Hindawi
Scientific Programming
Volume 2018, Article ID 2615096, 8 pages
https://doi.org/10.1155/2018/2615096

Research Article

Reentrant Flow Shop Scheduling considering Multiresource Qualification Matching

Feng Chu,[1] Ming Liu,[2] Xin Liu,[2] Chengbin Chu,[2,3] and Juan Jiang[4]

[1]*Management Engineering Research Center, Xihua University, Chengdu 610039, China*
[2]*School of Economics & Management, Tongji University, Shanghai 200092, China*
[3]*Systems Engineering Department, Université Paris-Est, ESIEE Paris, Noisy-le-Grand Cedex, France*
[4]*Glorious Sun School of Business & Management, Donghua University, Shanghai 200051, China*

Correspondence should be addressed to Xin Liu; liuxin9735@126.com

Received 7 March 2018; Accepted 6 June 2018; Published 9 July 2018

Academic Editor: José E. Labra

With the development of technology and industry, new research issues keep emerging in the field of shop scheduling. Most of the existing research assumes that one job visits each machine only once or ignores the multiple resources in production activities, especially the operators with skill qualifications. In this paper, we consider a reentrant flow shop scheduling problem with multiresource considering qualification matching. The objective of the problem is to minimize the total number of tardy jobs. A mixed integer programming (MIP) model is formulated. Two heuristics, namely, the hill climbing algorithm and the adapted genetic algorithm (GA), are then developed to efficiently solve the problem. Numerical experiments on 30 randomly generated instances are conducted to evaluate the performance of proposed MIP formulation and heuristics.

1. Introduction

Flow shop scheduling problem has been widely studied since it is first proposed ([1–14]; Che and Chu, 2005; Desprez et al., 2006; Kuo and Yang, 2010; Xu and Yin, 2011, etc.). With the development of economy and technology, new research issues keep emerging in this field these years [15–19]. The traditional flow shop scheduling problem assumes that the jobs only visit each machine one time. However, this assumption is not always consistent with the actual production activities. Indeed, there are situations where one job can visit a certain machine twice or more times, i.e., reentrance, such as the production of nuclear materials and aircraft manufacturing process. Reentrant flow shop scheduling problem is first proposed by Graves [1], which is illustrated in Figure 1: there are three procedures for each job, and each job is processed from machine m_1 to m_2 and then back to m_1.

Most existing works addressing flow shop scheduling problem with resource requirement only consider machines and raw materials. However, the impact of many other kinds of resources on the solution is not negligible, including the operators with different abilities. For example, doctors can be considered as expensive and rare surgical resources in the surgical scheduling problem, and drivers can be also regarded as resources in the vehicle scheduling problem, etc. In this paper, reentrance flow shop scheduling considering multiresource with personal qualification matching is investigated. The main contributions of this paper mainly include the following:

(1) We consider a reentrant flow shop scheduling problem, taking personal qualification matching into consideration.

(2) A new mixed integer programming (MIP) formulation is proposed.

(3) Two heuristics are developed to efficiently solve the problem, i.e., the hill climbing algorithm and the adapted genetic algorithm (GA).

(4) Numerical experiments are conducted to evaluate the performance of our developed heuristics. Computational results show that hill climbing algorithm is more time-saving, and GA performs better in terms of the solution quality.

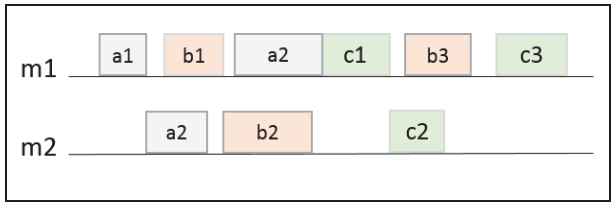

Figure 1: Reentrance.

The rest of this paper is structured as follows. Section 2 reviews the literature on reentrant flow shop scheduling problem and shop scheduling with multiresource. Section 3 describes the problem and proposes a mixed integer programming (MIP) formulation. In Section 4, a hill climbing algorithm and an adapted GA are developed. Section 5 reports computational results. Section 6 summarizes this work and states future research directions.

2. Literature Review

There are a significant amount of researches addressing the flow shop scheduling problem. Only few researches have been conducted to deal with the reentrance of jobs (e.g., [20, 21]; Kim, 2005). Besides, existing works considering multiple resources including the personnel are also very rare.

2.1. Reentrant Scheduling. Graves [1] first proposed the reentrant flow shop scheduling problem and develop a heuristic algorithm to solve the problem. Wang et al. [5] study a chain-reentrant shop scheduling problem, in which every job goes first to the primary machine, then to a series of machines, and finally back to the primary machine. They focus on the two-machine case and prove some properties that can identify a specific class of optimal schedules. Based on the properties, they develop an approximation algorithm and a branch-and-bound algorithm.

