BOHR International Journal of Operations Management Research and Practices
2024, Vol. 3, No. 1, pp. 1–8
DOI: 10.54646/bijomrp.2024.25
www.bohrpub.com

CASE STUDY

Application of genetic algorithm, GA, to solve a flow shop scheduling problem with changeover times in operations: a case study

Phong Nguyen Nhu*, Kim Ngan Nguyen Thi and Thanh Huyen Tran Vo Thi

Department of Industrial Systems Engineering, HCMC University of Technology, VNU HCM, Ho Chi Minh City, Vietnam

*Correspondence:
Phong Nguyen Nhu,
nnphong@hcmut.edu.vn

Received: 02 January 2024; Accepted: 19 January 2024; Published: 01 March 2024

Flow Shop Scheduling (FSS) Problems are examples of combinatorial optimization issues that are classified as NP-hard. Because of the NP-hard structure of FSS problems, it can be extremely challenging to find mathematical modeling methodologies that will result in an optimal solution for these problems. The Genetic Algorithm (GA), which is a metaheuristic approach, is one of the most important factors in the process of locating near-optimal answers to NP-hard optimization issues. In this research, a GA model for addressing an FSS problem was developed with the goal of lowering the overall weighted tardiness time and placing a constraint on the operation changeover time. When compared with the performance of the standard heuristics EDD, being used in the company under study, the GA model's performance was shown to be superior. Based on the findings, it can be shown that the objective value was cut by 43%, going from 215.95 (h) to 123.07 (h). This demonstrates that the GA model is an effective strategy for addressing FSS problems.

Keywords: genetic algorithm, metaheuristics, flow shop scheduling, tardiness time, changeover times

1. Introduction

The process of assigning resources to a set of activities so that they can be completed throughout a period of time is known as scheduling. The order or sequence in which a collection of jobs are to be processed by a number of machines in the most efficient manner can be determined via scheduling problems. In FSS problems, m distinct machines and n distinct jobs are considered; each job comprises m operations, and each operation demands a different machine; also, all of the tasks are processed in the same order; this is known as the processing order.

The company studied is currently having problems with late orders, leading to low customer service level. After analysis, the root cause was found to be due to a bad scheduling method. The company is currently using the EDD heuristic method. Order tardiness times were quite high. The company wanted to improve its scheduling methods with the objective of reducing order tardiness times, thereby improving on-time delivery rates, and enhancing customer service levels.

The problem that needs to be handled is an FSS problem, and it is assumed that the orders are ready before the scheduling process begins. The overall weighted tardiness of orders needs to be reduced as much as possible in order to accomplish the objectives of the problem. The constraints are on the sequence of orders, on the sequence of operations in the orders, and on machine changeover time.

In this paper, given the aforementioned assumptions, objectives, and constraints, a model of the problem is constructed, and a GA algorithm is developed to solve it. The GA algorithm will identify an appropriate solution based on the problem model, and then its efficacy will be determined by comparing that solution to the solution obtained by the currently used heuristic model.

2. Literature review and research methodology

2.1. Flow shop scheduling FSS problems

Flow shop scheduling problems consider different machines and different jobs. Each job consists of different operations and each operation requires a different machine and all the jobs are processed in the same processing order (1). Flow shop scheduling problems are NP-hard combinatorial optimization problems. For such problems, heuristics play a major role in searching for near-optimal solutions (2).

O Etiler, B Toklu, M Atak, and J Wilson developed a genetic algorithm-based heuristic for the flow shop scheduling problem with makespan as the criterion (3). Complex GA algorithms have also been researched to solve the FFS problem effectively. Orhan Engin, Gülsad Ceran, and Mustafa K. Yilmaz had developed an efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems (4).

Genetic algorithms are also combined with other algorithms to solve the FSS problem more effectively. Moch Saiful Umam, Mustafid Mustafid, and Suryono Suryono had combined the tabu search process with a genetic algorithm to solve the flow shop scheduling problem with the objective of minimizing makespan (5). Anna Burduk, Kamil Musiał, Joanna Kochańska, Dagmara Górnicka, and Anastasia Stetsenko had applied tabu search and genetic algorithm to solve production process scheduling problems and found that intelligent methods can find, in relatively short time, the solution that is close to the optimal and acceptable from the problem point of view (6).

