BOHR International Journal of Operations Management Research and Practices
2023, Vol. 2, No. 1, pp. 28–35
DOI: 10.54646/bijomrp.2023.14
www.bohrpub.com

CASE REPORT

Application of GATS, a hybrid mega-heuristic model, and DOE, to solve flexible flow shop scheduling problems: a case study

Phong Nguyen Nhu* **and Thu Trang Nguyen Le**

Industrial Systems Engineering, HCM University of Technology, Thành phố Hồ Chí Minh, Vietnam

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

Received: 17 July 2023; **Accepted:** 26 July 2023; **Published:** 14 August 2023

Flexible flow shop scheduling (FFSS) is an NP-hard combinatorial optimization problem. Solving this problem using mathematical modeling approaches is very difficult. Mega-heuristic algorithms, such as the genetic algorithm (GA) and tabu search (TS), are powerful tools for finding near-optimal solutions to problems of this type. This paper develops a GATS model by combining GA and TS for solving FFSS problems. In the model, GA is used as the platform for global search, and TS is used to support GA in local search. This paper also uses the design of experiments (DOE) to optimize the parameters of the GATS model. The performance of the models, GATS and GATS with DOE, is compared with traditional heuristics being used. The result indicates that the models are good approaches for FFSS problems.

Keywords: genetic algorithm, tabu search, design of experiment, flexible flow shop scheduling, mathematical modeling, parallel machine scheduling, setup time, batch production, hybrid mega heuristic model

Introduction

Scheduling problems define the sequence in which a group of orders is efficiently processed through several machines. Flow shop scheduling (FSS) problems consider m machines and n orders, and all orders are processed in the same sequence. Parallel machine scheduling (PMS) problems involve sending n jobs to m parallel machines. An order can be processed on any machine. Machines can be homogeneous or heterogeneous. Machines are homogenous when they have the same processing time for each job. Machines are heterogeneous when the times to process the same job on different machines are different.

Flow shop scheduling problems are problems that combine FFS problems and PMS problems. So, FFSS problems are more complicated than FFS problems and PMS problems. FFSS problems include dispatching n jobs across w stations; each station has several parallel machines. Each job is executed sequentially through the stations in the same sequence, and only one machine at each station is selected for execution.

The problem to be solved is an FFSS 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 tardiness of orders. The constraints are on the sequence of orders, on the sequence of operations in each order, on the allocation of orders on parallel machines in each station, on machine changeover time, and on batches produced in each machine.

The model of the problem is built based on the above assumptions, objectives, and constraints. The GATS algorithm is a combination of GA and TS, with the foundation of GA. Based on the model of the problem, the GATS algorithm will find a good solution for the problem. Design of experiment (DOE) is used to optimize the parameters of the GATS model. The solutions of GATS and GATS with DOE are compared with the solutions of the traditional heuristics being used to evaluate their effectiveness.

Literature review

Flexible flow shop scheduling

Flow shop scheduling is renowned and classified as an NP-hard optimization problem (Berlinska and Przybylski, 2021), and several algorithms have been developed to overcome this problem (1, Lee and Loong, 2019). Palmer et al. and Gupta et al. proposed heuristic methods to solve *n*-job problems on *m*-machines (2). Chen et al. applied GA to these problems. However, the problem solving time of GAs is quite long.

The PMS problem has two distinct decisions: allocation and sequencing. Allocation is a decision concerning the assignment of jobs to machines, while sequencing is to order the jobs assigned to each machine (3). When the number of machines is large, mega-heuristic algorithms such as GA and TS are usually used instead of optimal methods to solve the problem. To reduce the time to solve the problem, one solution is to coordinate between GA and TS (where GA is used for global search and TS is used for local search).

Genetic algorithm

Genetic algorithm, first introduced by Holland in 1975, is a powerful search engine to solve optimization problems using evolution and natural selection of selected individuals called chromosomes. Feasible solutions to the problem are modeled by chromosomes. The population of chromosomes represents the solution space of the problem. Chromosomes are represented by strings. A method of coding is used to define the string format for the chromosomes.

Each chromosome has a corresponding fitness value. The fitness function is a measure of the extent to which the objective of the problem is achieved. The fitness function is derived from the objective function of the problem.

An initial population is made up of several chromosomes. The next population is created by applying genetic operators to the chromosomes of the current generation. Genetic operators include selection, crossover, mutation, and replacement. The algorithm creates a new population from the existing population. The next population is probably better than the current population. This process is repeated until a specified stopping condition is met.

