BOHR International Journal of Operations Management Research and Practices
2025, Vol. 4, No. 1, pp. 12–20
DOI: 10.54646/bijomrp.2025.33
www.bohrpub.com

CASE STUDY

Application of TSGA, a hybrid meta-heuristic model, to solve a real size problem of flow shop scheduling with changeover times in operations

Phong Nguyen Nhu[1]*, Thuy Nhi Nguyen Thi[1] and Tu Anh Nguyen Nhu[2]

[1]Department of Industrial Systems Engineering, University of Technology – VNU-HCM, Ho Chi Minh City, Vietnam
[2]Department of Information Technology, Monash University, Melbourne, Australia

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

Received: 08 January 2025; **Accepted:** 03 March 2025; **Published:** 22 March 2025

Flow shop scheduling (FSS) problems are NP-hard combinatorial optimization problems. It's quite difficult to achieve an optimal solution for real size problems with mathematical modeling approach because of its NP-hard structure. Meta-heuristic algorithms, like Tabu Search (TS) and genetic algorithm (GA), play a major role in searching for near-optimal solutions for NP-hard optimization problems. In the case study, the current scheduling method is ineffective, and the total tardiness time of orders is still quite high. This paper develops a Tabu Search and Genetic Algorithm (TSGA) model for solving the real FSS problem, with the objective of dispatching orders more effectively than the current dispatching method. The TSGA model is a hybrid meta-heuristic model, combining TS and GA. In the model, TS is used as the platform for local search, and GA is used to support TS in global search. The performance of the model is compared with the traditional heuristic being used. The result indicates that the model is a good approach for FSS problems. However, the factors of the model are only selected empirically; therefore, the results are not particularly satisfactory. The future research is to use experimental design (DOE) to determine the model parameters to get better suboptimal results.

Keywords: hybrid meta-heuristics, Tabu Search, genetic algorithm, flow shop scheduling problems, changeover times.

Introduction

Scheduling is the allocation of resources to perform a collection of tasks over a period of time. Scheduling problems determine the order or sequence for processing a set of jobs through several machines in an optimal manner. Flow shop scheduling (FSS) problems consider m different machines and n jobs, each job consists of m operations, each operation requires a different machine, and all the jobs are processed in the same processing order.

The problem to be solved is a FSS problem with the assumption that the orders are ready at the start of the scheduling process. The objective of the problem is to minimize the total weighted tardiness of orders. The constraints are on the sequence of orders, on the sequence of operations in the orders, and on machine changeover time. The model of the problem is built based on the above assumptions, objectives, and constraints.

The Tabu Search and Genetic Algorithm (TSGA) algorithm is built based on the combination of Tabu Search (TS) and genetic algorithm (GA) with the foundation of TS. TS generates the initial solution and the neighborhood population. Then, GA generates an elite population from the neighborhood population, followed by generation genetic population from the elite population. Then TS finds the next solution from the neighborhood and genetic populations.

Based on the model of the problem, the TSGA algorithm will be used to solve the pilot problem of small size, and the real problem of real size. The solutions from the TSGA algorithm will be compared with the corresponding solutions of the currently used heuristic model to evaluate the effectiveness of the algorithm.

Literature review

FSS problems consider different machines and different jobs. Each job consists of different operations, each operation requires a different machine, and all the jobs are processed in the same processing order (1). FSS is categorized as an NP-hard problem, so it is difficult to develop algorithms to solve it (2).

TS, suggested by Glover and Laguna in 1997, is one of the most popular meta-heuristics to find solutions to various combinatorial optimization problems. Jatinder N. D. Gupta, Nagarajan Palanim Uthu, and Chuen-Lung Chen designed a TS-based heuristic for the two-stage flow shop problem with the makespan minimization as the primary criterion and the minimization of total flow time as the secondary criterion (3). Phong Nguyen Nhu et al. applied TS to solve an FSS problem with changeover times in operations (4).

GA, first introduced by Holland in 1975, is an artificial intelligence, meta-heuristic search method used to solve scheduling problems. O. Etiler, B. Toklu, M. Atak, and J. Wilson developed a GA-based heuristic for the FSS problem with makespan as the criterion (5). Complex GA algorithms have also been researched to solve the FFS problem effectively. Orhan Engin, Gülsad Ceran, and Mustafa K. Yilmaz developed an efficient GA for hybrid FSS with multiprocessor task problems (6). Phong Nguyen Nhu et al. applied the GA to solve an FSS problem with changeover times in operations (7).