This paper researches and applies a simple GA algorithm to solve the FSS problem to get better results than those of the current EDD dispatching method. This model is an initial basic model that can be developed into more complex GA models, or models that combine GA with other algorithms to be able to solve FSS problems more effectively.

2.2. Genetic algorithm

In 1975, Holland was the first to introduce the concept of a genetic algorithm (GA), a form of artificial intelligence search that mimics natural processes like evolution and natural selection by using a set of instructions encoded in each individual's chromosomes. It is an effective method for resolving optimization issues.

In GA, the solution space is typically represented as a population of chromosomes, with each chromosome standing in for a possible solution. In this concept, strings represent chromosomes. A specific string format can be used to code the chromosomes.

Each chromosome has an associated fitness value. The fitness function quantifies how close the solution comes to solving the problem. From the problem's goal function, we can infer the fitness function. The first generation is determined by the number of chromosomes that are selected. Selection, crossover, mutation, and replacement are only few of the genetic operators used on the current generation's chromosomes to produce the new generations.

The algorithm takes a starting population and generates offspring that are, in theory, healthier and more robust than their forebears. This procedure is performed until a criterion for stopping the process is met. Each new chromosome represents a different answer at each generation.

2.3. Research methodology

The FSS problem is an example of an NP-hard problem with a substantial amount of potential solution space. The methodology, used in this research to solve the problem, includes 2 phases:

- Phase A: Construct the model of the problem.
- Phase B: Use GA model to solve the problem.

In phase A, the model of the problem is formulated with the objective of minimizing the total weighted tardiness time and constraint on operation changeover time.

To solve the issue in phase B, a GA model is utilized, which is determined by the model of the problem. The process for the GA is as follows:

Step 1: Set the GA model's initial conditions.
Step 2: Create the first population $\mathbf{P}^{(0)}$. Set k = 0.
Step 3: Establish the elite population $\mathbf{P}_E^{(k)}$.
Step 4: Establish the genetic population $\mathbf{P}_G^{(k)}$.
Step 5: Develop the next population $\mathbf{P}^{(k+1)}$. Set k = k + 1.
Step 6: Make sure that the termination rule has been followed. In the event that the answer is "No," back to step 3. If the answer is "Yes," then the cycle should be completed.
Step 7: Run the algorithm a number of times to choose the best scheduling result.

Step 1 setups the structure and parameters of the GA model, including the method of coding, the GA factors, and the termination rule.

For coding, the orders are numbered, each gene is corresponding to an order, and each chromosome is a string of genes. The sequence of genes represents the sequence of order scheduled. For example, the chromosome format for a scheduling problem with 10 orders is as follows, the order number of order (where G_i is the order number of order i, i = 1 ÷ 10).

$$C = [G1, G2, G3, G4, G5, G6, G7, G8, G9, G10]$$

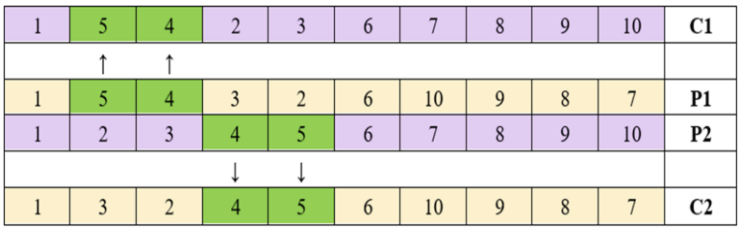

FIGURE 1 | Precedence operation crossover (POX).

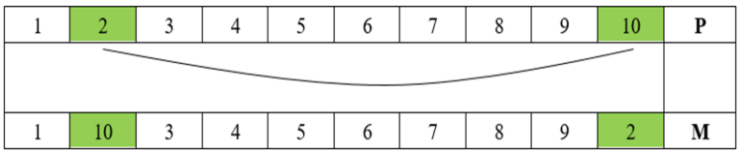

FIGURE 2 | SWAP mutation method.