Chen (2006) addresses the reentrant permutation flow shop scheduling problem, where every job must be processed on machines in the same order, $M_1, M_2, \ldots, M_m$, $M_1, M_2, \ldots, M_m$ and $M_1, M_2, \ldots, M_m$. They present a branch-and-bound algorithm to minimize the makespan. Chen et al. (2008) make a further study on the problem. They propose a hybrid tabu search and a hybrid genetic algorithm. Choi and Kim [22] address a two-machine reentrant flow shop scheduling problem, in which jobs have to be processed twice in the production system. They propose some dominance properties and lower bounds. Then based on the properties, they develop a branch-and-bound algorithm and a heuristic algorithm.

Chu et al. (2008) investigate a reentrant shop problem, which can be considered as a special case of the problem studied by Wang et al. [5]. They propose an optimal schedule to minimize the makespan. As for minimizing the total flow time, they decompose the problem into several subproblems which are solved by their proposed heuristics. Jing et al. [23] develop a heuristic algorithm for the reentrant flow shop scheduling problem to minimize total completion time. An effective k-insertion technique is used in the iterative process.

Desprez et al. [24] address a real-world industrial problem, which is a hybrid flow shop with reentrance. The purpose is to minimize the total weighted number of tardy jobs. A genetic algorithm is developed in order to deal with large size problems. Chakhlevitch and Glass [25] proposed a special two-stage hybrid reentrant flow shop scheduling problem. The objective is to minimize the makespan. The authors proved the problem is NP-hard, then they develop an effective heuristic algorithm by analyzing the characteristics of the problem.

Huang et al. [26] develop a particle swarm optimization algorithm to solve the reentrant two-stage multiprocessor flow shop scheduling problem. Xu et al. [27] present a memetic algorithm for the reentrant permutation flow shop scheduling problem to minimize the makespan. Sangsawang et al. [28] develop a hybrid genetic algorithm for the two-stage reentrant flexible flow shop with blocking constraint to minimize the makespan. Zhou et al. [29] study a reentrant flow shop scheduling problem with inspection and repair operations. And they propose a mathematical model and a hybrid differential evolution algorithm to minimize total weighted completion time.

Shop scheduling with multiresource has been studied in some research. However, researches focusing on shop scheduling problem with personal qualification matching are very rare.

2.2. Scheduling Considering Multiple Resources. Scheduling problem considering multiple resources has been investigated by many researchers. However, most existing works either ignore the personnel or assume they are identical and can be replaced with each other.

Dauzère-Pérès and Roux [30] first propose a job shop scheduling problem with multiresources to minimize the makespan. In this problem, the authors assume that every job needs all kinds of resources and operators are not considered. Dauzère-Pérès and Pavageau [31] extend the study by allowing resources to be released before completing in one procedure and considering that incompatible resources cannot be chosen by one procedure at the same time.

Artigues et al. [32] study an on-line scheduling in a job shop environment with multiresource requirements and setup times. They present a Petri net model for the problem. Artigues and Roubellat [33] present a polynomial insertion algorithm to minimize the maximum lateness.

Wang and Wu [34] address a multiperiod, multiproduct, and multiresource production-scheduling problem. The authors establish a mixed integer programming (MIP) formulation and propose a two-phase approach to solve the problem.

Rajkumar et al. [35] propose a Greedy Randomised Adaptive Search Procedures algorithm for a flexible job shop scheduling problem. And Gao et al. (2016) propose an algorithm named the shuffled multiswarm micromigrating birds optimization for a multi-resource-constrained flexible job shop scheduling problem.

This paper extends the literature on reentrant flow shop scheduling by considering multiresource, which contains the operators with different skill qualification.

3. Problem Description and Formulation

In this section, we first describe the problem and then propose a new mixed integer programming formulation.

3.1. Problem Description. In the deterministic problem, there are a set of jobs should be processed, i.e., $N = \{1, 2, \ldots, n\}$, and a set of machines, i.e., $M = \{1, 2, \ldots, m\}$. All jobs have to be processed in all procedures $H = \{1, 2, \ldots, |H|\}$ and following the same route; i.e., they have to be processed in procedure 1, then in procedure 2, and so on. Some procedures of jobs are processed on the same machine, i.e., reentrance.

The processing task is defined as one procedure of one job. We address the reentrance by considering that there should not be more than one processing task on one machine at a time. Multiple resources considered include raw materials or operators. Each processing task requires some raw materials, and the number of available raw materials is limited. Besides, some certain skill qualifications are also required by each processing task. Only operators processing the skill qualifications required can be assigned to a processing task, i.e., qualification matching. Moreover, one operator cannot assigned to two or more processing tasks at a time.

Each job should be processed after its own release time. It is assumed that the due dates of jobs are fixed and known in advance. The objective of the problem is to find a set of sequences and operators assignment of jobs in all procedures, in order to minimize the number of tardy jobs. To formally state the problem, a mixed integer programming (MIP) formulation is proposed in the following.

3.2. Mixed Integer Programming (MIP) Formulation. In the following, we give the definitions of parameters and decision variables. Then a new MIP model is formulated.

