KYBERNETIKA — VOLUME 47 (2011), NUMBER 1, PAGES 50–59

HYBRID FLOW-SHOP WITH ADJUSTMENT

Jan Pelikán

The subject of this paper is a flow-shop based on a case study aimed at the optimisation of ordering production jobs in mechanical engineering, in order to minimize the overall processing time, the makespan. The production jobs are processed by machines, and each job is assigned to a certain machine for technological reasons. Before processing a job, the machine has to be adjusted; there is only one adjuster who adjusts all of the machines. This problem is treated as a hybrid two-stage flow-shop: the first stage of the job processing is represented by the machine adjustment for the respective job, and the second stage by the processing of the job itself on the adjusted machine. In other words, the job-processing consists of two tasks, where the first task is the machine adjustment for the job, and the second task is the job processing itself. A mathematical model is proposed, a heuristic method is formulated, and the NP hardness of the problem, called a "hybrid flow-shop with adjustment," is proved.

Keywords: flow-shop, case study, integer programming, heuristics

Classification: 90B35, 90B90

1. INTRODUCTION – DESCRIPTION OF CASE STUDY

The case study consists of the scheduling and ordering of production jobs in a K-Baas production plant. There is a given set of production jobs to be processed by the machines installed on the shop floor. A job is a product batch for a number of units of given product assigned to processing on the machine. The goal is to minimize the makespan value. Each job is assigned to a certain machine, which has to be adjusted by an adjuster. This worker adjusts all of the machines installed on the floor but at a given time, he can adjust only one machine, i. e. he cannot adjust more machines simultaneously. Each adjusted machine is supposed to be capable of immediate starting to process the job for which it has been adjusted. No job processing is allowed to be broken (i. e., intermittent processing is not admissible), and each machine is able to process only one job at a time. The processing time minimization problem is aimed at the reduction of the waiting times of both the machines for the adjuster and of the adjuster for a machine to be adjusted. After completing a job on the machine, the machine has to wait if the adjuster is finishes adjusting another machine. If all machines are processing jobs, the adjuster has to wait. A solution is represented by the order of the jobs in which the adjuster adjusts the respective machines. This order also determines the order of the machines to be adjusted by the adjuster. At the same time, the job ordering generates the order of the jobs on the machines when several jobs are assigned to the same machine.

In the literature, hybrid flow-shop is defined as a problem of processing jobs which consists of two or more stages, with one or more processors at each stage [1]. Each of the jobs to be processed consists of two or more tasks and each task is processed within its own stage. The jobs are non-preemptable and each subsequent stage is only started after the processing of the previous stage is completed.

Hybrid flow-shop consisting of two stages is denoted HF2; in case of one processor on the first stage and m processors on the second stage, its notation is $\mathrm{HF2}_{1,\mathrm{m}}$. Such problem assumes the job can be processed on the first stage immediately after the previous job has been finished on this stage. Then, the job is being processed on any free processors on the second stage.

The flow-shop problem presented in case study also consists of two stages. On the first stage the processor adjusts the production machine which acts as the processor on the second stage. Thus, on the first stage there is the only one processor (adjuster), on the second stage there are m different processors (production batches, jobs are explicitly dedicated to one of them). We can denote this problem as hybrid flow-shop with adjustment HF2a, eventually $\mathrm{HF2a}_{1,\mathrm{m}}$. There are too significant differences between $\mathrm{HF2a}_{1,\mathrm{m}}$ and $\mathrm{HF2}_{1,\mathrm{m}}$:

a) jobs are dedicated to the processors on the second stage,

b) job can be processed on the first stage, if the processor is free on the first stage and dedicated processor on the second stage is released.

2. MATHEMATICAL MODEL

Let there be given a two-stage problem with a sole processor at the first stage, denoted by P_0, and processors $P_1, P_2, \ldots, P_m$ at the second stage. Denote by n the number of jobs $J_1, J_2, \ldots, J_n$; each of these jobs is first processed at the first stage and then at the second stage. Let us assume that job J_i is assigned to processor $P_{v(i)}$ at the second stage and the processing times are t_i^0 at the first stage and t_i^1 at the second stage. Denote by $S_k = \{J_i;\ \nu(i) = k,\ i = 1, 2, \ldots, n\}$ the set of jobs assigned to processor P_k.