The GA factors include fitness function, the population size, and the GA operators. The objective function of the problem is utilized to determine how the fitness function should be formulated. The objective value determines the degree to which one's fitness level can be improved. The selection operator, the crossover operator, the mutation operator, and the replacement operator are all different types of GA operators.

The rule of termination: After a predetermined number of repetitions in a row, the population's best objective value *Tbest* does not become any better.

Step 2 generates the first population $\mathbf{P}^{(0)}$ from solution space, with the population size defined by step 1. This step also starts the algorithm by setting the iteration counting index k to 0.

Step 3 uses the selection operator in order to produce an elite population $\mathbf{P}_E^{(k)}$ from current population $\mathbf{P}^{(k)}$. The fitness function is what the selection operator uses to determine which chromosomes from the current population should be included in the elite population. The higher the fitness value, the greater the probability that an individual will be chosen.

Step 4 the elite population $\mathbf{P}_E^{(k)}$ is used in conjunction with the crossover and mutation operators to produce the genetic population $\mathbf{P}_G^{(k)}$.

The $\mathbf{P}_E^{(0)}$ chromosomes that are included in the crossover list $\mathbf{P}_c$ are chosen by the crossover operator, which takes into account the crossover probability. Then, using a crossover technique, we choose certain pairs of $\mathbf{P}_c$ chromosomes to transpose into a new set of chromosomes that will become part of $\mathbf{P}_G^{(k)}$.

The crossover method used in this research is *POX (Precedence Operation Crossover)*. POX crosses over 2 parents P1 and P2 to make 2 children C1 and C2 as shown in **Figure 1**.

With a given mutation probability, the mutation operator chooses chromosomes from $\mathbf{P}_E^{(0)}$ to add to the mutation list $\mathbf{P}_m$. To create the genetic population $\mathbf{P}_G^{(k)}$, mutations

	M1	M2	M3	M4
P1	O1	O4	-	
P2	O2	O5	O7	O8
P3	O3	O6	-	

FIGURE 3 | The production process.

are then picked for each chromosome in $\mathbf{P}_m$ using a mutation technique.

The mutation method used in this research is *SWAP*. SWAP mutates parent P to make child M as shown in **Figure 2**.

Step 5 generates the next population $\mathbf{P}^{(k+1)}$ from the current population $\mathbf{P}^{(k)}$ and the elite population $\mathbf{P}_E^{(k)}$ by using replacement operator. If the fitness values of the chromosomes in $\mathbf{P}_G^{(k)}$ are higher above an acceptable threshold, then they will be joined to the chromosomes in the current population, $\mathbf{P}^{(k)}$, to produce the next population, $\mathbf{P}^{(k+1)}$.

The value of the n^{th} chromosome in the ranking of $\mathbf{P}^{(k)}$ is used to determine the threshold for the ranking. With population size P, threshold value K, n is defined as follows:

$$n = round\ \frac{P}{K}$$

In order to maintain a stable population size, the chromosomes with the lowest values are eliminated from the population after an influx of newcomers. This step also increases the iteration counting index k by 1 to prepare for the next iteration, if any.

Step 6 checks the termination rule. The algorithm usually terminates after a number of iterations if the objective function is not improved. If the termination rule is not met,

TABLE 1 | The weight W_i, i = 1÷10, and the due date D_i, i = 1÷10, of order i.

i	1	2	3	4	5	6	7	8	9	10
W_i	3.70	3.40	3.30	4.65	3.90	2.35	2.70	4.55	4.65	4.30
D_i (h)	24	36	40	60	68	80	88	88	96	96

TABLE 2 | The processing time P_{ij} of order i, i = 1÷10, on operation j.