Indices

(i) i, j: indices of jobs
(ii) r: index of resources
(iii) h: index of procedures
(iv) t: index of time
(v) q: index of qualifications
(vi) k: index of machines

Parameters

(i) N: set of jobs, $N = \{1, \ldots, n\}$
(ii) M: set of machines, $M = \{1, \ldots, m\}$
(iii) H: set of procedures
(iv) Q: set of skill qualifications
(v) P: set of procedures, and $P = \{1, 2, \ldots, |P|\}$
(vi) P_k: set of procedures on machine k
(vii) R_1: set of operators
(viii) R_2: set of raw materials
(ix) r_i: the release time of job i
(x) α_{rq}: a binary parameter, equal to 1 if resource $r \in R_1$ possesses the qualification q
(xi) β_{ih}^{q}: a binary parameter, equal to 1 if qualification q is required by job i in procedure h
(xii) d_i: the due date of job i
(xiii) p_{hi} the processing time of job i in procedure h
(xiv) a_{hi}^{r}: the amount of raw materials required by job i in procedure h, $r \in R_2$
(xv) π_r: the number of available raw materials $r \in R_2$
(xvi) L: a large enough number

Variables

(i) x_{ij}^{h}: binary variable, equal to 1 if job i is processed before job j in procedure h, 0 otherwise
(ii) z_{hi}^{t}: binary variable, equal to 1 if job i is being processed in procedure h at time t, 0 otherwise
(iii) y_{hi}^{rq}: binary variable, equal to 1 if resource $r \in R1$ is assigned to satisfy the qualification q required by job i being processed in procedure h, 0 otherwise
(iv) s_{hi}: nonnegative integer, start time of job i in procedure h
(v) c_{hi}: nonnegative integer, completion time of job i in procedure h
(vi) u_i: binary variable, equal to 1 if job i is a tardy job, 0 otherwise
(vii) γ_{hi}^{rqt}: binary variable, equal to 1 if operator r is in charge of the need of qualification q of job i in procedure h at time t, 0 otherwise

*The **MIP** Formulation*

$$\min \quad \sum_{i \in N} u_i \tag{1}$$

$$\text{s.t.} \quad x_{ij}^{h} + x_{ji}^{h} = 1, \quad \forall h \in P,\ i, j \in N,\ i \neq j \tag{2}$$

$$c_{hi} = s_{hi} + p_{hi}, \quad \forall h \in P,\ i \in N \tag{3}$$

$$s_{hi} \geq c_{h-1,i}, \quad \forall h \in \frac{P}{\{1\}},\ i \in N \tag{4}$$

$$s_{hj} + \left(1 - x_{ij}^{h}\right) L \geq c_{hi}, \quad \forall h \in P,\ i, j \in N,\ i \neq j \tag{5}$$

$$s_{hi} \geq r_i, \quad \forall h \in P,\ i \in N \tag{6}$$

$$t - c_{hi} \leq \left(1 - z_{hi}^{t}\right) L, \quad \forall h \in P,\ i \in N,\ t \in T \tag{7}$$

$$s_{hi} - t \leq \left(1 - z_{hi}^{t}\right) L, \quad \forall h \in P,\ i \in N,\ t \in T \tag{8}$$

$$\sum_{t\in T} z_{hi}^{t} = p_{hi}, \quad \forall h \in P,\ i \in N \tag{9}$$

$$y_{hi}^{rq} \le \alpha_{rq}, \quad \forall h \in P,\ i \in N,\ r \in R_1,\ q \in Q \tag{10}$$

$$y_{hi}^{rq} \le \beta_{hi}^{q}, \quad \forall h \in P,\ i \in N,\ r \in R_1,\ q \in Q \tag{11}$$

$$\sum_{r\in R_1} y_{hi}^{rq} = \beta_{hi}^{q}, \quad \forall h \in P,\ i \in N,\ q \in Q \tag{12}$$

$$\sum_{h\in P}\sum_{i\in N}\sum_{q\in Q} y_{hi}^{rq} z_{hi}^{t} \le 1, \quad \forall r \in R_1,\ t \in T \tag{13}$$

$$\gamma_{hi}^{rqt} \ge y_{hi}^{rq} + z_{hi}^{t} - 1, \quad \forall h \in P,\ i \in N,\ q \in Q,\ r \in R_1,\ t \in T \tag{14}$$

$$\gamma_{hi}^{rqt} \le y_{hi}^{rq}, \quad \forall h \in P,\ i \in N,\ q \in Q,\ r \in R_1,\ t \in T \tag{15}$$

$$\gamma_{hi}^{rqt} \le z_{hi}^{t}, \quad \forall h \in P,\ i \in N,\ q \in Q,\ r \in R_1,\ t \in T \tag{16}$$

$$\sum_{h\in P}\sum_{i\in N} \gamma_{hi}^{rqt} \le 1, \quad \forall r \in R_1,\ t \in T \tag{17}$$

$$\sum_{i\in N}\sum_{h\in P} z_{hi}^{t} a_{hi}^{r} \le \pi_r, \quad \forall r \in R_2,\ t \in T \tag{18}$$