Let us introduce binary variables x_{ij} $(i \neq j)$, which contain information about the ordering of the jobs processed at the first stage, i. e., on processor P_0, as follows: $x_{ij} = 1$ if J_i is processed before J_j, and $x_{ij} = 0$ if they are processed in the reverse order.

Parameters of the model:

n — a number of jobs;

m — a number of processors;

t_i^0 — processing time of job J_i on processor P_0 (the first stage);

t_i^1 — processing time of job J_i on processor $P_{\nu(i)}$ (the second stage);

$\nu(i)$ denotes the index of the second-stage processor on which ith job is processed;

$M \gg 0$ — a big number.

Model variables:

$C_{\max}$ — makespan, which is the total processing time of all jobs;

x_{ij} — binary variables determining the order of the jobs on the processor P_0 (on the first stage);

t_i — starting time of processing job J_i on processor P_0.

Model:

$$
\begin{aligned}
&C_{\max} \longrightarrow \min && && (1)\\
&x_{ij} + x_{ji} = 1 && i, j = 1, \ldots, n, \quad i < j, && (2)\\
&t_j \geq t_i + t_i^0 + t_i^1 - M(1 - x_{ij}) && i, j = 1, \ldots, n, \quad i \neq j, \quad \nu(i) = \nu(j), && (3)\\
&t_j \geq t_i + t_i^0 - M(1 - x_{ij}) && i, j = 1, \ldots, n, \quad i \neq j, && (4)\\
&t_i + t_i^0 + t_i^1 \leq C_{\max} && i = 1, \ldots, n, && (5)\\
&x_{ij} \in \{0, 1\} && i, j = 1, \ldots, n, \quad i \neq j, && (6)\\
&t_i \geq 0 && i = 1, \ldots, n. && (7)
\end{aligned}
$$

Equation (2) ensures that either job J_i is processed before J_j on P_0 or vice versa.

Inequalities (3) ensure that starting time t_j of the processing job J_j on P_0 has to follow the time $t_i + t_i^0 + t_i^1$ at which a preceding job J_i has been finished on processor $P_{\nu(i)}$ under the assumption that job J_i precedes job J_j on the first stage and both jobs are dedicated to the same processor on the second stage, i. e. $\nu(i) = \nu(j)$.

Processor P_0 cannot process more than one machine at the same time, so if job J_i precedes job J_j on the first stage, starting time t_j job J_j of the processing on P_0 has to follow the moment $t_i + t_i^0$ when the previous job J_i is finished on P_0; this is the inequality (4).

Inequalities (5) determine the total processing time of all jobs, which is (according to (1)) minimized. Time $C_{\max}$ has to be greater than or equal $t_i + t_i^0 + t_i^1$, for $i = 1, 2, \ldots, n$.

Example 1. As an example of solving the problem, let us refer to job processing optimization of batches at a K-Baas production plant within one working week, i. e. 4500 minutes. There are 10 machines on the shop floor and 27 jobs are processed, as shown in the Table 1.

The mathematical model was solved using the program CPLEX 11.0 for a one-week production program from the Table 1, a number of production batches was $n = 27$ and a number of machines $m = 10$. Sets S_k are given in Table 1, from which it

Table 1. The set of batches.

Job (batch) number	Machine	Amount in pieces	Adjustment time (minutes)	Processing time (minutes)
1	1	1000	8.16	100.00
2	1	50	8.16	5.50
3	2	1000	10.20	260.00
4	2	50	10.20	16.50
5	2	50	10.20	20.50
6	3	1500	61.20	2370.00
7	4	2000	91.80	2440.00
8	5	2000	61.20	1380.00
9	6	667	61.20	1280.64
10	7	100	61.20	40.00
11	7	300	61.20	54.00
12	7	100	61.20	82.00
13	8	200	91.80	172.00
14	9	1000	91.80	2800.00
15	9	100	91.80	11.00
16	9	100	91.80	168.00
17	10	100	6.00	14.00
18	10	2000	6.00	460.00
19	10	1500	6.00	795.00
20	10	1000	6.00	140.00
21	10	100	6.00	14.00
22	10	100	6.00	45.00
23	10	300	6.00	18.00
24	10	2000	6.00	820.00
25	10	100	6.00	28.00
26	10	667	6.00	426.88
27	10	200	6.00	58.00

follows that $S_1 = \{1,2\}, S_2 = \{3,4,5\}, S_3 = \{6\}, S_4 = \{7\}, S_5 = \{8\}, S_6 = \{9\}, S_7 = \{10,11,12\}, S_8 = \{13\}, S_9 = \{14,15,16\}, S_{10} = \{17,18,19,20,21,22,23,24,25,26,27\}$.