J	P_{1j}	P_{2j}	P_{3j}	P_{4j}	P_{5j}	P_{6j}	P_{7j}	P_{8j}	P_{9j}	P_{10j}
1	2.34	6.17	6.20	7.09	2.47	9.56	3.44	14.74	5.26	1.39
2	2.25	0.94	7.05	3.22	1.07	5.26	1.81	7.49	5.26	0.66
3	0.00	3.99	4.34	4.38	2.60	9.52	0.00	16.75	14.74	0.44
4	2.63	1.33	1.08	9.00	2.60	12.56	1.32	17.54	16.52	1.29
5	1.06	1.00	6.58	9.00	4.21	12.64	0.65	17.54	16.52	1.29
6	0.00	8.33	6.58	3.60	3.95	12.64	0.00	13.33	15.79	0.40
7	2.67	3.29	1.13	2.21	0.68	4.00	1.11	5.26	1.60	0.44
8	4.08	3.31	3.42	12.00	3.95	6.12	2.21	8.22	5.26	1.32

the algorithm returns to step 3 for the next iteration. If the termination rule is met, the iteration loop is stopped. One run has finished.

Step 7 runs the algorithm a number of times to choose the best scheduling result among the runs.

The research methodology of the GA model is shown in the following sections.

3. The model of the flow shop scheduling problem

The problem that needs to be solved is an FSS problem (2) with 10 orders, O_i, where i is an integer from 1 to 10, and scheduling on 4 machines, M_1, M_2, M_3, and M_4. Each order is comprised of three distinct pieces, labeled P_1, P_2, and P_3, which are divided among four different machines in the manner shown in **Figure 3**.

The weight W_i, i = 1÷10, and the due date D_i, i = 1÷10, of order i are estimated in **Table 1**.

The processing times P_{ij} of order i, i = 1÷10, on operation j, j = 1÷8, are estimated in **Table 2**.

The changeover times in hours on operation j, j = 4÷8 are equal to 0, S_j = 0, j = 4÷8. The changeover times in hours on operation j, j = 1÷3 are the same and depend on the current order i = 1÷10, and the next order, i' = 1÷10, as shown in **Table 3**.

The model is built up using the start time (TS_{ij}), the completion time (TE_{ij}) of order i at operation j, and the tardiness time (Ti) of order i as independent variables.

The constraints on the sequence of operation on each order are as follows:

TABLE 3 | Changeover time (h) S_j, j = 1÷3.

	1	2	3	4	5	6	7	8	9	10
1	0.0	0.5	2.0	0.0	0.0	2.0	2.0	2.0	2.0	2.0
2	0.5	0.0	0.0	0.0	2.0	2.0	2.0	2.0	2.0	2.0
3	2.0	0.0	0.0	2.0	0.5	2.0	0.5	0.5	2.0	0.5
4	0.0	0.0	2.0	0.0	0.0	0.0	2.0	2.0	2.0	2.0
5	0.0	2.0	0.5	0.0	0.0	2.0	0.5	2.0	0.5	0.5
6	2.0	2.0	2.0	0.0	2.0	0.0	0.5	2.0	0.0	0.0
7	2.0	2.0	0.5	2.0	0.5	0.5	0.0	0.5	2.0	0.0
8	2.0	2.0	0.5	2.0	2.0	2.0	0.5	0.0	0.0	0.0
9	2.0	2.0	2.0	2.0	0.5	0.0	2.0	0.0	0.0	2.0
10	2.0	2.0	0.5	2.0	0.5	0.0	0.0	0.0	2.0	0.0

$$TS_{i4} \geq TE_{i1},$$

$$TS_{i5} \geq TE_{i2},$$

$$TS_{i6} \geq TE_{i3},$$

$$TS_{i7} \geq TE_{i5},$$

$$TS_{i8} \geq \max(TE_{i4}, TE_{i6}, TE_{i7}).$$

The start time of order i at operation j, TS_{ij}, depends on the end time of the previous order i', $TE_{i'j}$, and the changeover time between the orders on operation j.

$$TS_{ij} = TC_{i'j} + S_j.$$

The end time of order i on operation j, TE_{ij}, is determined by the start time and processing time of the order.

$$TC_{ij} = TS_{ij} + P_{ij}.$$

The tardiness time of order i, T_i, is determined by the end time in the last operation and due time of the order.

$$T_i = \max(0, TE_{i8} - D_i).$$

The objective function that minimizes the total weighted tardiness is defined as follows:

Tbest = Min T,

$$T = \sum W_i * T_i \ (i = 1 : 11)$$

The company is currently using the EDD dispatching method. The sequence of dispatching S and the value of the objective function T are as follows:

$$S = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);$$

$$T = 215.95 \ (h).$$

The Gantt Chart is as in **Figure 4**.