Hybrid meta-heuristics combining different meta-heuristic algorithms has proven effective in solving scheduling problems. Moch Saiful Umam, Mustafid Mustafid, and Suryono Suryono combined the TS process with a GA to solve the FSS problem to minimize makespan (2). Anna Burduk, Kamil Musiał, Joanna Kochanska, Dagmara Górnicka, and Anastasia Stetsenko applied TS and GA to solve production process scheduling problems and found that intelligent methods can find, in a relatively short time, the solution that is close to the optimal and acceptable from the problem point of view (8). Phong Nguyen Nhu et al. applied TSGA, a hybrid meta-heuristic model, to solve an FFS problem with changeover times in operations (1). Phong Nguyen Nhu et al. applied GATS, a hybrid meta-heuristic model, to solve an FSS problem with changeover times in operations (9). Phong Nguyen Nhu et al. applied GATS, a hybrid meta-heuristic model and the design of experiment (DOE), to solve flexible FSS problems (10).

Research methodology

FSS problem is a NP-hard problem with a large size of solution space. The methodology, used in this research to solve the problem, includes three phases:

(1) Phase A: Construct the model of the pilot FSS problem
(2) Phase B: Use TSGA model to solve the pilot FSS problem
(3) Phase C: Use TSGA model to solve the real FSS problem

In phase A, the model of the pilot problem with a small size is formulated with the objective of minimizing the total weighted tardiness time and constraint on operation changeover time. Based on the model of the problem, a TSGA model is developed and used to solve the pilot problem of small size in phase B and the real problem of real size in phase C.

In the TSGA model, TS is used to perform a local search and GA is used to support TS in a global search on the solution space. The TSGA procedure is as follows:

(1) Step 1: Initialize the TSGA model.
(2) Step 2: Generate the initial solution S0, set k = 0.
(3) Step 3: Generate the neighborhood population PN(k).
(4) Step 4: Generate elite population PE(k).
(5) Step 5: Generate the genetic population PG(k).
(6) Step 6: Find the next solution Sk+1. Set k = k+1.
(7) Step 7: Check the termination rule. If No, return to step 3. If Yes, finish the loop.
(8) Step 8: Run the algorithm a number of times to choose the best scheduling result.

Step 1 sets up factors of TSGA models, including the method of coding, the TS factors, the GA factors, and the termination rule.

- The TS factors: the evaluation function, the tabu list, and the parameters of the TS operators (i.e., the neighborhood and selection operators).
- The GA factors: the fitness function and the parameters of GA operators (i.e., the crossover, mutation, and search operators).

Step 2 generates the initial solution S_0 and resets the iteration counter k to 0. A good initial solution will be a good starting point for the search. The initial solution is often chosen by heuristic methods.

Step 3 uses the neighborhood operator to generate the neighborhood population $P_N^{(k)}$ from the current solution S_k.

Step 4 uses the selection operator to generate an elite population $P_E^{(k)}$ from the neighborhood population $P_N{}^{(k)}$.

Step 5 uses the crossover and mutation operators to generate a genetic population $P_G{}^{(k)}$ from the elite population $P_E{}^{(k)}$.

Step 6 finds the next solution S_{k+1} by using the search operator to find the best solution in the solution regions defined by $P_N^{(k)}$ and $P_G^{(k)}$.

Step 7 checks the termination rule. If the rule is not satisfied, the next iteration is executed by returning to step 3. If the rule is satisfied, finish the run.

Step 8 runs the algorithm several times to choose the best scheduling result among the runs.

The pilot FSS problem

The pilot problem to be solved, as shown in (1, 4, 7, 9), is an FSS problem with 10 orders (O_i, i = 1÷10), scheduling on four machines (M_1, M_2, M_3, M_4). Each order has three parts (P1, P2, P3), processed in eight operations (O_j, j = 1÷8), distributed on the four machines as in **Figure 1**.

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 time P_{ij} of order i (i = 1÷10) on operation j (j = 1÷8), is estimated in **Table 2**.

The changeover times in hours on operation j 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 (i = 1÷10), and the next order, i′ (i′ = 1÷10), as shown in **Table 3**.