Tabu search

Tabu search, introduced by Glover and Laguna in 1997, is also an effective search engine to solve optimization problems. Feasible solutions to the problem are modeled by strings. The population of strings represents the solution space of the problem. Strings are represented by codes. A method of coding is used to define the string format for the codes.

Each string has a corresponding evaluation value. The evaluation function is a measure of the extent to which the objective of the problem is achieved. The evaluation function is derived from the objective function of the problem. The F-best value is the best evaluation value found during the search.

Tabu search explores the solution space beyond local optimality. The properties of the search path are memorized, and choices are made to guide the search out of the local optimum and into different regions. Some moves to previously explored areas are prohibited to avoid local optimization. Recent moves are taboo and stored in the tabu list. The tabu list identifies all moves that do not apply to the current solution. The size of the tabu list is an algorithm parameter that needs to be determined.

An initial solution is identified. The neighborhood operator will determine the neighborhood region for the current solution. The next solution, which may not be as good as the current solution, is determined by applying the selection operator to select the best solution in the current neighborhood region. The selected solution must not violate the tabu list. The process is repeated until a specified stopping condition is met.

The flexible flow shop scheduling problem

The problem to be solved is an FFSS problem with nine orders, O_i, $i = 1 \div 9$, and scheduling on four stations, S_j, $j = 1 \div 4$. Each order consists of many products, manufactured in batches at each machine in the stations. Each station has three homogeneous machines. The due time D_i, the processing time P_{ij}, and the batch time B_{ij} for order i, $i = 1 \div 9$, on station j, $j = 1 \div 4$, are estimated in **Table 1**.

The setup times of the machines in station j are estimated in **Table 2**.

The model is set up with variables. TS_{ijk} being the time to start order O_i, $i = 1 \div 9$ on station S_j, $j = 1 \div 4$, machine k, $k = 1 \div 3$; X_{ijk} being the binary variable, whether order O_i, $i = 1 \div 9$ will process on station S_j, $j = 1 \div 4$, machine k, $k = 1 \div 3$ or not; TE_{ijk} being the time to end order O_i, $i = 1 \div 9$ on station S_j, $j = 1 \div 4$, machine k, $k = 1 \div 3$; T_i being the tardiness time of job i. The model of the problem is as follows:

$$T_{\min} = \text{MinT}, T = \Sigma(T_i, i = 1 \div 9)$$

St.

- $TE_{ijk} = TS_{ijk} + (S_j + P_{ij}) * X_{ijk}$, $i = 1 \div 9$, $j = 1 \div 4$, $k = 1 \div 3$
- $T_i = \max(TE_{i4k} - D_i, 0)$, $i = 1 \div 9$, $k = 1 \div 3$
- $\Sigma_k X_{ijk} = 1$, $\Sigma_i X_{ijk} = 1$, $i = 1 \div 9$, $j = 1 \div 4$, $k = 1 \div 3$
- $TS_{ijk} \geq \text{Max}(TS_{i(j-1)k} + B_{i(j-1)}, TE_{(i-1)jk})$, $i = 1 \div 9$, $j = 1 \div 4$, $k = 1 \div 3$

TABLE 1 | Input data for model.

I	D_j (h)	P_{i1} (h)	P_{i2} (h)	P_{i3} (h)	P_{i4} (h)	B_{i1} (h)	B_{i2} (h)	B_{i3} (h)	B_{i4} (h)
1	21	20.04	12.14	0	13.74	1.33	4.83	0	2.5
2	12	6.83	10.62	7.63	1.33	0.92	1.75	1.57	0.67
3	15	0	4.58	1.65	7.04	0	1.17	1.31	0.09
4	11	23.05	7.42	0	0	0.28	0.12	0	0
5	21	0	5.39	9.18	0	0	0.31	0.68	0
6	25	0	2.67	8	0	0	1	3	0
7	18	9	13	14.67	3.67	1.35	1.95	2.2	0.55
8	20	0	3.12	3.63	0	0	1.81	2.1	0
9	18	0	1.17	1.53	0	0	2.33	3.06	0

TABLE 2 | Setup times of machine stations.

j	1	2	3	4
S_j (h)	0.15	0.2	0.15	0.1

- $TS_{ij1} = \max(0, \min(TE_{(i-1),j1}))$, $i = 1 \div 9$, $j = 1 \div 4$
- $TS_{ijk} = \max(\min(TE_{(i-1)jk}), TS_{ij(k-1)} + B_{ij})$, $i = 1 \div 9$, $j = 1 \div 4$, $k = 2 \div 3$
- TS_{ijk}, $TE_{ijk} \geq 0$; $X_{ijk} = \{0,1\}$, $i = 1 \div 9$, $j = 1 \div 4$, $k = 1 \div 3$

The company is currently using the LPT dispatching method. The objective value of T is 46.93 (h).