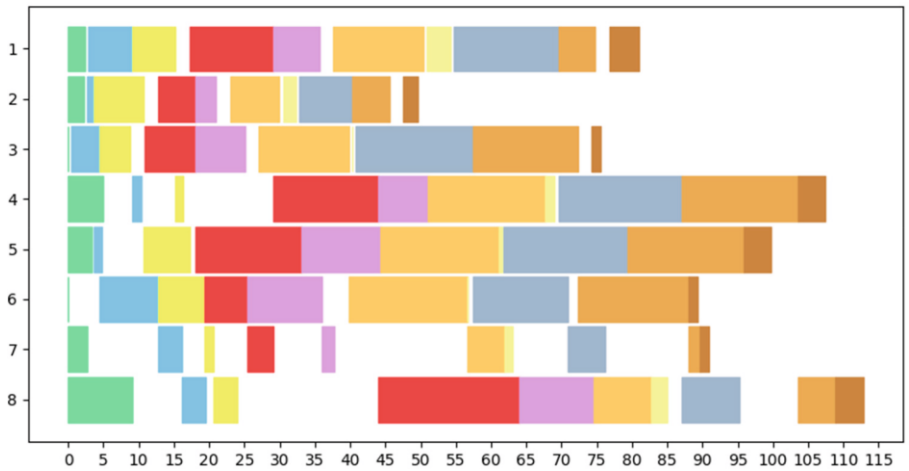

FIGURE 4 | The Gantt chart for the pilot problem by EDD dispatching method.

TABLE 4 | The population size, and the GA operators.

Factors	Values
Population size P	10
Crossover probability P_c	0.8;
Crossover method	*POX*
Mutation probability P_m	0.2
Mutation method	SWAP
Replacement method	*Acceptance threshold*
Replacement threshold K	2

4. The GA model for the flow shop scheduling problem

The previously mentioned FSS problem is an NP-hard problem, and the total number of possible solutions is 10!, which is equivalent to 3,628,800. The problem is solved by applying the GA model in the same steps as described in the section "2.3. Research methodology".

Step 1: Set the GA model's initial conditions

In this stage, GA model factors such as coding technique, GA parameters, and termination rule are set up.

The method of coding: Each chromosome is a string of 10 genes. Each gene corresponds to an order. The orders are numbered from 1 to 10. The sequence of genes represents the sequence of order scheduled. For example, the chromosome of EDD scheduling method is as follows:

$$C = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$$

The GA factors include fitness function, the population size, and the GA operators. Here is how we characterize F, the fitness function:

$$F_i = T_{max} - T_i.$$

TABLE 5 | The initial population $P^{(0)}$ with fitness values.

$P^{(0)}$	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi
EDD	1	2	3	4	5	6	7	8	9	10	1217.6710
SPT	5	3	2	1	4	10	7	8	9	6	1210.7427
LPT	10	7	1	5	2	3	4	6	9	8	1120.0530
R1	1	5	4	3	2	6	10	9	8	7	1094.8691
R2	5	3	4	1	2	10	8	9	6	7	986.0005
R3	6	8	7	9	10	1	3	2	4	5	455.6409
R4	10	6	9	7	8	5	1	4	2	3	315.4120
R5	8	10	7	9	6	3	5	2	4	1	283.1211
R6	7	8	10	9	6	2	3	5	4	1	260.3407
R7	8	9	6	4	3	2	5	1	7	10	0.0000

TABLE 6 | The initial population $P^{(0)}$ with selection and cumulative probabilities.

