$FH2(P2, P2)$ hybrid flow shop scheduling with recirculation of jobs

Nadjat Meziani[1] and Mourad Boudhar[2]

[1]University of Abderrahmane Mira Bejaia, Algeria
[2]USTHB University Algiers, Algeria
ro_nadjet07@yahoo.fr and mboudhar@yahoo.fr

Abstract. In this paper, two hybrid flow shop scheduling problems to minimize the makespan are presented. The first is the hybrid flow shop on two stages with only one machine on each stage and recirculation of jobs. This problem is polynomial. The second one is the hybrid flow shop on two stages such as each one contains two identical parallel machines and every job recirculates a finite number of times. This problem is NP-hard in general. Linear mathematical formulation and heuristics are also presented with numerical experimentations.

Key words: Hybrid flow shop, complexity, recirculation, makespan

1 Introduction

Problems of the flood production with recirculation marked the interest of the researchers these last years. In [1], Bertel and Billaut consider the hybrid flow shop scheduling problem with recirculation of jobs and suggest a genetic algorithm to minimize the weighted number of late jobs. To minimize the maximum lateness in the hybrid flow shop scheduling problem with recirculation, Choi et al. [3] propose several lists of scheduling algorithms . An exact method and heuristics were presented in [4], to solve the problem on two stages with recirculation to minimize the makespan under the maximum dues dates. Another type of the hybrid flow shop problem on two stages with recirculation of jobs, to minimize the maximum completion processing times of jobs, was studied in [2] where the authors presented two problems. The first problem is a flow shop with two machines with recirculation of jobs which is polynomial. The second is of the same type, except that the first stage consists of only one machine and the second of two identical parallel machines. This problem is proved NP-hard. In our work, we interest to the hybrid flow shop scheduling problem on two stages with recirculation of jobs with the objective to minimize the makespan.

The problem of jobs recirculation is drawn from a practical application and appears in the workshops painting of the metallic doors, bicycles, cars or other finite or semi-finite processes where each product has to pass through several operations to complete the painting process. Thus any product passes by two stages. On the first stage, each product undergoes a test or a control of quality and compliance on machines for several times and noncomplying products are rejected. The product retaines, will pass on the second stage whose the first operation is the anti-rust treatment, then the product must return once more on the machine to carry out the following operation, which is the actual painting. By recirculating one final time on the same machine, the operation of brightness is applied to each product. The time of drying that separates two operations can be included in the processing time.

In this paper, two scheduling problems to minimize the makespan were studied. The first is the hybrid flow shop on two stages with only one machine on each stage and recirculation of jobs. This problem is polynomial. The second one is the hybrid flow shop on two stages such as each one contains two identical parallel machines and every job recirculates a finite number of times. This problem is NP-hard in general. Linear mathematical formulation and heuristics are also presented with numerical experimentations.

2 Complexity

The two stage hybrid flow shop scheduling problem with m_1 identical parallel machines on the first stage and m_2 identical parallel machines at the second one in order to minimize the makespan (C_{max}), noted by $FH2(Pm_1, Pm_2)//C_{max}$, was studied by Gupta [5]. He has shown that the problem is NP-hard in the strong sens as soon as a stage contains more than one machine whereas its opposite problem was tackled in [6] by Gupta and Tunc. Hoogeveen et al. [8] proved that the same problem with two machines at the first stage and only one machine at the second stage, $FH2(P2, 1)//C_{max}$, is NP-hard in the ordinary sens. They have also proved that this latter problem with preemtion of jobs noted by $FH2(P2, 1)/pmtn/\ C_{max}$ is NP-hard in the strong sens.

The following theorem is a generalization of the theorem of equivalence of the hybrid flow shop scheduling problem with two stages and its reverse stated in [7].

Theorem 1. *The hybrid flow shop problem on two stages with recirculation of jobs on the second stage $FH2(Pm_1, Pm_2)/recr(2)/C_{max}$ and its reverse $FH2$ $(Pm_2, Pm_1)/recr(1)/C_{max}$ with recirculation of jobs on the first stage are equivalent.*

3 Polynomial subproblem: one machine at each stage

We consider the flow shop scheduling problem on two machines with recirculation of jobs on the two machines, $F2/recr(1, 2)/C_{max}$, which the objective is to minimize the makespan C_{max} of the jobs. Let n independent jobs to schedule on these machines. Each job J_i must be processed a finite number of times n_{i1} on the first machine with the processing time p_{1ij} and a finite number n_{i2} on the second machine with the processing time p_{2il}. The objective is to minimize the makespan. For the resolution of the problem, we propose the following algorithm which is based on the Johnson's rule [9] and provided an optimal solution.