The GATS model for the Flexible flow shop scheduling problem

The above FFSS problem is a non-linear problem with a solution space of 4*9!, or 1,451,520. The GATS model is used to solve the problem. In the model, GA is used to perform a global search of the solution space, and TS is used to perform a local search to refine the solution found by GA. The GATS procedure is as follows:

Step 1: Initialize the GATS model.
Step 2: Generate the initial population, $\mathbf{P}^{(0)}$. Set k to 0.
Step 3: Generate elite population $\mathbf{P}_E{}^{(k)}$.
Step 4: Generate the genetic population $\mathbf{P}_G{}^{(k)}$.
Step 5: Generate the neighborhood population $\mathbf{P}_N{}^{(k)}$.
Step 6: Generate the next population $\mathbf{P}^{(k+1)}$. Set $k = k + 1$.
Step 7: Check the termination rule. If not, return to Step 3. If yes, finish the loop.
Step 8: Run the algorithm a number of times to choose the best scheduling result.

Step 1: Initialize the GATS model.

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

The method of coding: Each chromosome C is a string of four sub-chromosomes, S_j, $j = 1 \div 4$, corresponding to four stations. The orders are numbered from 1 to 9. Each sub-chromosome is a string of nine genes, G_i, $i = 1 \div 9$, corresponding to nine orders. The sequence of genes in a sub-chromosome represents the sequence of orders scheduled in the corresponding station:

$$C = [S_1, S_2, S_3, S_4]; S_j = (G_{j1}, G_{j2}, G_{j3}, G_{j4}, G_{j5}, G_{j6}, G_{j7}, G_{j8}, G_{j9}), j = 1 \div 4$$

The GA parameters include fitness function, population size, and the parameters of GA operators, including selection, crossover, mutation, and replacement operators. The fitness function F is defined as $F_i = T_{max} - T_i$, where F_i and T_i are the fitness and objective values of chromosome i and T_{max} is the maximum objective value in the population. The crossover method is *POX (precedence operation crossover)*, the mutation method is *REM (reciprocal exchange mutation)*, and the replacement method is acceptance threshold. The model parameters P, P_c, P_m, and K are chosen as follows:

$$P = 9, P_c = 0.8; P_m = 0.2, K = 2$$

The TS parameters include the *neighborhood operator* and the *tabu list*. The neighborhood operator uses the SWAP method to find the neighborhood chromosomes of a chromosome. The *tabu list* contains the chromosomes found in the previous iterations. At the end of each iteration, chromosomes moving into the next population must not be on the *tabu list*.

***The termination rule*:** The best objective value of the population T_{min} does not improve or decrease after 10 consecutive iterations.

Step 2: Generate the initial population $\mathbf{P}^{(0)}$ and set $k = 0$.

The initial population consists of nine chromosomes. There are four chromosomes generated from four heuristic rules: EDD, ERD, SPT, and LPT. The remaining chromosomes R1, …, R5 are randomly generated. The chromosomes in the initial population, arranged in

TABLE 3 | The initial population $\mathbf{P}^{(0)}$.

$\mathbf{P}^{(0)}$	S1	S2	S3	S4	T_i (h)	F_i (h)	P_i
EDD	423798156	423798156	423798156	423798156	9.41	37.52	0.215
ERD	123456789	123456789	123456789	123456789	13.68	33.25	0.191
R2	741295863	172635498	876425913	238915674	15.21	31.72	0.182
SPT	356892714	968354217	149382657	456892731	16.34	30.59	0.176
R1	247195683	512637894	856724913	436215879	30.32	16.61	0.095
R4	546298173	342156879	874256913	236194567	33.87	13.06	0.075
R3	146295873	372651489	372456918	136254897	39.32	7.61	0.044
R5	826495173	649153872	124736985	631294758	43.15	3.78	0.022
LPT	417235689	712453869	756283914	137245689	46.93	0	0

descending order of their objective values with their corresponding objective and fitness values, are shown in Table 3.

$$P^{(0)} = \{EDD, ERD, R2, SPT, R1, R4, R3, R5, LPT\}$$

This step also initializes the tabu list, TL, containing the chromosomes in $\mathbf{P}^{(0)}$.

$$TL = \{EDD, ERD, R2, SPT, R1, R4, R3, R5, LPT\}$$

Step 3: Generate an elite population $\mathbf{P}_E{}^{(k)}$.