$$\sum_{i\in N}\sum_{h\in p_k} z_{ki}^{t} \le 1, \quad \forall t \in T,\ k \in M \tag{19}$$

$$c_{hi} - d_i \le u_i L, \quad \forall i \in N,\ h = |P| \tag{20}$$

$$x_{ij}^{h} \in \{0,1\}, \quad \forall h \in P,\ i,j \in N,\ i \ne j \tag{21}$$

$$z_{hi}^{t} \in \{0,1\}, \quad \forall h \in P,\ i \in N,\ t \in T \tag{22}$$

$$y_{hi}^{rq} \in \{0,1\}, \quad \forall h \in P,\ i \in N,\ q \in Q,\ r \in R_1 \tag{23}$$

$$\gamma_{hi}^{rqt} \in \{0,1\}, \quad \forall h \in P,\ i \in N,\ q \in Q,\ r \in R_1,\ t \in T \tag{24}$$

$$u_i \in \{0,1\}, \quad \forall i \in N \tag{25}$$

$$s_{hi}, c_{hi} \ge 0, \quad \forall i \in N,\ h \in P \tag{26}$$

The objective function (1) aims at minimizing the number of tardy jobs. Constraint (2) ensures that job i and job j are processed in procedure h, and job i is either processed before job j or after it. Constraint (3) ensures that a job must be processed in a procedure without interruption. Constraints (4)-(6) guarantee that the start time of a job i must be after the previous procedure and its release time and the completion time of jobs prior to it. And constraints (7)-(9) ensure that z_{hi}^{t} is equal to 1 if job i is being processed in procedure h at time t. Constraints (10)-(12) ensure the relationship between the qualities needed by a job and those of an operator. The objective of constraints (14)-(16) is to make the formulation $y_{hi}^{rq} z_{hi}^{t}$ linearized. Constraint (17) ensures that an operator cannot be assigned to more than one processing activity at one time t. Constraint (18) guarantees that the amount of resources needed by the jobs being processed at time t cannot exceed that of available resources. Constraint (19) ensures that there cannot be more than one job being processed on a machine at a time. Constraint (20) gives the definition of tardy jobs. And constraints (21)-(26) give the ranges of decision variables.

As we can see from the computational results reported in Section 5, solving the MIP formulation by calling CPLEX is very time-consuming. Therefore, we develop heuristics to efficiently solve the problem.

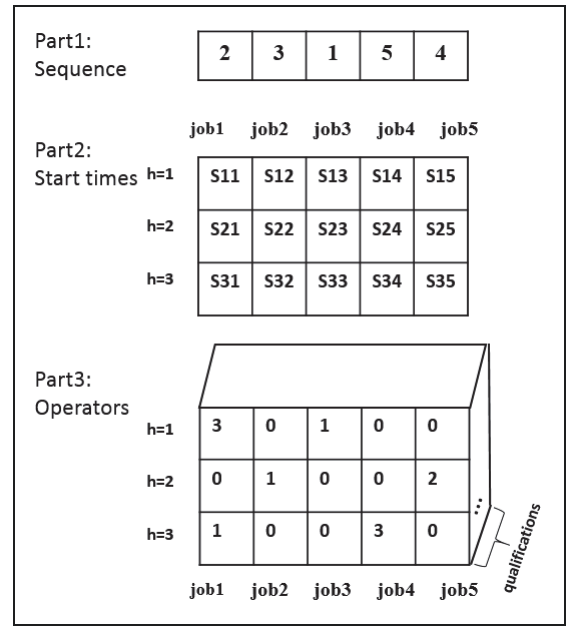

Figure 2: Coding.

4. Solution Approaches

In this section, two heuristics to solve the problem, i.e., a hill climbing algorithm and an adapted genetic algorithm (GA), are developed.

4.1. Coding and Initialization. Solutions should be transformed into individuals that can be operated by the algorithms. For our problem, a solution is composed of three decision parts: (1) the first part is the sequence of jobs in each procedure, (2) the second part is the start time of each in each procedure, and (3) the third part is the operator assignment to each job in each procedure (see Figure 2).

Figure 2 illustrates the coding method of both heuristics. It is assumed that there are 5 jobs to be processed in 3

Require: *Parameters for proposed problem*
Ensure: *Solution*
(1) $MAXITR = 1000$;
(2) $bestStr = Initialize()$;
(3) $k = 1$;
(4) **while** $k \leq MAXITR$ **do**
(5) $bestF = evaluate\_objective(bestStr)$;
(6) $new = string\_operator(bestStr)$;
(7) $newF = evaluate\_objective(newStr)$;
(8) **if** $newF \leq bestF$ **then**
(9) $bestStr = newStr$;
(10) **end if**
(11) $k = k + 1$;
(12) **end while**

Algorithm 1: Hill climbing algorithm.

procedures, and procedure 1 and procedure 3 are on machine 1. The first part of individual is a vector with length of N, and it can be obtained that sequence of jobs processing in each procedure is $\{2, 3, 1, 5, 4\}$. The second part of individual is a $P \times N$ matrix, which gives the information on start times of jobs in all procedures. Besides, the third part of the individual is a $P \times N \times Q$ matrix, which shows the operator assigned to each job in each procedure. For example, operator 3 is assigned to satisfy qualification 1 required by job 1 in procedure 1.