Algorithm $F2RC(1, 2)$

1. Transform the considered problem to $F2//C_{max}$ as follows:
   the processing time on the first and the second machine are given respectively by: $p'_{1i} = \sum_{j=1}^{n_{i1}} p_{1ij}$, $p'_{2i} = \sum_{l=1}^{n_{i2}} p_{2il}$
2. Solve the problem $F2//C_{max}$ with Johnson's algorithm.
3. Build the solution of the initial problem $F2/recr(1, 2)/C_{max}$ by subdividing the processing time of each job on the two machines of each stage in processing time of its initial elementary operations.

Theorem 2. *The algorithm F2RC(1,2) provides an optimal solution to the problem $F2/recr(1, 2)/C_{max}$ in $O(max\{nlogn, N\})$ where $N = \sum_{i=1}^{n} (n_{i1} + n_{i2})$.*

Proof. Let us suppose that in an optimal solution of the problem $F2/recr(1, 2)/C_{max}$, we have at least two unspecified operations of the same job which are not processed successively on one or two machines (see Fig. 1).

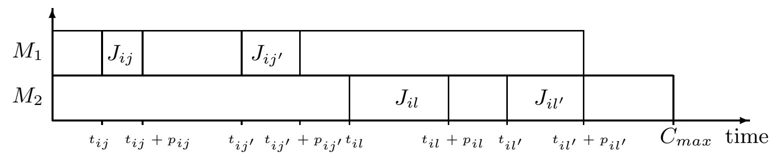

Fig. 1. An example of jobs scheduling

By scheduling on the first machine M_1 the operation J_{ij} at the date $t_{ij'} - p_{ij}$, the completion time of jobs doesn't changed because the operations processed between jobs J_{ij} and $J_{ij'}$ will be moved in the worst of the cases with p_{ij} units of time towards the left, therefore processed before $t_{ij'} + p_{ij'}$. In this manner, the dates of the beginning operations process on the second machine are respected.

By scheduling on the second machine the operation $J_{il'}$ at the date $t_{il} + p_{il}$ the solution remains feasible and C_{max} does not change a value because the treated operations between J_{il} and $J_{il'}$ will be moved in the worst of the cases with $p_{il'}$ units of time towards the right, therefore treated before $t_{il'} + p_{il'}$. By repeating these two operations a finite number of times, we obtain a solution where all the operations of the same job are successively scheduled on the two machines (one after the other without idle time). The first step of the algorithm requires $O(N)$ operations and the algorithm of Johnson turns in $O(nlogn)$. Therefore the problem $F2/recr(1, 2)/C_{max}$ is polynomial.

4 Problem $FH2(P2,P2)/recr(1,2)/C_{max}$

Let n independent jobs to schedule on two stages. The first stage consists of two identical parallel machines M_{11}, M_{12} and the second stage is composed of two others identical parallel machines M_{21}, M_{22}. The workshop is of hybrid flow shop type on two stages. Each job J_i must be processed a finite number of times n_{i1} and n_{i2} with the processing times p_{1ij} and p_{2il} on the first and the second stages respectively. The objective is the minimization of the makespan (C_{max}). The problem is denoted by $FH2(P2,P2)/recr(1,2)/C_{max}$.

The summing up of the processing times operations of the same job and their successive processing on the same machine of the second stage does not always give the best solution. We give here the following counter example:

J_i	J_1	J_2	J_3
nb_{i1}	1	2	1
p_{1i1}	2	1	2
p_{1i2}	/	1	/
nb_{i2}	2	1	2
p_{2i1}	1	3	2
p_{2i2}	2	/	1

Table 1. Processing time of jobs

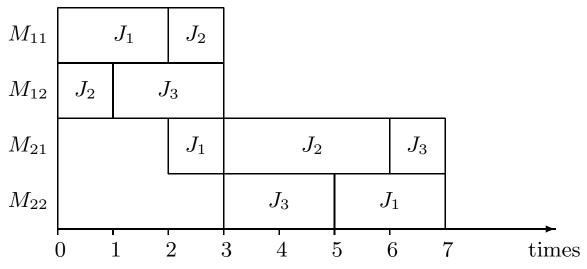

Fig. 2. Scheduling of jobs

4.1 Mathematical modeling

The problem $FH2(P2,P2)/recr(1,2)/C_{max}$ is formulated in a linear mathematical program with real and binary variables.