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

$$P_i = F_i / \Sigma_{i=1 \div 9}(F_i)$$

The selection probabilities P_i are calculated as shown in Table 3. Random numbers are generated nine times based on P_i; the chromosomes in $\mathbf{P}^{(0)}$ selected into the population $\mathbf{P}_E{}^{(0)}$ are as follows:

$$P_E^{(0)} = \{R2, ERD, SPT, R3, R3, SPT, ERD, R4, EDD\}$$

TABLE 4 | The crossover population P^C.

$\mathbf{P}^C$	S1	S2	S3	S4	T
C1	421356789	421356789	421356789	421356789	28.16
C2	327498156	327498156	327498156	327498156	32.05
C3	426958713	367254819	137245698	326548971	12.84
C4	136589274	983651427	493826517	146589273	21.29

TABLE 5 | The mutation population P^M.

$\mathbf{P}^M$	S1	S2	S3	S4	T
M1	741259863	172635498	876425913	238915674	17.15
M2	356892714	968453217	149382657	456892731	27.03

TABLE 6 | The neighborhood population $P_N{}^{(0)}$.

$\mathbf{P}_N{}^{(0)}$	S1	S2	S3	S4	T	Tabu
N1	423798156	423798156	423798156	423798156	9.41	Yes
N2	123456789	123456789	123456789	123456789	13.68	Yes
N3	172635498	741295863	876425913	238915674	14.11	No
N4	968354217	356892714	149382657	456892731	15.02	No
N5	546298173	342156879	236194587	874256913	25.94	No
N6	247195683	512637894	436215879	856724913	26.71	No
N7	146295873	372456918	372651489	136254897	38.79	No
N8	826495173	124736985	649153872	631294758	39.17	No
N9	712453869	417235689	756283914	137245689	42.13	No
N10	421356789	421356789	421356789	421356789	28.16	No
N11	327498156	327498156	327498156	327498156	32.05	No
N12	367254819	426958713	137245698	326548971	11.25	No
N13	136589274	493826517	983651427	146589273	75.00	No
N14	172635498	741259863	876425913	238915674	33.91	No
N15	356892714	968453217	456892731	149382657	18.55	No

TABLE 7 | The next population $\mathbf{P}^{(1)}$.

$\mathbf{P}^{(1)}$	S1	S2	S3	S4	T
EDD	423798156	423798156	423798156	423798156	9.41
N12	367254819	426958713	137245698	326548971	11.25
ERD	123456789	123456789	123456789	123456789	13.68
N3	172635498	741295863	876425913	238915674	14.11
N4	968354217	356892714	149382657	456892731	15.02
R2	741295863	172635498	876425913	238915674	15.21
SPT	356892714	968354217	149382657	456892731	16.34
R1	247195683	512637894	856724913	436215879	30.32
R4	546298173	342156879	874256913	236194567	33.87

Step 4: Generate the genetic population $\mathbf{P}_G{}^{(k)}$.

This step uses the crossover and mutation operators to generate the genetic population $\mathbf{P}_G{}^{(k)}$ from the elite population $\mathbf{P}_E{}^{(k)}$. The genetic population $\mathbf{P}_G{}^{(k)}$ includes the new chromosome generated from the crossover and

TABLE 8 | The results after 21 iterations.

Iterations	1	2	3	4	5	6	7	8	9	10	11	12	...	21
$T(h)$	9.41	8.89	8.89	6.11	6.11	5.55	5.55	5.12	5.12	3.82	3.82	2.37	...	2.37

TABLE 9 | The results after 10 runs.

Runs	1	2	3	4	5	6	7	8	9	10
$T(h)$	2.37	5.98	4.86	3.71	3.63	5.01	4.12	2.88	3.96	3.82

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

$$P_c = \{EDD, ERD, SPT, R3\}.$$

Each pair of chromosomes in $\mathbf{P}_c$ is selected to cross over by the POX method, resulting in two new chromosomes in population $\mathbf{P}^C$. R2 and ERD are crossed over to each other and generate two children, C1 and C2. SPT and R4 are crossed over to each other and generate two children, C3 and C4. The chromosomes in $\mathbf{P}^C$ and their objective values are shown in **Table 4**.

$$P^C = \{C1, C2, C3, C4\}$$

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

$$P_m = \{R2, SPT\}.$$

Each chromosome in $\mathbf{P}_m$ is selected to mutate by the REM method, resulting in one new chromosome in population $\mathbf{P}^M$. R3 is mutated and generates M1. SPT is mutated and generates M2. The chromosomes in $\mathbf{P}^M$ and their objective values are shown in **Table 5**:

$$P^M = \{M1, M2\}$$

After crossover and mutation, six new chromosomes are created in the population $\mathbf{P}_G{}^{(0)}$. Their chromosomes and objective values are shown in **Table 5**:

$$P_G^{(0)} = \{C1, C2, C3, C4, M1, M2\}$$

Step 5: Generate the neighborhood population $\mathbf{P}_N{}^{(k)}$.