For the generalization of an initial solution, first we generate a vector including random permutation of N, i.e., the first part. Then the start times can be calculated by arranging the processing activities as close as possible. Operator satisfying each qualification required by each processing activity is randomly chosen from those processing the qualification. Then based on the start times calculated and operator assignment, we rearrange the start times: for two processing activities, if there is overlap in time, and (i) one operator is assigned to both processing activities, or (ii) the summation of raw materials required by them exceeds the available materials, then the processing activity with larger start time will be arranged after the completion time of the other one. The rearrangement is repeated N times, and an initial solution is obtained.

4.2. Hill Climbing Algorithm. Hill climbing algorithm is a local search approach, the basic idea of which is to find a string with better solution to replace the existing one. As shown in Algorithm 1, hill climbing starts with an initial solution, denoted as *BestStr*. During each iteration, new solution, i.e., *NewStr*, is generated by mutating *BestStr*. If the solution of *NewStr* is better than that of *BestStr*, then *BestStr* will be replaced with *NewStr*. Solution is obtained when stopping criteria are reached.

The procedure *string_operator* includes two steps: (i) randomly select two jobs and exchange their positions in the first part of existing solution and (ii) randomly assign operators with the required qualification to processing activities. Then start times are rearranged in the method detailed above. Then a new solution is obtained. For the procedure *evaluate_objective*, except for calculating the total number of tardy jobs, feasibility check is conducted by adding penalty to the objective.

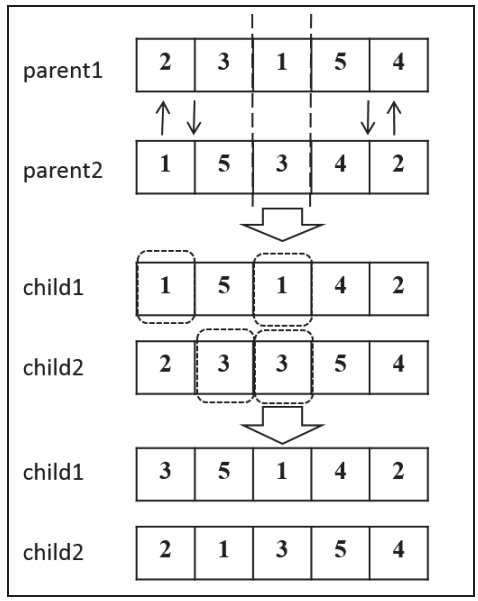

Figure 3: Crossover.

4.3. Genetic Algorithm. Genetic algorithm (GA) is first introduced by Holland [36] and based on the biological reproduction rules. GA starts with a set of initial solutions. During each iteration, offspring individuals are produced by current solutions by conducting genetic operators, i.e., crossover and mutation. Then population combining current solutions and offspring solutions is renewed. The algorithm stops when a stopping criterion is reached.

4.3.1. Crossover and Mutation. There are two common crossover operators in scheduling problem, namely, one-point crossover and multipoint crossover. In this paper, we adopt two-point crossover for our problem. For two-point crossover operator, two parent solutions are selected randomly. As shown in Figure 3, we randomly select two numbers pA and pB in N, i.e., 2 and 3. Then all genes between pA and pB are copied from parent 1 to child 1, the remaining genes of child 1 are copied from parent 2. Child 2 is produced in the same method. Then replace the redundant genes with missing genes. Then the operators are reassigned to processing activities, and the start times are calculated in the same way stated above.

For the mutation operator, solutions are mutated in the same way as the *string_operator* in hill climbing.

5. Computational Results

In this section, the performance of our proposed formulation and two heuristics are evaluated by 30 randomly generated instances. Formulation and proposed heuristics are coded

Table 1: Parameters for GA.

Parameter	Value (GA)
Population size (*pop*)	50
Generation number (*gen*)	20
Crossover probability	0.8
Mutation probability	0.6

in MATLAB_2014b. CPLEX 12.6 solver is called to solve the formulation. All numerical experiments are conducted on a personal computer with Core I5 and 3.30GHz processor and 8GB RAM under Windows 7 operating system. The computational times of formulation and two heuristics are limited to 3600 seconds.

A preliminary analysis is conducted to fine-tune the parameters of proposed two heuristics. For the hill climbing algorithm, the maximum number of iterations is set to be 500. For the genetic algorithm (GA), parameters are presented in Table 1. Population size and generation number are set to be 50 and 20, respectively. The crossover probability and the mutation probability are set to be 0.8 and 0.6, respectively.