$P^{(0)}$	F_i	P_i	CP_i
EDD	1217.6710	0.1754	0.1754
SPT	1210.7427	0.1744	0.3497
LPT	1120.0530	0.1613	0.5110
R1	1094.8691	0.1577	0.6687
R2	986.0005	0.1420	0.8107
R3	455.6409	0.0656	0.8763
R4	315.4120	0.0454	0.9217
R5	283.1211	0.0408	0.9625
R6	260.3407	0.0375	1.0000
R7	0.0000	0.0000	1.0000

When F_i and T_i represent the fitness and objective values of chromosome i, respectively, T_{max} refers to the highest possible value in the population. **Table 4** provides an overview of the population size as well as the GA operators.

TABLE 7 | The chromosomes selected into elite population $P_E^{(0)}$.

RN	0.8190	0.7430	0.4142	0.2922	0.1047	0.3187	0.3009	0.0508	0.7966	0.9298
$P_E^{(0)}$	R3	R2	LPT	SPT	EDD	SPT	SPT	EDD	R2	R5

TABLE 8 | The chromosomes selected into the crossover list P_c.

$P_E^{(0)}$	R3	R2	LPT	SPT	EDD	SPT	SPT	EDD	R2	R5
RN	0.1132	0.8678	0.1592	0.4801	0.8902	0.7269	0.9400	0.3963	0.9875	0.8699
P_c	R3	–	LPT	SPT	EDD	SPT	–	EDD	–	–

TABLE 9 | Population P^C.

P^C	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi
C1	2	1	3	4	5	10	7	8	9	6	1245.02
C2	5	3	1	2	4	6	7	8	9	10	1156.34
C3	10	1	5	2	3	6	7	4	9	8	1083.70
C4	1	7	2	3	4	5	8	6	9	10	1080.84
C5	7	9	10	1	3	6	2	8	4	5	603.65
C6	6	8	1	2	3	4	5	7	9	10	553.08
C7	10	7	1	5	4	2	3	9	8	6	1104.16
C8	5	3	2	1	10	7	4	6	8	9	1069.54
C9	8	7	10	1	3	2	4	5	9	6	806.04
C10	6	5	3	9	2	1	4	10	7	8	611.50
C11	6	8	7	5	10	1	3	2	9	4	617.67
C12	10	7	1	9	2	3	4	6	8	5	941.70

The termination rule: After 10 iterations, there is no change in the population's best objective value *Tbest*.

Step 2: Create the first population $P^{(0)}$, set k = 0

The initial population comprises 10 chromosomes. There are 3 chromosomes generated from 3 heuristic rules, EDD, SPT, and LPT. The remaining chromosomes R1, . . . , R7 are randomly generated.

$$\mathbf{P^{(0)}} = \{\text{EDD, SPT, LPT, R1, R2, R3, R4, R5, R6, R7}\}$$

The chromosomes in the initial population with their fitness values are shown in **Table 5**.

Step 3: Establish the elite population $P_E^{(k)}$

This stage involves selecting a subset of the population, $\mathbf{P^{(k)}}$, to form the elite population, $\mathbf{P_E^{(k)}}$. Based on its fitness value F_i, each member of the present population is given a certain chance of being included in the elite population $\mathbf{P_E^{(k)}}$, denoted by the selection probability P_i.

$$P_i = \frac{F_i}{\sum_{1}^{10} F_i}$$

The selection probabilities P_i and cumulative probabilities CP_i of chromosomes in the initial population are calculated and shown in **Table 6**.

Based on CP_i, 10 random numbers RN are generated, the chromosomes selected into the population $\mathbf{P_E^{(0)}}$ are as in **Table 7**.

$$\mathbf{P_E^{(0)}} = \{\text{R3, R2, LPT, SPT, EDD, SPT, SPT, EDD, R2, R5}\}$$

Step 4: Establish the genetic population $P_G^{(k)}$

The newly created chromosome is a part of the genetic population known as $\mathbf{P_G^{(k)}}$, which was produced by the crossover and mutation operators.