The populations $\mathbf{P}^{(k)}$ and $\mathbf{P}_G{}^{(k)}$ form the union population $\mathbf{P}_U{}^{(k)} = \mathbf{P}^{(k)} \cup \mathbf{P}_G{}^{(k)}$. This step uses the neighborhood operator to generate the neighborhood population $\mathbf{P}_N{}^{(k)}$ from the union population $\mathbf{P}_U{}^{(k)}$. For the first iteration, $\mathbf{P}_U{}^{(0)}$ includes 15 chromosomes, of which 9 are in $\mathbf{P}^{(0)}$ and 6 are in $\mathbf{P}_G{}^{(0)}$:

$$P_U^{(0)} = P^{(0)} \cup P_G^{(0)} = \{EDD, ERD, R2, SPT, R1, R4, R3, R5, LPT, C1, C2, C3, C4, M1, M2\}$$

The neighborhood operator swaps adjacent sub-chromosomes in each chromosome to generate neighborhood chromosomes. Each chromosome in $\mathbf{P}_U{}^{(0)}$ will have three neighborhood chromosomes. The best chromosome will be selected to move into $\mathbf{P}_N{}^{(0)}$ to go forward. Population $\mathbf{P}_N{}^{(0)}$ includes the best neighborhood chromosomes, as shown in **Table 6**:

$$P_N^{(0)} = \{N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, N11, N12, N13, N14, N15\}$$

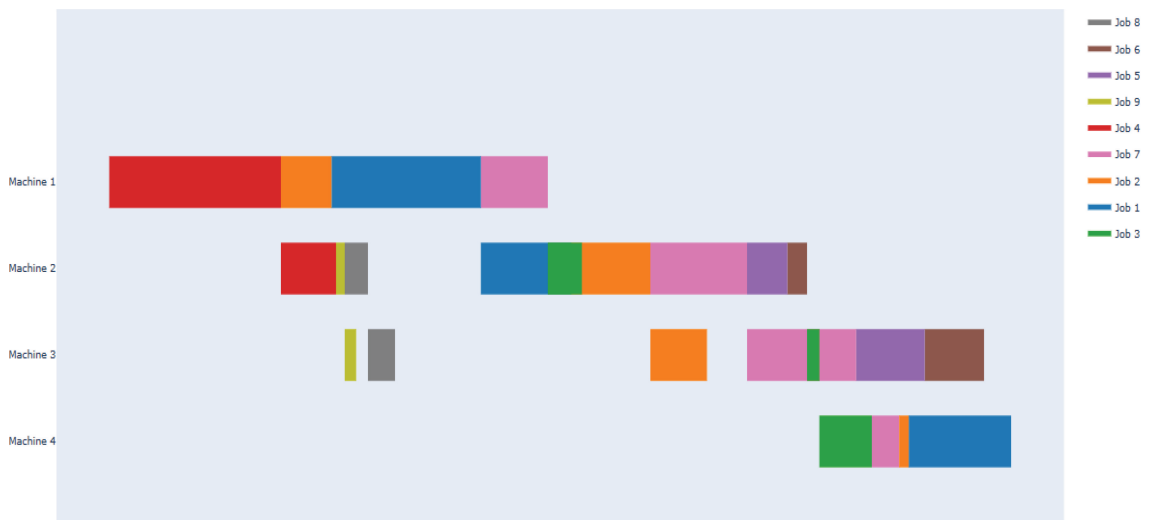

FIGURE 1 | The Gantt chart by GATS model.

TABLE 10 | The binary levels of the input factors.

Factors	Levels	
P	8	12
P_c	0.6	0.8
P_m	0.2	0.3

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	7	116.960	16.7086	4.36	0.002
Linear	3	98.188	32.7292	8.53	0.000
P	1	30.102	30.1023	7.85	0.009
Pc	1	67.029	67.0292	17.48	0.000
Pm	1	1.056	1.0562	0.28	0.603
2-Way Interactions	3	5.708	1.9026	0.50	0.688
P*Pc	1	0.299	0.2993	0.08	0.782
P*Pm	1	3.283	3.2833	0.86	0.362
Pc*Pm	1	2.125	2.1252	0.55	0.462
3-Way Interactions	1	13.064	13.0645	3.41	0.074
P*Pc*Pm	1	13.064	13.0645	3.41	0.074
Error	32	122.729	3.8353		
Total	39	239.689			

FIGURE 2 | The ANOVA of the binary experiment.

