheduling. Journal of industrial engineering and management studies (JIEMS), 1(1): 1-19.

Bartal, Y., Leonardi, S., Marchetti-Spaccamela, A., Sgall, J., & Stougie, L. (2000). Multiprocessor scheduling with rejection. SIAM Journal on Discrete Mathematics, 13(1), 64-78.

Cordone, R., & Hosteins, P. (2019). A bi-objective model for the single-machine scheduling problem with rejection cost and total tardiness minimization. Computers & Operations Research, 102, 130-140.

Dabiri, M., Darestani, S. A., & Naderi, B. (2019). Multi-machine flow shop scheduling problems with rejection using genetic algorithm. International Journal of Services and Operations Management, 32(2), 158-172.

Dhiman, G., & Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. Knowledge-Based Systems, 165, 169-196.

Dudek, R. A., Panwalkar, S. S., & Smith, M. L. (1992). The lessons of flowshop scheduling research. Operations Research, 40(1), 7-13.

Emami, S., Sabbagh, M., & Moslehi, G. (2016). A Lagrangian relaxation algorithm for order acceptance and scheduling problem: a globalised robust optimisation approach. International Journal of Computer Integrated Manufacturing, 29(5), 535-560.

Esmaeilbeigi, R., Charkhgard, P., & Charkhgard, H. (2016). Order acceptance and scheduling problems in two-machine flow shops: new mixed integer programming formulations. European Journal of Operational Research, 251(2), 419-431.

Figielska, E. (2018). Scheduling in a two-stage flowshop with parallel unrelated machines at each stage and shared resources. Computers & Industrial Engineering, 126, 435-450.

Framinan, J. M., Leisten, R., & García, R. R. (2014). Manufacturing scheduling systems. An integrated view on Models, Methods and Tools, 51-63.

Gao, L., & Pan, Q. K. (2016). A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem. Information Sciences, 372, 655-676.

Geramipour, S., Moslehi, G., & Reisi-Nafchi, M. (2017). Maximizing the profit in customer's order acceptance and scheduling problem with weighted tardiness penalty. Journal of the Operational Research Society, 68(1), 89-101.

Gong, G., Chiong, R., Deng, Q., Han, W., Zhang, L., Lin, W., & Li, K. (2019). Energy-efficient flexible flow shop scheduling with worker flexibility. Expert Systems with Applications, 112902.

Gong, X., Deng, Q., Gong, G., Liu, W., & Ren, Q. (2018). A memetic algorithm for multi-objective flexible job-shop problem with worker flexibility. International Journal of Production Research, 56(7), 2506-2522.

Gupta, J. N. (1988). Two-stage, hybrid flowshop scheduling problem. Journal of the operational Research Society, 39(4), 359-364.

Hasani, A., & Hosseini, S. M. H. (2020). A bi-objective flexible flow shop scheduling problem with machine-dependent processing stages: Trade-off between production costs and energy consumption. *Applied Mathematics and Computation*, *386*, 125533.

Hasani, A., & Hosseini, S. M. H. (2022). Auxiliary resource planning in a flexible flow shop scheduling problem considering stage skipping. *Computers & Operations Research*, *138*,

105625.
Jan, D., & W Patrick, N. (2009). Ergonomics contributions to company strategies. Applied Ergonomics, 40, 745-752.
Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. Naval research logistics quarterly, 1(1), 61-68.
Lei, D., & Guo, X. (2014). Variable neighbourhood search for dual-resource constrained flexible job shop scheduling. International Journal of Production Research, 52(9), 2519-2529.
Lei, D., & Guo, X. (2015). A parallel neighborhood search for order acceptance and scheduling in flow shop environment. International Journal of Production Economics, 165, 12-18.
Li, J., Huang, Y., & Niu, X. (2016). A branch population genetic algorithm for dual-resource constrained job shop scheduling problem. Computers & Industrial Engineering, 102, 113-131.
Lin, H. T., & Liao, C. J. (2003). A case study in a two-stage hybrid flow shop with setup time and dedicated machines. International Journal of Production Economics, 86(2), 133-143.
Lin, S. W., & Ying, K. C. (2015). Order acceptance and scheduling to maximize total net revenue in permutation flowshops with weighted tardiness. Applied Soft Computing, 30, 462-474.
Mehravaran, Y., & Logendran, R. (2013). Non-permutation flowshop scheduling with dual resources. Expert Systems with Applications, 40(13), 5061-5076.
Meng, L., Zhang, C., Zhang, B., & Ren, Y. (2019). Mathematical Modeling and Optimization of Energy-Conscious Flexible Job Shop Scheduling Problem With Worker Flexibility. IEEE Access.
Naderi, B., Gohari, S., & Yazdani, M. (2014). Hybrid flexible flowshop problems: Models and solution methods. Applied Mathematical Modelling, 38(24), 5767-5780.
Nakhaeinejad, M. (2019). Production Scheduling Optimization Algorithm for the Steel-Making Continuous Casting Processes. *Advances in Industrial Engineering*, *53*(4), 127-147.
Nguyen S, Zhang M, Johnston M (2014) Enhancing branch-and-bound algorithms for order acceptance and scheduling with genetic programming. In: Nicolau M, Krawiec K, Heywood MI, Castelli M, Garcia-Sanchez P, Merelo JJ, Rivas Santos VM, Sim K (eds) Genetic programming, 1st edn. Springer, Berlin, pp 124–136
Nguyen, S., Zhang, M., Johnston, M. (2014) A sequential genetic programming method to learn forward construction heuristics for order acceptance and scheduling. In: 2014 IEEE Congress on Evolutionary Computation (CEC). IEEE, pp. 1824–1831.
Pan, Q. K., Gao, L., Li, X. Y., & Gao, K. Z. (2017). Effective metaheuristics for scheduling a hybrid flowshop with sequence-dependent setup times. Applied Mathematics and Computation, 303, 89-112.
Pan, Q. K., Ruiz, R., & Alfaro-Fernández, P. (2017). Iterated search methods for earliness and tardiness minimization in hybrid flowshops with due windows. Computers & Operations Research, 80, 50-60.
Pan, Q. K., Wang, L., Li, J. Q., & Duan, J. H. (2014). A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation. Omega, 45, 42-56.
Pan, Q. K., Wang, L., Mao, K., Zhao, J. H., & Zhang, M. (2012). An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process. IEEE Transactions on Automation Science and Engineering, 10(2), 307-322.
Quadt, D., & Kuhn, H. (2005). Conceptual framework for lot-sizing and scheduling of flexible flow lines. International Journal of Production Research, 43(11), 2291-2308.
Ramezanian, R., & Hallaji, M. (2021). A Hybrid Approach for Home Health Care Routing and Scheduling Using an Agent-Based Model. *Advances in Industrial Engineering*, *55*(2), 165-