A crossover probability of 0.8 is applied when selecting the chromosomes of $\mathbf{P_E^{(0)}}$ to be included in the $\mathbf{P_c}$ list of potential crossover partners. After producing 10 random numbers RN, the set $\mathbf{P_c}$ is calculated, and the results are presented in **Table 8**.

$$P_c = \{\text{R3, LPT, SPT, EDD, SPT, EDD}\}$$

By using the POX approach, we choose to cross over 6 pairs of chromosomes in population $\mathbf{P_c}$, which results in the addition of 12 new chromosomes to population $\mathbf{P^C}$, as shown in **Table 9**.

With a mutation probability of 0.2, the chromosomes of $\mathbf{P_E^{(0)}}$ are as well chosen for inclusion in the mutation list $\mathbf{P_m}$. Ten sets of random numbers (RN) are generated, and then $\mathbf{P_m}$ is calculated and displayed in **Table 10**.

$$\mathbf{P_m} = \{\text{EDD}\}$$

TABLE 10 | The chromosomes selected into the mutation list P_m.

$P_E^{(0)}$	R3	R2	LPT	SPT	EDD	SPT	SPT	EDD	R2	R5
RN	0.7629	0.6461	0.2326	0.7462	0.1376	0.7797	0.4283	0.6789	0.2254	0.5371
P_m	–	–	–	–	EDD	–	–	–	–	–

TABLE 11 | Population P^M.

P^M	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi
M1	1	10	3	4	5	6	7	8	9	2	1030.69

TABLE 12 | The next population $P^{(1)}$.

$P^{(1)}$	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi
C1	2	1	3	4	5	10	7	8	9	6	1245.02
EDD	1	2	3	4	5	6	7	8	9	10	1217.67
SPT	5	3	2	1	4	10	7	8	9	6	1210.74
C2	5	3	1	2	4	6	7	8	9	10	1156.34
LPT	10	7	1	5	2	3	4	6	9	8	1120.05
C7	10	7	1	5	4	2	3	9	8	6	1104.16
R1	1	5	4	3	2	6	10	9	8	7	1094.8691
C3	10	1	5	2	3	6	7	4	9	8	1083.70
C4	1	7	2	3	4	5	8	6	9	10	1080.84
C8	5	3	2	1	10	7	4	6	8	9	1069.54

As can be seen in **Table 11**, the SWAP method is used to select for mutations in each chromosome in $\mathbf{P_m}$, leading to the gain of 1 additional chromosome in population $\mathbf{P^M}$.

A total of 13 additional chromosomes are added into the population $\mathbf{P_G^{(0)}}$ as a result of crossover and mutation:

$$\mathbf{P_G^{(0)}} = \{C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, M1\}$$

Step 5. Develop the next population $\mathbf{P^{(k+1)}}$

The next population, $\mathbf{P^{(k+1)}}$, is formulated with the help of the replacement operator at this stage. If chromosomes from $\mathbf{P_G^{(k)}}$ have fitness values higher than the threshold value, they will be included into the current population $\mathbf{P^{(k)}}$ in order to generate the following population $\mathbf{P^{(k+1)}}$. With K = 2:

$$n = \frac{P}{K} = \frac{10}{2} = 5$$

The threshold value is the value of the 5th chromosome of $\mathbf{P^{(k)}}$ in the ranking. For this iteration, the 5th chromosome of $\mathbf{P^{(0)}}$ in the ranking is R2, and the threshold value is 986.0005. Chromosomes C1, C2, C3, C4, C7, C8, and M1 are selected for inclusion to $\mathbf{P^{(1)}}$.

In order to maintain the same total number of individuals in the $\mathbf{P^{(1)}}$ population, the chromosomes M1, R7, R6, R5, R4, R3, R2, and R1 are eliminated. The next population, $\mathbf{P^{(1)}}$, is calculated and displayed as in **Table 12**.

Step 6. Make sure the termination rule is followed

The best chromosome after the first iteration is C1, which has an objective value of 188.60 and appears only once in the population. Since the rule of termination has not been met, iteration 2 will be carried out. **Table 13** displays the outcomes after 17 iterations.

TABLE 13 | The results after 17 iterations.