In $\mathbf{P}_N{}^{(0)}$, there are two chromosomes, N1 and N2, in the tabu list; the rest are not.

Step 6: Generate the next population $\mathbf{P}^{(k+1)}$.

The step uses the replacement operator to generate the next population $\mathbf{P}^{(k+1)}$ from the populations $\mathbf{P}^{(k)}$ **and** $\mathbf{P}_N{}^{(k)}$. The chromosomes from $\mathbf{P}_N{}^{(k)}$ will be added to the current population $\mathbf{P}^{(k)}$ to make the next population $\mathbf{P}^{(k+1)}$, if they are not in the current tabu list and their objective values exceed an acceptable threshold, defined by the objective value of the threshold chromosome. The threshold chromosome is a chromosome in $\mathbf{P}^{(k)}$ with position n defined by population size $\boldsymbol{P}$ and threshold parameter $\boldsymbol{k}$ as follows:

$$n = P/k$$

With a population size of nine and a threshold parameter of two chosen, the threshold chromosome is the fourth chromosome of $\mathbf{P}^{(k)}$ in the top-down ranking. In order to keep the next population size constant, chromosomes in $\mathbf{P}^{(k)}$ with the lowest values are removed from the next population. This step also updates the tabu list by adding the chromosomes moved into the next population $\mathbf{P}^{(k+1)}$.

In this iteration, the threshold chromosome is SPT, with a threshold value of 16.34. Chromosomes N3, N4, and N12 move into, and R3, R5, and LPT move out of the population. The chromosomes in the next population, $\mathbf{P}^{(1)}$, are shown in Table 7:

$$P^{(1)} = \{EDD, N12, ERD, N3, N4, R2, SPT, R1, R4\}$$

The tabu list is updated after iteration 1 as follows:

$$TL = \{EDD, ERD, R2, SPT, R1, R4, R3, R5, LPT, N3, N4, N12\}$$

Step 7: Check the termination rule.

After iteration 1, EDD is again the best chromosome with the best objective value of 9.41, appearing twice. The termination rule is not satisfied, so iteration 2 is executed. The results after 14 iterations are shown in Table 8.

From the 12th iteration to the 21st iteration, the best objective value remains the same, the termination rule is satisfied, and the algorithm ends. The scheduling result for this run is as follows:

$$C = (413296857, 413297856, 142379856, 472319856),$$
$$T = 2.37(h)$$

Step 8: 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 9.

The best scheduling result is found on the first run. The scheduling sequence C and the objective value T are as follows:

$$C = (423981756, 498132756, 982713546, 439782156),$$
$$T = 2.37(h).$$

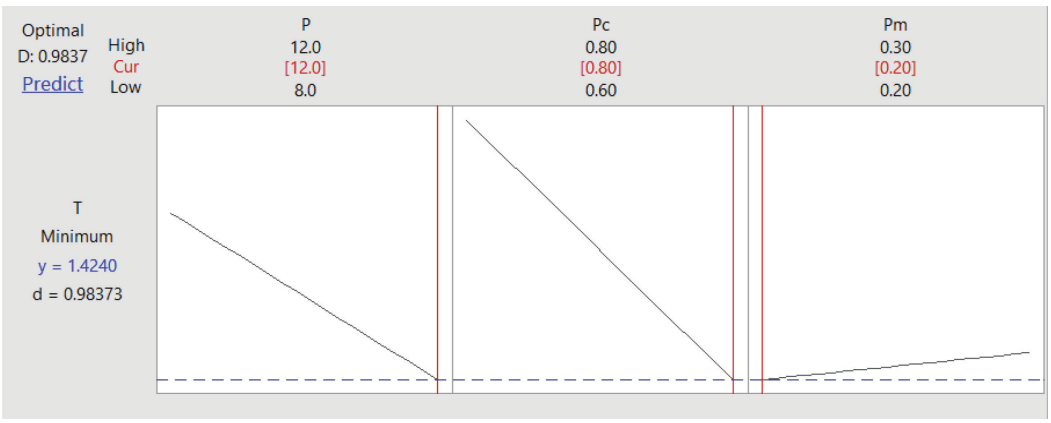

FIGURE 3 | The values of parameters to give the minimum objective value.