Variables
Let's consider the following variables s_{ijk}, r_{ilh}, X_{ijk} , $\alpha_{ij}^{i'j'}$, Y_{ilh} and $\beta_{il}^{i'l'}$:

- s_{ijk} : the starting time of the operation j of the job J_i on the machine M_k at the first stage, for $i = \overline{1,n}$; $j = \overline{1,n_{i1}}$; $k = 1,2$;

- r_{ilh} : the starting time of the operation l of the job J_i on the machine M_h at the second stage, for $i = \overline{1,n}$; $l = \overline{1,n_{i2}}$; $h = 3,4$;

- $X_{ijk} = \begin{cases} 1, \text{if the operation } j \text{ of the job } J_i \text{ processed on the machine } k \\ \quad \text{at the first stage;} \\ 0, \text{else.} \end{cases}$
for $i = \overline{1,n}$; $j = \overline{1,n_{i1}}$; $k = 1,2$;

- $\alpha_{ij}^{i'j'} = \begin{cases} 1, \text{if the starting time of the operation } j \text{ of the job } J_i \\ \quad \leq \text{at starting time of the operation } j' \text{ of the job } J_{i'}; \\ 0, \text{else.} \end{cases}$

- $Y_{ilh} = \begin{cases} 1, \text{if the operation } l \text{ of the job } J_i \text{ processed on the machine } h \\ \quad \text{at the second stage ;} \\ 0, \text{else.} \end{cases}$
for $i = \overline{1,n}$; $l = \overline{1,n_{i2}}$; $h = 3,4$;

- $\beta_{il}^{i'l'} = \begin{cases} 1, \text{if the starting time of the operation } l \text{ of the job } J_i \\ \quad \leq \text{at starting time of the operation } l' \text{ of the job } J_{i'}; \\ 0, \text{else.} \end{cases}$
for $i, i' = \overline{1,n}$; $l, l' = \overline{1,n_{i2}}$;

Z : the completion time of all the jobs.

Constraints related to the first stage

– An operation of a job is assigned only to one machine:
$\sum_{k=1}^{2} X_{ijk} = 1$; $\quad i = \overline{1,n}$; $\quad j = \overline{1,n_{i1}}$;

– For any pair of operations we have:
$\alpha_{ij}^{i'j'} + \alpha_{i'j'}^{ij} = 1$; $\quad i, i' = \overline{1,n}$; $\quad j = \overline{1,n_{i1}}$; $\quad j' = \overline{1,n_{i'1}}$; $\quad j \neq j' \; if \; i = i'$;

– On the same machine, the processing of an operation of a job starts only if the processing of the operation which precedes it is over:

$s_{ijk} + p_{1ij} - s_{i'j'k} \leq M_1 \cdot (1 - \alpha_{ij}^{i'j'} + 2 - X_{ijk} - X_{i'j'k})$; $\quad i, i' = \overline{1,n}$; $j = \overline{1,n_{i1}}$; $j' = \overline{1,n_{i'1}}$; $j \neq j'$ if $i = i'$; $\quad k = 1,2$;
$s_{i'j'k} + p_{1i'j'} - s_{ijk} \leq M_1 \cdot (\alpha_{ij}^{i'j'} + 2 - X_{ijk} - X_{i'j'k})$; $\quad i, i' = \overline{1,n}$; $j = \overline{1,n_{i1}}$; $j' = \overline{1,n_{i'1}}$; $j \neq j'$ if $i = i'$; $\quad k = 1,2$;

where M_1 is a very large value which may be equal to $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n_{i1}} p_{1ij}$.

- Two operations of the same job cannot be processed at the same time on two different machines:

  $s_{ijk} + p_{1ij} - s_{ij'k'} \leq M_1 \cdot (1 - \alpha_{ij}^{ij'} + 2 - X_{ijk} - X_{ij'k'})$; $i = \overline{1,n}$; $j, j' = \overline{1, n_{i1}}$; $k, k' = 1, 2$; $j \neq j'$; $k \neq k'$;

  $s_{ij'k} + p_{1ij'} - s_{ijk'} \leq M_1 \cdot (\alpha_{ij}^{ij'} + 2 - X_{ij'k} - X_{ijk'})$; $i = \overline{1,n}$; $j, j' = \overline{1, n_{i1}}$;$k, k' = 1, 2$; $j \neq j'$; $k \neq k'$;

  where M_1 is a very large value which may be equal to $\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n_{i1}} p_{1ij}$.