The model is set up with variables TS_{ij} being the start time, TE_{ij} being the completion time of order i at operation j, and T_i being the tardiness time of order i. The constraints on the sequence of operations in each order are as follows:

$$TS_{i4} \geq TE_{i1},$$
$$TS_{i5} \geq TE_{i2},$$
$$TS_{i6} \geq TE_{i3},$$
$$TS_{i7} \geq TE_{i5},$$
$$TS_{i8} \geq \max(TE_{i4}, TE_{i6}, TE_{i7})$$

In 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.

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

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

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

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

$$T_i = \text{Max}(0, TE_{i8} - D_i)$$

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

$$Tbest = Min\ T,\ T = \sum_{i=1}^{10} w_i T_i$$

The company is currently using the EDD dispatching method. The sequence of dispatching S, the value of the objective function, and the Gantt chart are as shown in **Figure 2**.

$$S = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);$$
$$T = 215.95(\text{h})$$

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

FIGURE 1 | Order processing in the FSS problem.

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

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

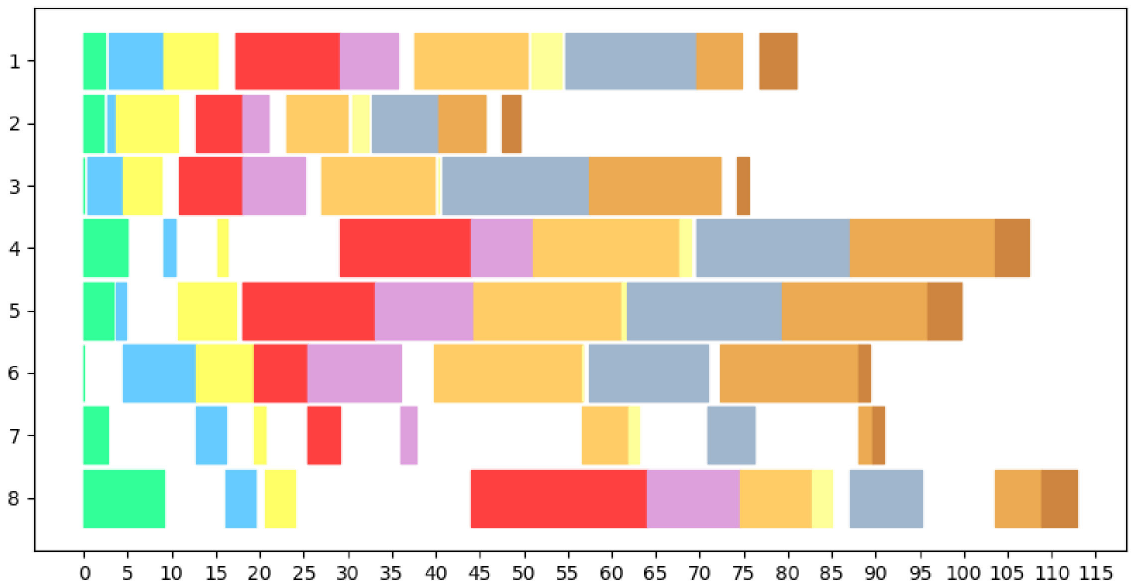

FIGURE 2 | The Gantt chart for the pilot problem by EDD dispatching method.

The TSGA model for the pilot FSS problem

The above pilot FSS problem is a NP-hard problem with the size of the solution space of 10!, or 3,628,800. The above TSGA model is applied to solve the pilot problem as follows.

Step 1: Initialize the TSGA model

This step sets up factors of TSGA models, including the method of coding, the TS parameters, the GA parameters, and the termination rule.

The method of coding: The orders are numbered from 1 to 10. Each code is a string of 10 numbers. Each number corresponds to an order. The sequence of numbers G_i (i = 1÷10) represents the sequence of order scheduled:

$$C = (G_1, G_2, G_3, G_4, G_5, G_6, G_7, G_8, G_9, G_{10})$$

The TS parameters include the evaluation function, the current best value, the parameters of the TS operators, and the tabu list.

- The evaluation function is the objective function T used to evaluate scheduling codes.
- The current best value Tbest is the best objective value updated after each iteration.
- The TS operators include the neighborhood and the search operator.