Iteration	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Tbest
0	1	2	3	4	5	6	7	8	9	10	215.95
1	2	1	3	4	5	10	7	8	9	6	188.60
2	1	2	3	4	5	7	9	8	10	6	167.66
3	1	2	3	4	5	7	9	8	10	6	167.66
4	1	2	3	4	5	7	8	9	10	6	141.62
5	1	2	3	4	5	10	8	7	9	6	129.87
6	1	2	3	4	5	10	8	7	9	6	129.87
7	1	3	2	4	5	8	7	10	9	6	126.43
8	1	3	2	4	5	10	8	7	9	6	123.07
9	1	3	2	4	5	10	8	7	9	6	123.07
	1	3	2	4	5	10	8	7	9	6	123.07
17	1	3	2	4	5	10	8	7	9	6	123.07

Seeing that from the 8th iteration to the 17th iteration, the best objective value remains the same, the termination rule is satisfied; hence, the algorithm ends. The scheduling result in this run is as follows:

$$S = (1, 3, 2, 4, 510, 8, 7, 9, 6),\ T = 123.07\ (h).$$

Step 7. Run the algorithm a number of times to choose the best scheduling result

The algorithm is run 10 times with the results as shown in **Table 14**.

TABLE 14 | The results after 10 runs.

Run	Tbest	N
1	123.07	17
2	138.35	16
3	143.03	16
4	123.07	35
5	124.69	24
6	129.87	14
7	125.04	17
8	124.69	17
9	126.43	20
10	123.07	19

The best scheduling result is found on the 1st run, with a number of iterations n of 17. The sequence of dispatching S, the values of the objective function are as follows:

$$S = (1, 3, 2, 4, 510, 8, 7, 9, 6),$$

$$T = 123.07 (h).$$

The objective value by the GA model, 123.07 (h) is smaller than the objective value of EDD models, 215.95 (h).

5. Conclusion

With the goal of minimizing the total weighted tardiness time and constraint on changeover time in operations, the Flow Shop Scheduling Problem with 10 orders on 4 machines has been created. The Flow Shop Scheduling Problem has been successfully resolved with the help of the GA model. Compared to the heuristic EDD technique, the results reveal that the GA model provides a lower objective value of weighted tardiness time.

Nevertheless, given that the model's factors, such as the population size, the crossover method and probability Pc, the mutation method and probability, as well as the method and parameter of the termination rule, are only determined empirically, the findings are not very impressive. In the upcoming study, the experimental design DOE will be utilized to identify the model parameters in order to produce more accurate results. Another area for research for the future is to apply the model to more extended order quantities.

Moreover, GA, a global search method, if combines with another local search method like Tabu search, the result would be better in terms of quality, better objective value, and cost, smaller number of iterations.

References

1. Phong N, Thu N. Application of GATS, a hybrid mega-heuristic model, and DOE, to solve flexible flow shop scheduling problems: a case study. *BOHR Int J Operat Manag Res Pract.* (2023) 2:28–35. doi: 10.54646/bijomrp.2023.14
2. Phong N, Thuy N. Application of Tabu Search, TS to solve a flow shop scheduling problem with changeover times in operations. A case study. *BOHR Int J Operat Manag Res Pract.* (2024) 3:1–7. doi: 10.54646/bijomrp.2024.22
3. Etiler O, Toklu B, Atak M, Wilson J. A genetic algorithm for flow shop scheduling problems. *J Operat Res Soc.* (2004) 55:830–5. doi: 10.1057/palgrave.jors.2601766
4. Engin O, Ceran G, Yilmaz MK. An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems. *Appl Soft Comput.* (2011) 11:3056–65. doi: 10.1016/j.asoc.2010.12.006
5. Umam MS, Mustafid M, Suryono S. A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. *J King Saud Univ Comput Informat Sci.* (2022) 34:7459–67. doi: 10.1016/j.jksuci.2021.08.025
6. Burduk A, Musiał K, Kochan´ska J, Górnicka D, Stetsenko A. Tabu Search and genetic algorithm for production process scheduling problem. *Sci J Logist.* (2019) 5:181–9. doi: 10.17270/J.LOG.2019.315