5.1. Data Generation. The tested data are generated following the way in Liu et al. (2015) and Chen (2009). The processing times of jobs are randomly generated from a discrete Uniform distribution over $[1, 10]$. The due dates of jobs are randomly generated from discrete Uniform distribution on interval $[(1 - C - Q/2) \cdot \sum_{i=1}^{n} p_i, (1 - C + Q/2) \cdot \sum_{i=1}^{n} p_i]$, in which Q and C imply the due date range and the factor of tardiness, respectively. During numerical experiments, we consider the first combination of C and Q described in Liu et al. (2015), i.e., $C = 0.2, Q = 0.2$. The range of t is set to be $[1, \sum_{i \in N} \sum h \in Pp_{hi}]$. It is assumed that the number of raw material types, i.e., $|R_2|$, is 4, and the number of raw material in each type, i.e., π_r is randomly generated a discrete Uniform distribution over $[20, 40]$. The number of raw material required by each job in each procedure is randomly generated from a discrete Uniform distribution on $[0, \lceil \pi_r / |H| \rceil]$. The total number of skill qualification types, i.e., $|S|$, is 4, and the skill qualifications processed by each operator are randomly generated, ensuring that one operator processes at least one skill qualification. In numerical experiments, we assume that the schedule is permutation and the first procedure and the last procedure of jobs are on machine 1.

5.2. Results and Discussion. Computational results on 30 randomly generated instances are shown in Table 1 in the following. For each instance, we run formulation and each heuristic 30 times and obtain the average value.

In Table 2, Obj, CT, HCA denote objective value, computational time, and hill climbing algorithm, respectively. The optimal solution cannot be obtained within 3600 seconds even for the instance with 23 jobs, 4 procedures, and 13 operators. Then we can observe from Table 2 that it is very time-consuming for solving the formulation by calling CPLEX. The average computational time of hill climbing is 1522.5s, which is smaller than that of GA, i.e., 1760.9s. Besides, the average objective values of solutions obtained by hill climbing algorithm are 22.2s, and those obtained by GA are 17.5. For the first 11 instances, we can observe that, compared with hill climbing algorithm, the objective values of solutions obtained by GA are closer to the optimal solution. Therefore, from Table 2, we can conclude that (i) the proposed two heuristics, i.e., hill climbing algorithm and genetic algorithm (GA), are time-saving compared with solving the MIP formulation by calling CPLEX solver, (ii) GA performs better in terms of solution quality compared with hill climbing algorithm, and (iii) hill climbing algorithm is more time-saving than GA.

6. Conclusion

This work investigates the reentrant flow shop scheduling problem, considering multiresource qualification matching, in which operators processing required skill qualifications can be assigned to serve a job in a procedure. The objective of the problem is to schedule the jobs in each machine and assign operators to serve job processing. For the problem, a new mixed integer programming (MIP) formulation is proposed, and two heuristics are then developed, i.e., the hill climbing algorithm and the adapted genetic algorithm (GA). Numerical experiments on 30 randomly generated instances are conducted to evaluate the performance of our proposed MIP formulation and two heuristics. Computational results show that hill climbing is more time-saving, and GA performs better in terms of solution quality.

In the future, we should develop more effective algorithms to solve the problem and improve the quality of the solutions (on the basis of Yin et al. [37]). Future researches also could take more factors into account, including the robustness, work balance of the operators (on the basis of Xu et al., 2014 [27]), maintenance activities (on the basis of Yang and Yang, 2010 [38]), and space constraints (on the basis of Xu et al., 2014 [27]).

Data Availability

The data set in the numerical experiments are randomly generated, which can be accessed by the method stated in Section 5.1.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 71428002, 71531011, 71571134, and 71771048. This work was also supported by the Fundamental Research Funds for the Central Universities.

Table 2: Computational results.

Set	$(\|N\|, \|H\|, \|R_2\|)$	CPLEX		HCA		GA	
		Obj	CT	Obj	CT	Obj	CT
1	(3, 2, 1)	1	142.5	1	0.9	1	1.6
2	(5, 2, 2)	2	215.5	3	1.4	2	3.1
3	(7, 2, 3)	2	628.7	3	2.3	2	6.9
4	(9, 2, 4)	2	1230.4	5	3.8	3	8.6
5	(11, 2, 5)	3	2352.1	8	5.7	4	13.2
6	(13, 3, 6)	5	3325.3	8	74.8	7	105.8
7	(15, 3, 7)	4	3600.0	6	105.7	6	197.2
8	(17, 3, 8)	6	3600.0	9	126.9	7	256.6
9	(19, 3, 9)	7	3600.0	10	189.8	9	385.1
10	(21, 3, 10)	6	3600.0	10	251.7	8	493.3
11	(23, 4, 11)	8	3600.0	15	386.1	12	621.9
12	(25, 4, 12)	-	-	17	452.2	19	828.5
13	(27, 4, 13)	-	-	18	514.3	15	956.4
14	(29, 4, 14)	-	-	21	703.6	20	1129.3
15	(31, 4, 15)	-	-	19	912.5	12	1408.2
16	(33, 4, 16)	-	-	16	1030.4	15	1606.2
17	(35, 4, 17)	-	-	24	1342.5	24	1946.8
18	(37, 4, 18)	-	-	27	1608.0	22	2234.8
19	(39, 4, 19)	-	-	29	1918.2	23	2415.3
20	(41, 4, 20)	-	-	30	2213.4	24	2809.7
21	(43, 5, 21)	-	-	37	2524.2	35	2998.7
22	(45, 5, 22)	-	-	38	2896.1	33	3600.0
23	(47, 5, 23)	-	-	31	3234.3	28	3600.0
24	(49, 5, 24)	-	-	36	3577.1	30	3600.0
25	(51, 5, 25)	-	-	38	3600.0	28	3600.0
26	(53, 5, 26)	-	-	35	3600.0	30	3600.0
27	(55, 5, 27)	-	-	42	3600.0	21	3600.0
28	(57, 5, 28)	-	-	40	3600.0	24	3600.0
29	(59, 5, 29)	-	-	45	3600.0	31	3600.0
30	(61, 5, 30)	-	-	45	3600.0	30	3600.0
Average		-	-	22.2	1522.5	17.5	1760.9