The neighborhood operator uses the method of permutation of adjacent numbers in the string to find the neighborhood. The search operator finds the best solution in the neighborhood solution region to prepare for the next iteration, if any. This solution must not be on the tabu list. The tabu list will contain the solutions found in the previous loops and is updated after each iteration.

The GA parameters include the fitness function and the parameters of GA operators. The fitness function F is

defined as:

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

- F_i, T_i: the fitness and objective values of chromosome i
- T_{max}: the maximum objective value in the population

The crossover method is POX (precedence operation crossover), and the mutation method is SWAP. The crossover probability P_c and the mutation probability P_m are chosen as follows:

$$P_c = 0.8$$
$$P_m = 0.2$$

The termination rule: The best objective value Tbest does not improve after 10 consecutive iterations.

Step 2: Generate the initial solution S_0. Set k = 0

A good initial solution will be a good starting point for the search. The initial solution is often chosen by heuristic methods. With the objective of minimizing lateness, the initial solution selected from the EDD method is represented as the following string.

$$S_0 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$
$$Tabu\ list = \{C_0\}$$
$$Tbest = 215.95$$

Step 3: Generate the neighborhood population $P_N^{(k)}$

This step uses the neighborhood operator to generate the neighborhood population $\mathbf{P}_N^{(k)}$ from the current string. The neighborhood population $\mathbf{P}_N^{(0)}$ generated from the initial string is as shown in **Table 4**.

$$\mathbf{P}_N^{(0)} = \{N1, N2, N3, N4, N5, N6, N7, N8, N9\}$$

Generate elite population $P_E^{(k)}$

This step uses the selection operator to generate elite population $\mathbf{P}_E^{(k)}$ from the neighborhood population $\mathbf{P}_N^{(k)}$. Each chromosome in $\mathbf{P}_N^{(k)}$ has a corresponding fitness value F_i and is selected for inclusion in the elite population P_E with selection probability P_i determined as follows:

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

With population $\mathbf{P}_N^{(0)}$ the values of F_i and P_i are calculated as shown in **Table 5**.

Based on P_i, 10 random numbers are generated, the strings selected into $\mathbf{P}_E^{(0)}$ are as follows:

$$\mathbf{P}_E^{(0)} = \{N1, N2, N3, N4, N5, N1, N7, N4, N4\}$$

Step 5: Generate the genetic population $P_G^{(k)}$

This step uses the crossover and mutation operators to generate genetic population $P_G^{(k)}$ from the elite population $P_E^{(k)}$. The genetic population $P_G^{(k)}$ includes the new strings generated from the crossover and mutation operators.

The strings of $\mathbf{P}_E^{(0)}$ are selected to be included in the crossover list $\mathbf{P}_c$ with the crossover probability of 0.8. After generating random numbers, the set $\mathbf{P}_c$ is determined as follows:

$$\mathbf{P_c} = \{N1, N3, N4, N5\}$$

Each pair of strings in $\mathbf{P}_c$ is selected to cross over by the POX method, resulting in 12 new strings in population $\mathbf{P}^C$ as shown in **Table 6**.

The strings of $\mathbf{P}_E^{(0)}$ are also selected to be included in the mutation list $\mathbf{P}_m$ with the mutation probability of 0.2. After generating random numbers, the set $\mathbf{P}_m$ is determined as

TABLE 4 | The neighborhood population $P_N^{(0)}$.

$P_N^{(0)}$	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Ti
N1	2	1	3	4	5	6	7	8	9	10	224.60
N2	1	3	2	4	5	6	7	8	9	10	203.75
N3	1	2	4	3	5	6	7	8	9	10	242.49
N4	1	2	3	5	4	6	7	8	9	10	175.90
N5	1	2	3	4	6	5	7	8	9	10	241.39
N6	1	2	3	4	5	7	6	8	9	10	221.26
N7	1	2	3	4	5	6	8	7	9	10	224.11
N8	1	2	3	4	5	6	7	9	8	10	239.71
N9	1	2	3	4	5	6	7	8	10	9	175.65

TABLE 5 | The values of F_i and P_i of strings in $P_N^{(0)}$.