Constraints related to the second stage

- An operation of a job is assigned only to one machine:
  $\sum\limits_{h=3}^{4} Y_{ilh} = 1$; $i = \overline{1,n}$; $l = \overline{1, n_{i2}}$;
- For any pair of operations we have:
  $\beta_{il}^{i'l'} + \beta_{i'l'}^{il} = 1$; $i, i' = \overline{1,n}$; $l = \overline{1, n_{i2}}$; $l' = \overline{1, n_{i'2}}$; $l \neq l'$ *if* $i = i'$;
- On the same machine, the processing of an operation of a job starts only if the processing of the operation which precedes it is over:

  $r_{ilh} + p_{2il} - r_{i'l'h} \leq M_2 \cdot (1 - \beta_{il}^{i'l'} + 2 - Y_{ilh} - Y_{i'l'h})$; $i, i' = \overline{1,n}$; $l = \overline{1, n_{i2}}$; $l' = \overline{1, n_{i'2}}$; $l \neq l'$ if $i = i'$; $h = 3, 4$;
  $r_{i'l'h} + p_{2i'l'} - r_{ilh} \leq M_2 \cdot (\beta_{il}^{i'l'} + 2 - Y_{ilh} - Y_{i'l'h})$; $i, i' = \overline{1,n}$; $l = \overline{1, n_{i2}}$; $l' = \overline{1, n_{i'2}}$; $l \neq l'$ if $i = i'$; $h = 3, 4$;
  where M_2 is a very large value which may be equal to $\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}$.
- Two operations of the same job cannot be processed at the same time on two different machines:

  $r_{ilh} + p_{2il} - r_{il'h'} \leq M_2 \cdot (1 - \beta_{il}^{il'} + 2 - Y_{ilh} - Y_{il'h'})$; $i = \overline{1,n}$; $l, l' = \overline{1, n_{i2}}$; $h, h' = 3, 4$; $l \neq l'$; $h \neq h'$;
  $r_{ij'k} + p_{2ij'} - r_{ijk'} \leq M_2 \cdot (\beta_{il}^{il'} + 2 - Y_{il'h} - Y_{ilh'})$; $i = \overline{1,n}$; $l, l' = \overline{1, n_{i2}}$; $h, h' = 3, 4$; $l \neq l'$; $h \neq h'$;
  where M_2 is a very large value which may be equal to $\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}$.

Constraints binding the starting times of the jobs of the first stage with the second stage

- If no sequence is imposed for the operations:

- The processing of a job starts on the second stage only if the processing of its last operation on the first stage is over:
  $s_{ijk}+p_{1ij} \le r_{ilh}$; $i=\overline{1,n}$; $j=\overline{1,n_{i1}}$; $k=1,2$; $l=\overline{1,n_{i2}}$; $h=3,4$;

- The completion time of a job on the second stage is lower than, or equal to, Z :
  $r_{ilh}+p_{2il} \le Z$; $i=\overline{1,n}$; $l=\overline{1,n_{i2}}$; $h=3,4$;

– If an operation sequence is imposed:

- The execution of an operation of a job J_i can start only if the processing of the preceding operation is over:
  $s_{i1k}+p_{1i1} \le s_{i2k}$; $i=\overline{1,n}$; $k=1,2$;

  $\vdots$

  $s_{i(n_{i1}-1)k}+p_{1i(n_{i1}-1)} \le s_{in_{i1}k}$; $i=\overline{1,n}$; $k=1,2$;

  $s_{in_{i1}k}+p_{1in_{i1}} \le r_{i1h}$; $i=\overline{1,n}$; $k=1,2$ $h=3,4$;

  $r_{i1h}+p_{2i1} \le r_{i2h}$; $i=\overline{1,n}$; $k=1,2$; $h=3,4$;

  $\vdots$

  $r_{i(n_{i2}-1)h}+p_{2i(n_{i2}-1)} \le r_{in_{i2}h}$; $i=\overline{1,n}$; $k=1,2$; $h=3,4$;

- The completion time of a job on the second stage is lower than, or equal to, Z :
  $r_{in_{i2}k}+p_{2in_{i2}} \le Z$; $i=\overline{1,n}$; $k=1,2$; $h=3,4$;

The objective function is: $min(Z)$.

4.2 Lower bounds

In what follows, two lower bounds on the makespan C_{max} are proposed: LB_1 and LB_2.