References

[1] S. C. Graves, H. C. Meal, D. Stefek, and A. H. Zeghmi, "Scheduling of re-entrant flow shops," *Journal of Operations Management*, vol. 3, no. 4, pp. 197–207, 1983.

[2] C. Wang, C. Chu, and J. M. Proth, "A branch-and-bound algorithm for n-job two machine flow shop scheduling problems," *Emerging Technologies and Factory Automation*, vol. 2, pp. 375–383, 1995.

[3] D.-L. Yang and M.-S. Chern, "A two-machine flowshop sequencing problem with limited waiting time constraints," *Computers & Industrial Engineering*, vol. 28, no. 1, pp. 63–70, 1995.

[4] H. Hwang and J. U. Sun, "Production sequencing problem with re-entrant work flows and sequence dependent setup times," *International Journal of Production Research*, vol. 36, no. 9, pp. 2435–2450, 1998.

[5] C. Wang, C. Chu, and J.-M. Proth, "Heuristic approaches for n/m/F/Ci scheduling problems," *European Journal of Operational Research*, vol. 96, no. 3, pp. 636–644, 1997.

[6] D.-L. Yang and M.-S. Chern, "Two-machine flowshop group scheduling problem," *Computers & Operations Research*, vol. 27, no. 10, pp. 975–985, 2000.

[7] V. Gordon, J.-M. Proth, and C. Chu, "A survey of the state-of-the-art of common due date assignment and scheduling research," *European Journal of Operational Research*, vol. 139, no. 1, pp. 1–25, 2002.

[8] D.-L. Yang, C.-J. Hsu, and W.-H. Kuo, "A two-machine flowshop scheduling problem with a separated maintenance constraint," *Computers & Operations Research*, vol. 35, no. 3, pp. 876–883, 2008.

[9] I. Kacem and C. Chu, "Worst-case analysis of the wspt and mwspt rules for single machine scheduling with one planned setup period," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1080–1089, 2008.

[10] Y. Ma, C. B. Chu, and C. R. Zuo, "A survey of scheduling with deterministic machine availability constraints," *Computers & Industrial Engineering*, vol. 58, no. 2, pp. 199–211, 2010.

[11] M. Liu and C. Chu, "Optimal semi-online algorithms for *m*-batch-machine flow shop scheduling," *Discrete Mathematics, Algorithms and Applications*, vol. 4, no. 4, 2012.

[12] W. Lei, A. Che, and C. Chu, "Optimal cyclic scheduling of a robotic flowshop with multiple part types and flexible processing times," *European Journal of Industrial Engineering*, vol. 8, no. 2, pp. 143–167, 2014.

[13] S. Wang, M. Liu, and C. Chu, "A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling," *International Journal of Production Research*, vol. 53, no. 4, pp. 1143–1167, 2015.

[14] C.-Y. Lee and T. Lu, "Inventory competition with yield reliability improvement," *Naval Research Logistics (NRL)*, vol. 62, no. 2, pp. 107–126, 2015.

[15] W. H. Kuo, D. L. Yang, and C. J. Hsu, "An unrelated parallel machine scheduling problem with past-sequence-dependent setup time and learning effects," in *Proceedings of the International Conference on Computers and Industrial Engineering*, vol. 35, pp. 764–776, 2010.

[16] S.-J. Yang, D.-L. Yang, and T. C. E. Cheng, "Single-machine due-window assignment and scheduling with job-dependent aging effects and deteriorating maintenance," *Computers & Operations Research*, vol. 37, no. 8, pp. 1510–1514, 2010.

[17] T. Cheng E, C. Hsu J, and D. Yang L, "Unrelated parallel-machine scheduling with deteriorating maintenance activities," *Computers & Industrial Engineering*, vol. 60, no. 4, pp. 602–605, 2011.