$P_N^{(0)}$	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi	Pi
N1	2	1	3	4	5	6	7	8	9	10	17.89	0.08
N2	1	3	2	4	5	6	7	8	9	10	38.74	0.17
N3	1	2	4	3	5	6	7	8	9	10	0.00	0.00
N4	1	2	3	5	4	6	7	8	9	10	66.59	0.29
N5	1	2	3	4	6	5	7	8	9	10	1.10	0.00
N6	1	2	3	4	5	7	6	8	9	10	21.23	0.09
N7	1	2	3	4	5	6	8	7	9	10	18.38	0.08
N8	1	2	3	4	5	6	7	9	8	10	2.78	0.01
N9	1	2	3	4	5	6	7	8	10	9	66.84	0.29

TABLE 6 | The population P^C.

P^C	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Ti
C1	1	2	3	4	5	6	7	8	9	10	215.95
C2	1	4	3	2	5	6	7	8	10	9	254.34
C3	1	4	3	2	5	7	6	8	9	10	294.78
C4	1	4	3	2	5	6	7	8	10	9	294.78
C5	1	2	3	4	5	6	7	8	9	10	215.95
C6	1	4	3	2	5	6	7	8	10	9	254.34
C7	1	4	3	2	5	7	6	8	9	10	294.78
C8	1	2	3	4	5	6	7	8	9	10	215.95
C9	1	4	3	2	5	6	7	8	10	9	254.34
C10	1	4	3	2	5	7	6	8	9	10	294.78

TABLE 7 | The population P^M.

$P_N^{(k)}$	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Fi
N1	1	2	3	4	5	10	7	8	9	6	1285.22

follows:

$$\mathbf{P_m} = \{N_5\}$$

Each chromosome in $\mathbf{P}_m$ is selected to mutate by the SWAP method, resulting in new chromosomes in population $\mathbf{P}_m$ as shown in **Table 7**.

After crossover and mutation, 13 new chromosomes are created in $\mathbf{P}_G^{(0)}$:

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

Step 6. Find the next solution S_{k+1}

This step uses the search operator to find the best solution in the solution regions defined by $\mathbf{P}_N^{(k)}$ and $\mathbf{P}_G^{(k)}$. The search operator relies on the objective function and the tabu list to find the best solution that is not in the tabu list. This solution will be selected for the next iteration, if any. This step also updates the tabu list and the current best Tbest if the objective value of the solution is better than the current Tbest.

At iteration 1, based on $\mathbf{P}_N^{(k)}$, $\mathbf{P}_G^{(k)}$, and the current tabu list, the next solution is:

$$\begin{aligned} N9 &= (1, 2, 3, 4, 5, 6, 7, 8, 10, 9) \\ TL &= \{C0, N9\} \\ Tbest &= 175.65 \end{aligned}$$

Step 7. Check the termination rule

After iteration 1, N1 is the best string with the best objective value of 175.65, appearing only once. The termination rule is not satisfied, so iteration 2 is executed. The results after 16 iterations are shown in **Table 8**.

TABLE 8 | The results after 16 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	1	2	3	4	5	6	7	8	10	9	175.65
2	1	2	3	5	4	6	7	8	10	9	135.60
3	1	2	3	5	4	6	7	8	10	9	135.60
4	1	2	3	5	4	6	7	8	10	9	135.60
5	1	2	3	5	4	6	7	8	10	9	135.60
6	1	2	3	4	5	10	8	7	9	6	129.87
...	1	2	3	4	5	10	8	7	9	6	129.87
16	1	2	3	4	5	10	8	7	9	6	129.87

TABLE 9 | The results after five runs.

Run	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	Tbest	n
1	1	2	3	4	5	10	8	7	9	6	129.87	16
2	1	3	2	4	5	7	8	9	10	6	124,67	18
3	1	3	2	4	5	7	8	9	10	6	124,67	14
4	1	3	2	4	5	10	8	7	9	6	123.07	16
5	1	3	2	4	5	10	8	7	9	6	123.07	14

The best objective value Tbest remains the same from the 6th iteration to the 14th iteration, the termination rule is satisfied, and the algorithm ends. The scheduling result in this run is as follows:

$$S = (1, 2, 3, 4, 5, 10, 8, 7, 9, 6)$$
$$\text{Tbest} = 129.87(\text{h})$$

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

The algorithm is run five times with the results as shown in **Table 9**.

The best scheduling result is found on the 5th run, with a number of iterations NoI of 14. The sequence of dispatching S, the value of the objective function, and the Gantt chart are shown in **Figure 3**.

$$S = (1, 3, 2, 4, 5, 7, 8, 9, 10, 6)$$
$$T = 123.07$$

The objective value according to TSGA model (123.07 h) is better than the objective value according to the EDD heuristic currently used (215.95 h).