Proposition 1. $LB_1 = \min\limits_{1\le i\le n}\left\{\sum\limits_{j=1}^{n_{i1}} p_{1ij}\right\} + \left\lceil \frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}\right\rceil$ *is a lower bound on the makespan.*

Proof. $\left\lceil \frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}\right\rceil$ is a lower bound for the total completion time of the operations in the second stage if their processing begins at $t=0$. It is deduced from the problem $P2//C_{max}$, and the processing of the operations in the second stage can begin only if at least one operation of the same job is completed in the first stage.

Proposition 2. *$LB_2 = \left\lceil \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n_{i1}} p_{1ij} \right\rceil + \min_{1 \leq i \leq n} \left\{ \sum_{l=1}^{n_{i2}} p_{2il} \right\}$ is a lower bound on the makespan.*

Proof. $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n_{i1}} p_{1ij} \right\rceil$ a lower bound for the total completion time of the operations in the first stage. It is deduced from the problem $P2//C_{max}$. Also, the processing of the operations in the second stage can begin only if at least one operation of the same job is completed in the first stage.

Consequently $LB = max\{LB_1, LB_2\}$ is also a lower bound.

4.3 Heuristics

For the resolution of the $FH2(P2, P2)/recr(1, 2)/C_{max}$ problem, we present three heuristics.

Heuristic1 The heuristic $H1\_J$ is based on the algorithm $F2RC(1, 2)$ which determines the sequence of jobs to schedule on the first stage according to the Johnson rule[9], then jobs are assigned on the second stage according to the First Available Machine (FAM) rule.

H1_J heuristic

1. Transform the problem $FH2(P2, P2)/recr(1, 2)/C_{max}$ into the problem $F2//C_{max}$, by considering only one machine on each stage, and by summing up the processing times of the operations of each job on the two stages.
2. Solve the obtained problem, $F2//C_{max}$, by Johnson algorithm's.
3. Process the jobs on the two stages according to their order obtained at the preceding stage.
4. Return to the initial problem.

Heuristic2 In the second heuristic $H2\_SPT$, we add initially, the processing time of the operations of each job on the two stages. Then, to obtain the sequence of the jobs to be followed, we applie the Shortest Processing Time (SPT) rule according the processing time of the jobs of the first stage for then affecting them on the two stages according to the third phase of this heuristic.

H2_SPT heuristic
Step 1: Add the processing time of the operations of each job on the two stages.
Step 2: Apply the SPT (Shortest Processing Time) rule over processing times of the new jobs obtained on the first stage.
Step 3: Assignment of jobs:

– For i:=1 to n do
  Let r^1 and r^2 be the smallest times of availabilities of the machines of the first and second stages

```
    – If $r^1 + p_{1i} \geq r^2$ then
    - The job $J_i$ is assigned on the machine realizing $r^1$.
    - On the second stage, it's assigned on the machine which minimizes its
      completion time. In the event of the multiple choice, take that which is
      free latest.
    – Else the job $J_i$ is assigned on the machine of the second stage realizing
      $r^2$. On the first stage, it's assigned on the machine which minimize its
      waiting.
    – Endif
– Update $r^1$ et $r^2$.
– Enddo.
– Return to the initial problem.
```

Heuristic3 The only difference from the $H2\_SPT$ heuristic, is that on stage two the Longest Processing Time (LPT) rule is applied unstead of Shortest Processing Time (SPT) rule.

H3_LPT heuristic

Step 1: Add the processing time of the operations of each job on the two stages.
Step 2: Apply the LPT (Longest Processing Time) rule over processing times of the new jobs obtained on the first stage.
Step 3: Assignment of jobs:

```
– For i:=1 to n do
  Let $r^1$ and $r^2$ be the smallest times of availabilities of the machines of the
  first and second stages
    – If $r^1 + p_{1i} \geq r^2$ then
    - The job $J_i$ is assigned on the machine realizing $r^1$.
    - On the second stage, it's assigned on the machine which minimizes its
      completion time. In the event of the multiple choice, take that which is
      free latest.
    – Else the job $J_i$ is assigned on the machine of the second stage realizing
      $r^2$. On the first stage, it's assigned on the machine which minimizes its
      waiting.
    – Endif
– Update $r^1$ et $r^2$.
– Enddo.
– Return over to the initial problem.
```

5 Experimentations

Several instances are randomly generated according to the uniform law on which we applied the three heuristics cited above. For each instance, the number n of jobs take its values on the set $[10, 20, 50, 100, 250, 500, 1000]$. Processing times of the jobs will be taken in the intervals $[1, 50]$. Operation numbers of the jobs will be taken in the intervals $[1, 5]$ and $[1, 10]$. We have coded our algorithms in Delphi (Version7.0) and have run them on a Pentium III 1.0GHz Personal Computer with 256MB RAM.