TABLE 11 | The multilevels of the input factors.

Factors	Levels				
P	11	12	13	14	15
P_c	0.5	0.6	0.7	0.8	0.9

Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	24	210.27	8.761	2.27	0.004
Linear	8	132.09	16.511	4.28	0.000
P	4	95.37	23.844	6.18	0.000
Pc	4	36.72	9.179	2.38	0.059
2-Way Interactions	16	78.18	4.886	1.27	0.241
P*Pc	16	78.18	4.886	1.27	0.241
Error	75	289.27	3.857		
Total	99	499.54			

FIGURE 4 | The ANOVA of the multilevels experiment.

The Gantt chart for the GATS model is shown in **Figure 1**.

The objective value according to the GATS model (2.37h) is less than the objective value according to the LPT model currently used (46.93h).

The GATS with DOE to solve the FFSS problem

The parameters of the above GATS model were selected empirically, so the results are not very good. Experimental design is used to determine the model parameters. Firstly, a binary experiment is conducted with the model parameters P, P_c, and P_m chosen as the input factors and the total tardiness time T chosen as the output factor. The levels of the input factors are chosen as in **Table 10**.

Each combination of input factors is tested five times. The total number of experiments is 40. The experimental results are collected. From the data, Analysis of variance is performed as in **Figure 2**.

From the above ANOVA table, with α of 0.05, the affected factors are factors P and P_c. The values of parameters to give the minimum objective value are shown in **Figure 3**.

$$P = 12, P_c = 0.8, P_m = 0.2$$

A further experiment is conducted with the two affected factors chosen as the input factors and the total tardiness time T chosen as the output factor. The mutation probability P_m is kept at 0.2. The levels of the input factors are chosen as in **Table 11**.

There are 25 combinations of input factors. Each combination is experimented with four times. The total number of experiments is 100. The experimental results are collected. From the data, analysis of variance is performed as in **Figure 4**.

From the ANOVA shown in **Figure 4**, with α of 0.05, the most affected factor is population size P. The values of parameters to give the minimum objective value are shown in **Figure 5**.

$$P = 15, P_c = 0.9$$

The GATS model with DOE is run 10 times with the parameters defined by the experiment as follows:

$$P = 15, P_c = 0.9, P_m = 0.2$$

The results of the GATS model with DOE are shown in **Table 12**.

TABLE 12 | The results of the GATS with DOE after 10 runs.

Runs	1	2	3	4	5	6	7	8	9	10
$T(h)$	0.86	0.04	0.65	5.57	3.18	1.18	0.04	0	2.45	3.82

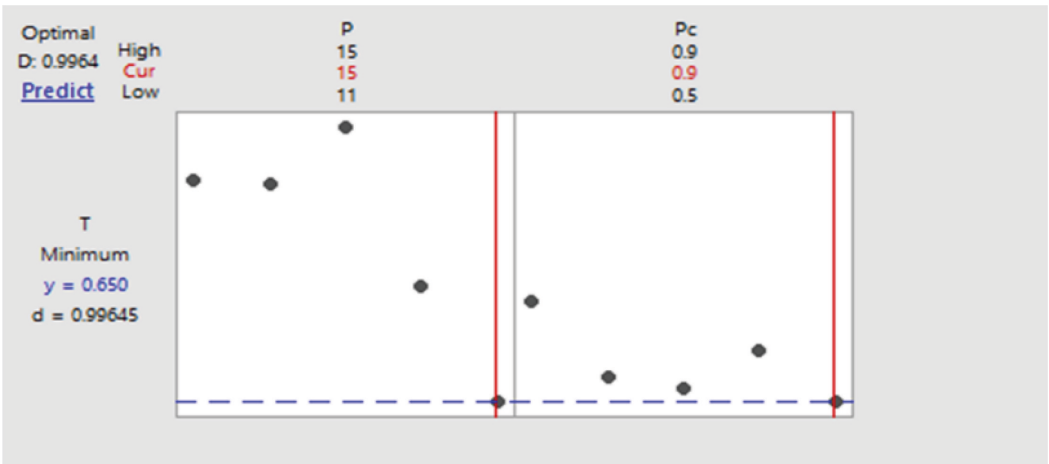

FIGURE 5 | The values of parameters must give the minimum objective value.