The TSGA model for the real FSS problem

The real problem to be solved is a FSS problem with 120 orders, O_i (i = 1÷120), scheduling on four machines (M_1, M_2, M_3, M_4). Each order has three parts (P1, P2, P3), processed in eight operations, O_j (j = 1÷8), distributed on the four machines as in **Figure 1**.

Problem data has been collected to estimate the parameters of the weight W_i, the due date D_i of order i (i = 1÷120), of the processing time P_{ij} of order i (i = 1÷120), on operation j (j = 1÷8), and of the changeover times S_j (j = 1÷8).

The company is currently using the EDD dispatching method. The sequence of the order to be scheduled by the ESS model is as follows:

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120]

The total weighted tardiness time T and the number of late delivery orders N of the real size problem is as follows:

$$T = 354.95 \text{ (h)}$$
$$N = 31.$$

The above TSGA model has been applied to solve the real problem, with the size larger than the pilot problem size; thus, some parameters or factors of the TSGA model have been changed to suit the real problem. The parameters or factors of the TSGA model for the real problem are as in **Table 10**.

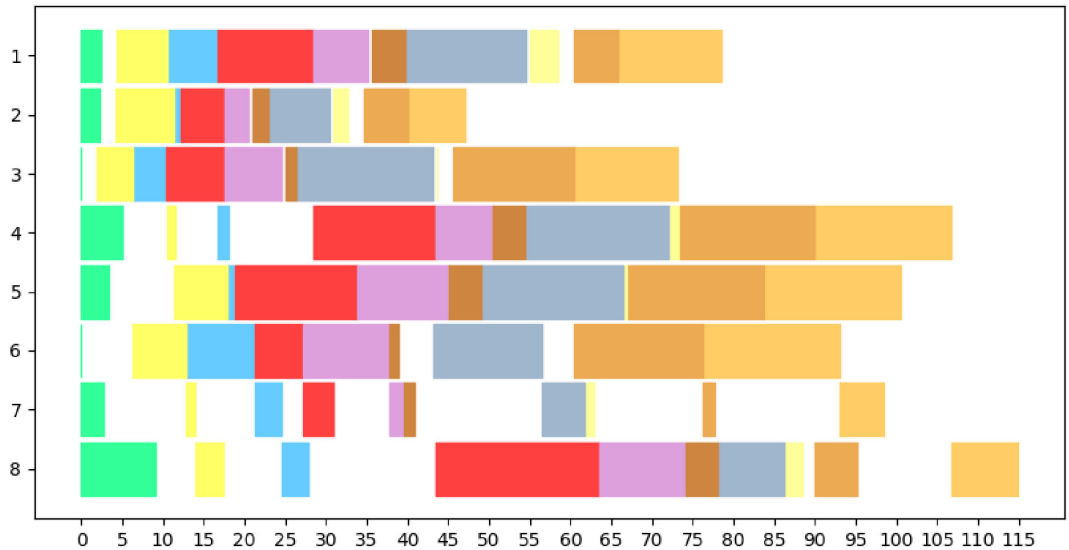

FIGURE 3 | The Gantt chart for the pilot problem by TSGA model.

TABLE 10 | The parameters or factors of the TSGA algorithm.

Parameters or factors	Values
Initial solution	EDD
Neighborhood operator	Permutation
Tabu list size	Not limited
Population size	25
Crossover method	POX
Mutation method	Swap
P_c	0.8
P_m	0.4
The termination rule	The best objective value Tbest does not improve after 10 consecutive iterations or CPU time is more than 3 h

The sequence of the orders to be scheduled by the TSGA model is as follows:

[1, 2, 3, 103, 5, 119, 6, 69, 8, 9, 11, 14, 12, 15, 13, 16, 17, 76, 18, 78, 21, 22, 24, 101, 25, 27, 10, 90, 28, 29, 31, 30, 33, 35, 32, 34, 37, 36, 38, 44, 39, 40, 41, 42, 19, 45, 47, 46, 49, 48, 52, 54, 51, 53, 55, 56, 57, 59, 91, 60, 61, 62, 63, 65, 64, 66, 7, 67, 68, 70, 85, 73, 75, 72, 77, 74, 83, 43, 79, 20, 84, 81, 82, 71, 87, 88, 89, 86, 26, 92, 58, 94, 95, 93, 96, 98, 99, 97, 100, 23, 102, 4, 105, 104, 106, 107, 108, 109, 111, 110, 50, 112, 80, 117, 116, 115, 114, 113, 118, 120].