[18] Y. Q. Yin, M. Liu, J. H. Hao, and M. C. Zhou, "Single-machine scheduling with job-position-dependent learning and time-dependent deterioration," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 42, no. 1, pp. 192–200, 2012.

[19] Y. Yin, W.-H. Wu, W.-H. Wu, and C.-C. Wu, "A branch-and-bound algorithm for a single machine sequencing to minimize the total tardiness with arbitrary release dates and position-dependent learning effects," *Information Sciences*, vol. 256, pp. 91–108, 2014.

[20] W. Kubiak, S. X. C. Lou, and Y. Wang, "Mean flow time minimization in reentrant job shops with a hub," *Operations Research*, vol. 44, no. 5, pp. 764–776, 1996.

[21] Y. Park, S. Kim, and C.-H. Jun, "Performance analysis of re-entrant flow shop with single-job and batch machines using mean value analysis," *Production Planning and Control*, vol. 11, no. 6, pp. 537–546, 2000.

[22] S.-W. Choi and Y.-D. Kim, "Minimizing makespan on a two-machine re-entrant flowshop," *Journal of the Operational Research Society*, vol. 58, no. 7, pp. 972–981, 2007.

[23] C. Jing, W. Huang, and G. Tang, "Minimizing total completion time for re-entrant flow shop scheduling problems," *Theoretical Computer Science*, vol. 412, no. 48, pp. 6712–6719, 2011.

[24] C. Desprez, F. Chu, and C. Chu, "Minimising the weighted number of tardy jobs in a hybrid flow shop with genetic algorithm," *International Journal of Computer Integrated Manufacturing*, vol. 22, no. 8, pp. 745–757, 2009.

[25] K. Chakhlevitch and C. A. Glass, "Scheduling reentrant jobs on parallel machines with a remote server," *Computers & Operations Research*, vol. 36, no. 9, pp. 2580–2589, 2009.

[26] R.-H. Huang, S.-C. Yu, and C.-W. Kuo, "Reentrant two-stage multiprocessor flow shop scheduling with due windows," *The International Journal of Advanced Manufacturing Technology*, vol. 71, no. 5-8, pp. 1263–1276, 2014.

[27] J. Xu, Y. Yin, T. C. E. Cheng, C.-C. Wu, and S. Gu, "A memetic algorithm for the re-entrant permutation flowshop scheduling problem to minimize the makespan," *Applied Soft Computing*, vol. 24, pp. 277–283, 2014.

[28] C. Sangsawang, K. Sethanan, T. Fujimoto, and M. Gen, "Meta-heuristics optimization approaches for two-stage reentrant flexible flow shop with blocking constraint," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2395–2410, 2015.

[29] B.-H. Zhou, L.-M. Hu, and Z.-Y. Zhong, "A hybrid differential evolution algorithm with estimation of distribution algorithm for reentrant hybrid flow shop scheduling problem," *Neural Computing and Applications*, vol. 8, pp. 1–17, 2016.

[30] S. Dauzère-Pérès, W. Roux, and J. B. Lasserre, "Multi-resource shop scheduling with resource flexibility," *European Journal of Operational Research*, vol. 107, no. 2, pp. 289–305, 1998.

[31] S. Dauzere-Peres and C. Pavageau, "Extensions of an integrated approach for multi-resource shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 33, no. 2, pp. 207–213, 2003.

[32] C. Artigues and F. Roubellat, "A Petri net model and a general method for on and off-line multi-resource shop floor scheduling with setup times," *International Journal of Production Economics*, vol. 74, no. 1-3, pp. 63–75, 2001.

[33] C. Artigues and F. Roubellat, "An efficient algorithm for operation insertion in a multi-resource job-shop schedule with sequence-dependent setup times," *Production Planning and Control*, vol. 13, no. 2, pp. 175–186, 2002.

[34] H.-F. Wang and K.-Y. Wu, "Modeling and analysis for multi-period, multi-product and multi-resource production scheduling," *Journal of Intelligent Manufacturing*, vol. 14, no. 3-4, pp. 297–309, 2003.

[35] M. Rajkumar, P. Asokan, and V. Vamsikrishna, "A GRASP algorithm for flexible job-shop scheduling with maintenance constraints," *International Journal of Production Research*, vol. 48, no. 22, pp. 6821–6836, 2010.

[36] J. H. Holland, "Adaptation in natural and artificial systems," *Quarterly Review of Biology*, vol. 6, no. 2, pp. 126–137, 1975.

[37] Y. Yin, C.-C. Wu, W.-H. Wu, C.-J. Hsu, and W.-H. Wu, "A branch-and-bound procedure for a single-machine earliness scheduling problem with two agents," *Applied Soft Computing*, vol. 13, no. 2, pp. 1042–1054, 2013.

[38] S.-J. Yang and D.-L. Yang, "Minimizing the makespan on single-machine scheduling with aging effect and variable maintenance activities," *Omega* , vol. 38, no. 6, pp. 528–533, 2010.