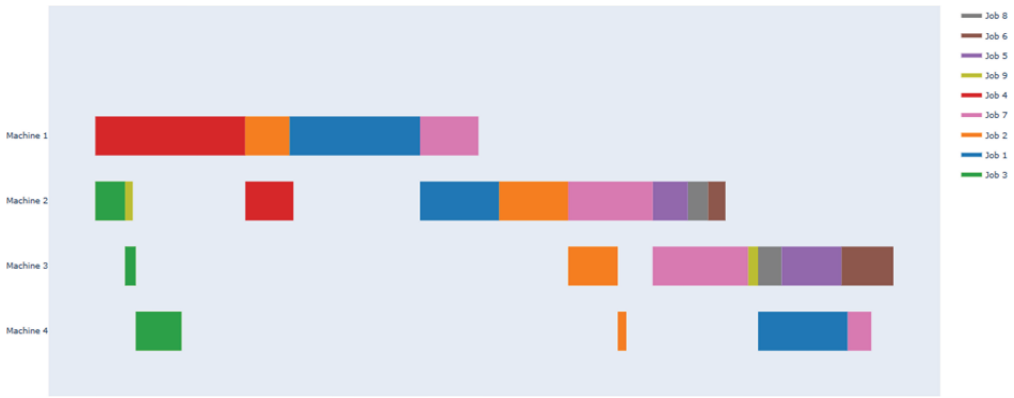

FIGURE 6 | The Gantt chart for the GATS model with DOE.

The best scheduling result is found on the 8th run. The scheduling sequence C and the objective value T are as follows:

$$\mathrm{C} = (394218756, 394127586, 342791856, 432198756),$$
$$\mathrm{T} = 0(\mathrm{h}).$$

The Gantt chart for the GATS model with DOE is shown in **Figure 6**.

The objective value according to the GATS model with DOE ($0h$) is less than the objective value according to the GATS model without DOE ($2.37h$).

Conclusion

The GATS model has been used to solve the FFSS problem with nine orders on four stations and three homogeneous parallel machines. In the model, GA is used as the platform to perform a global search of the solution space, and TS is used to perform a local search to refine the solution found by GA. The results show that the GATS model gives a better objective value of tardiness time than the heuristic LPT method being used.

Design of experiments has been used to optimize the parameters of the GATS model. The results show that the GATS model with DOE gives a better objective value of tardiness time than the GATS model without DOE. The GATS models without and with DOE have been used to solve the FFSS problem with small sizes. Future research will use the models to solve real-world problems.

Author contributions

PN is the thesis advisor for TL. PN has developed the research models for the thesis. TL has collected the data and written a program to run the algorithms based on the models. PN has composed the research article based on the thesis. Both authors contributed to the article and approved the submitted version.

Acknowledgments

We extend our heartfelt appreciation to everyone who has contributed to the completion of this research article, especially our families, the HCMC University of Technology, and the scientific community for their invaluable support. We endorse the fact that your contributions have been instrumental in the successful completion of this work.

References

1. Umam M, Mustafid M, Suryono S. A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. *J King Saud Univ Comput Inform Sci.* (2022) 34:7459–67.
2. Etiler O, Toklu B, Atak M, Wilson J. A genetic algorithm for flow shop scheduling problems. *J Operat Res Soc.* (2004) 55:830–5.
3. Kim S, Choi H, Lee D. Tabu search heuristics for parallel machine scheduling with sequence-dependent setup and ready times. *Proceedings of the International Conference on Computational Science and its Applications, vol Part III, Glasgow, UK, May 8-11, 2006*. Glasgow: (2006). p. 728–37.
4. Helal M, Rabadi G, Al-Salem A. A tabu search algorithm to minimize the makespan for the unrelated parallel machines scheduling problem with setup times. *Int J Operat Res.* (2006) 3:182–92.
5. Bilge Ü, Kirac F, Kurtulan M, Pekgun P. A tabu search algorithm for parallel machine total tardiness problem. *Comput Operat Res.* (2002) 31:397–414.
6. Kolahan F, Tavakoli A, Tajdin B, Hosayni M. Analysis of neighborhood generation and move selection strategies on the performance of Tabu Search. *Proceedings of the 6th WSEAS International Conference on Applied Computer Science, Tenerife, Canary Islands, Spain, December 16-18, 2006*. Tenerife: (2006).
7. Gupta J, Palanimuthu N, Chen C-L. Designing a tabu search algorithm for the two-stage flow shop problem with secondary criterion. *Product Plann Control Manag Operat.* (1999) 10:251–65.
8. Burduk A, Musiał K, Kochańska J, Górnicka D, Stetsenko A. Tabu Search and genetic algorithm for production process scheduling problem. *Logforum.* (2019) 15:181–9.