176.
Ruiz, R., & Maroto, C. (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. European Journal of Operational Research, 169(3), 781-800.
Shabtay, D., & Gasper, N. (2012). Two-machine flow-shop scheduling with rejection. Computers & Operations Research, 39(5), 1087-1096.
Shahvari, O., & Logendran, R. (2017). A bi-objective batch processing problem with dual-resources on unrelated-parallel machines. Applied Soft Computing, 61, 174-192.
Silva, Y. L. T., Subramanian, A., & Pessoa, A. A. (2018). Exact and heuristic algorithms for order acceptance and scheduling with sequence-dependent setup times. Computers & Operations Research, 90, 142-160.
Thevenin, S., & Zufferey, N. (2019). Learning Variable Neighborhood Search for a scheduling problem with time windows and rejections. Discrete Applied Mathematics, 261, 344-353.
Thevenin, S., Zufferey, N., & Widmer, M. (2015). Metaheuristics for a scheduling problem with rejection and tardiness penalties. Journal of Scheduling, 18(1), 89-105.
Thevenin, S., Zufferey, N., & Widmer, M. (2016). Order acceptance and scheduling with earliness and tardiness penalties. Journal of Heuristics, 22(6), 849-890.
Udo, G. G., & Ebiefung, A. A. (1999). Human factors affecting the success of advanced manufacturing systems. Computers & Industrial Engineering, 37, 297-300.
Waldherr, S., & Knust, S. (2017). Decomposition algorithms for synchronous flow shop problems with additional resources and setup times. European Journal of Operational Research, 259(3), 847-863.
Wang, J., Zhuang, X., & Wu, B. (2017). A new model and method for order selection problems in flow-shop production. *Optimization and Control for Systems in the Big-Data Era: Theory and Applications*, 245-251.
Wang, S., & Ye, B. (2019). Exact methods for order acceptance and scheduling on unrelated parallel machines. Computers & Operations Research, 104, 159-173.
Xiao, Y., Yuan, Y., Zhang, R. Q., & Konak, A. (2015). Non-permutation flow shop scheduling with order acceptance and weighted tardiness. Applied Mathematics and Computation, 270, 312-333.
Xie, X., & Wang, X. (2016). An enhanced ABC algorithm for single machine order acceptance and scheduling with class setups. Applied Soft Computing, 44, 255-266.
Xu, L., Wang, Q., & Huang, S. (2015). Dynamic order acceptance and scheduling problem with sequence-dependent setup time. International Journal of Production Research, 53(19), 5797-5808.
Yavari, M., Marvi, M. & Akbari, A.H. (2019). Semi-permutation-based genetic algorithm for order acceptance and scheduling in two-stage assembly problem. Neural Comput & Applic doi:10.1007/s00521-019-04027-w.
Yazdani, M., Zandieh, M., & Tavakkoli-Moghaddam, R. (2019). Evolutionary algorithms for multi-objective dual-resource constrained flexible job-shop scheduling problem. OPSEARCH, 1-24.
Yazdani, Mehdi; Zandieh, Mustafa; Tavakoli Moghaddam; Reza (2014). "A hybrid meta-heuristic algorithm for the problem of flexible flowshop job scheduling with limited dual human-machine resources." Journal of Industrial Management Studies, 12 (33), 43-74. (in Persian).
Yu, C., Semeraro, Q., & Matta, A. (2018). A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility. Computers & Operations Research, 100, 211-229.
Zhang, B., Pan, Q. K., Gao, L., Zhang, X. L., Sang, H. Y., & Li, J. Q. (2017). An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot

streaming. Applied Soft Computing, 52, 14-27.
Zhang, J., Wang, W., & Xu, X. (2017). A hybrid discrete particle swarm optimization for dual-resource constrained job shop scheduling with resource flexibility. Journal of Intelligent Manufacturing, 28(8), 1961-1972.
Zheng, X. L., & Wang, L. (2016). A knowledge-guided fruit fly optimization algorithm for dual resource constrained flexible job-shop scheduling problem. International Journal of Production Research, 54(18), 5554-5566.
Zohali, H., Naderi, B., & Mohammadi, M. (2019). The economic lot scheduling problem in limited-buffer flexible flow shops: Mathematical models and a discrete fruit fly algorithm. Applied Soft Computing, 80, 904-919.