The results obtained are given in a table below. For each value of n, 100 instances are generated. The first line indicated the percentage where the solution found by the heuristic is better compared with other solutions, the second line indicated the average execution time of each heuristic (in milliseconds).

		$p_{1ij}, p_{2il} \in [1, 50]$ $n_{i1}, n_{i2} \in [1, 5]$			$p_{1ij}, p_{2il} \in [1, 50]$ $n_{i1}, n_{i2} \in [1, 10]$		
		$H1\_J$	$H2\_SPT$	$H3\_LPT$	$H1\_J$	$H2\_SPT$	$H3\_LPT$
$n = 10$	C_{max}	61	11	28	66	16	18
	Avr-time	2.7	1.52	3	2.1	1.9	3.6
$n = 20$	C_{max}	68	11	21	68	6	26
	Avr-time	2.8	2.2	6.32	3.01	2.6	6.5
$n = 50$	C_{max}	70	7	23	68	4	28
	Avr-time	5.5	4.4	14.83	5.2	5.5	17.12
$n = 100$	C_{max}	73	3	24	74	3	23
	Avr-time	8.51	12.23	26.23	7.91	10.51	33.64
$n = 250$	C_{max}	76	2	22	76	2	22
	Avr-time	19.52	28.11	69.84	18.61	27.72	84.91
$n = 500$	C_{max}	80	5	15	80	1	19
	Avr-time	41.76	62.91	145.98	43.27	64.9	185.28
$n = 1000$	C_{max}	84	4	12	84	1	15
	Avr-time	107.7	150.62	336.37	110.88	152.81	414.58

According to the results obtained, we note that the heuristics $H1\_J$ works better for any number of jobs and a number of operations with the average execution time lower than the other two heuristics if the number of jobs is large.

6 Conclusion

We have presented two recirculate hybrid flow shop scheduling problems with two stages to minimize the maximum completion time. The first problem is the scheduling on two machines with recirculation of jobs on these latter. This problem is polynomial and an algorithm for its resolution is proposed. The second problem consists of scheduling on two stages, with two identical parallel machines on every one. Jobs can be treated a finite number of times on the two stages. This problem is NP-hard and it is formulated as a linear mathematical program in real and binary variables. We also proposed lower bounds and heuristics, that we have tested, for the second problem. Like prospect for our work, we plan to use other methods of resolution such as the exact methods and metaheuristics.

References

1. Billaut, J.C., Bertel, S.: A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation. European J. Oper. Res. 159, 651–662 (2004)
2. Boudhar, M., Meziani, N.: Two-stage hybrid flow shop with recirculation. Int. Trans. Oper. Res. 17(2), 239–255, (2010)
3. Choi S.W., Kim Y.D., Lee G.C.: Minimizing total tardiness of orders with reentrant lots in a hybrid flow shop. Int. J. Prod. Res. 43, 2149–2167 (2005)
4. Choi H.S., Kim H.W., Lee D.H., Yoon J., Yun Y.C. and Chae K.B.: Scheduling algorithms for two-stage reentrant hybrid flow shops: minimizing makespan under the maximum allowable due dates. The Int. J. Advanced Man. Tech. 42 (2009)
5. Gupta J.N.D.: Two-stage, hybrid flow shop Scheduling problem. J. Oper. Res. Soc. 39, 359–364 (1988)
6. Gupta J.N.D., Tunc E.A.: Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. Int. J. Prod. Res. 29(7), 1489–1502 (1991)
7. Gupta J.N.D., Hariri A.M.A., Potts C.N.: Scheduling a two-stage hybrid flow shop with parallel machines at the first stage, Annals Oper. Res. 69, 171–191(1997)
8. Hoogeveen J.A., Lenstra J.K., Veltman B.: Preemptive scheduling in a two stage multiprocessor flow shop is NP-hard. Euro. J. Oper. Res. 89, 172–175 (1996)
9. Johnson S.M.: Optimal two and three stage production schedules with setup time included. Nav. Res. log. Quar. 1, 61–67 (1954)