The total weighted tardiness time T and the number of late delivery orders N of the real size PMS problem are as follows:

$$T = 284.13 \text{ (h)}$$
$$N = 5.$$

The total weighted tardiness time T has decreased by 20 % from 354.95 (h) to 284.13 (h). The number of late delivery orders N has decreased from 31 to 5.

Conclusion

The FSS problem to be solved is modeled with the objective to minimize the total weighted tardiness of orders, and constraints on the sequence of orders, on the sequence of operations in the orders, and on machine changeover time.

The TSGA algorithm is constructed based on the combination of TS and GA with the foundation of TS. In the model, TS is used as the platform to perform a local search of the solution space, and GA is used to perform a global search to support TS.

Based on the problem model, the TSGA algorithm is used to solve the pilot problem of a small size, with 10 orders and scheduling on four machines. The total weighted tardiness time according to the TSGA model (123.07 h) is better than the total weighted tardiness time according to the EDD heuristic currently used (215.95 h).

Then the TSGA algorithm is also used to solve the real FSS problem with 120 jobs on 4 machines. The results show that the TSGA model gives better results than the heuristic EDD method being used. The total weighted tardiness time T has decreased by 20 % from 354.95 (h) to 284.13 (h). The number of late delivery orders N has decreased from 31 to 5.

However, the factors of the model, including the method of finding the neighborhood strings, the crossover probability P_c, the mutation probability P_m, and the method and parameter of the termination rule, are only selected empirically; therefore, the results are not particularly satisfactory. The future research should use DOE to determine the model parameters for achieving better suboptimal results.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Nhu PN, Thi TNN. Application of TSGA, a hybrid mega heuristic model, to solve flow shop scheduling problems FFS with changeover times in operations. A case study. *The 4th International Conference on Applied Convergence Engineering (ICACE 2023)* Ho Chi Minh City, Vietnam: HCMC University of Technology, VNUHCM (2023).
2. 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 Comp Inf Sci.* (2022) 34:7459–67.
3. Gupta JND. Designing a Tabu Search algorithm for the two-stage flow shop problem with secondary criterion. *Prod Plann Cont Manage Oper.* (1999) 10:251–65. doi: 10.1080/095372899233217
4. Nhu PN, Thi TNN. Application of Tabu Search, TS to solve a flow shop scheduling problem with changeover times in operations. A case study. *BOHR Int J Oper Manag Res Pract.* (2024). doi: 10.54646/bijomrp.2024.22
5. Etiler O, Toklu B, Atak M, Wilson J. A genetic algorithm for flow shop scheduling problems. *J Oper Res Soc.* (2004) 55:830–5. doi: 10.1057/palgrave.jors.2601766
6. Engin O, Ceran G, Yilmaz MK. An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems. (2007). *Appl Soft Comp.* (2011) 11:3056–65. doi: 10.1016/j.asoc.2010.12.006
7. Nhu PN, Thi KNN, Thi THTV. Application of genetic algorithm, GA to solve a flow shop scheduling problem with changeover times in operations: a case study. *BOHR Int J Oper Manag Res Pract.* (2024). doi: 10.54646/bijomrp.2024.25
8. Burduk A, Musiał K, Kochanska J, Górnicka D, Stetsenko A. Tabu Search and genetic algorithm for production process scheduling problem. *LogForum.* (2019) 15:181–9. doi: 10.17270/J.LOG.2019.315
9. Nhu PN, Thi KNN. Application of GATS, a hybrid mega heuristic model, to sovle flow shop scheduling problems FFS with changeover times in operations. A case study. *The 4th International Conference on Applied Convergence Engineering (ICACE 2023)* University of Technology, VNU HCM (2023).
10. Nhu PN, Le TTN. Application of GATS, a hybrid mega heuristic model, and DOE, design of experiment, to solve flexible flow shop scheduling problems - a case study. *BOHR Int J Oper Manag Res Pract.* (2023). doi: 10.54646/bijomrp.2023.14