The mathematical model includes 757 variables (729 of them are binary variables) and 1238 constraints. The computations took 3.1 minutes (PC 2.1 GHz). The resulting order, in which the machines are adjusted, is (14, 20, 4, 8, 18, 2, 7, 27, 19, 6, 26, 5, 10, 12, 9, 22, 24, 3, 13, 11, 15, 16, 1, 21, 17, 23, 25). The optimal makespan is 3254.4 minutes, which is about 30% shorter than the completion time in reality.

3. COMPUTATIONAL COMPLEXITY OF THE PROBLEM HF2a

In order to prove the NP hardness of HF2a, we will show that a partition problem can be reduced to the decision form of HF2a (similarly in [2, 3, 4]).

Partition problem.

Input: Given positive integers $a_1, a_2, \dots, a_n$ for which it holds $\sum_{i=1}^{n} a_i = 2B$.

Output: determine if there exist sets A_1, A_2 for which it holds

$$\sum_{i\in A_1} a_i = \sum_{i\in A_2} a_i = B, \quad A_1, A_2 \subset \{1,2,\dots,n\}, \quad A_1 \cap A_2 = \emptyset.$$

Decision form of HF2a.

Input: Given $m+1$ processors $P_0, P_1, \dots, P_m$ and n jobs $J_1, J_2, \dots, J_n$ with processing times t_i^0 for the job $i = 1, 2, \dots, n$ and processors P_0 and t_i^1 for job J_i which is dedicated to processor $P_{\nu(i)}$. A deadline T is given.

Output: determine if there exists a schedule for which $C_{\max} \le T$.

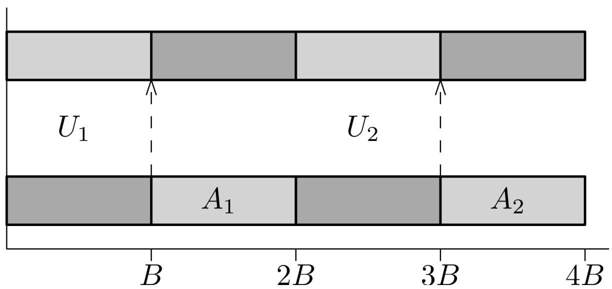

Figure. Optimal schedule of HF2a.

Proposition 1. The partition problem can be reduced to the decision form of HF2a.

Proof. Let the positive integers $a_1, a_2, \dots, a_n$ be given and $\sum_{i=1}^{n} a_i = 2B$ holds. Define jobs U_1, U_2 and $V_1, V_2, \dots, V_n$, and processors P_0, P_1, P_2. The adjustment and processing times are given in Table 2.

Table 2. Adjustment and processing times.

	U_1	U_2	V_1	V_2	$\dots$	V_n
Adjustment time t_i^0	B	B	a_1	a_2		a_n
Dedicated processor $P_{\nu(i)}$	P_1	P_1	P_2	P_2	P_2	P_2
Processing time t_i^1	B	B	0	0	0	0

The optimal schedule of U_1 and U_2 is shown in Figure and Table 3, with the corresponding makespan value equal to $4B$. If there exists a partition A_1 and A_2

Table 3. Optimal schedule of U_1 and U_2.

	P_0 start	P_0 finish	P_1 start	P_1 finish
	t_i	$t_i + t_i^0$	$t_i + t_i^0$	$t_i + t_i^0 + t_i^1$
U_1	0	B	B	$2B$
U_2	$2B$	$3B$	$3B$	$4B$

for which it holds $\sum_{i \in A_1} a_i = \sum_{i \in A_2} a_i = B$, $\quad A_1, A_2 \subset \{1, 2, \ldots, n\}$, jobs J_i for $i \in A_1$ can be scheduled on P_0 within the interval $\langle B, 2B\rangle$ in any order, and jobs J_i for $i \in A_2$ within the interval $\langle 3B, 4B\rangle$. The makespan value does not change, it equals $4B$ and it is optimal. Now set the deadline $T = 4B$ and find the schedule with the makespan value $C_{\max} \leq T = 4B$ of the decision form of HF2a for the given set of jobs U_1, U_2, V_1, V_2, $\ldots$, V_n.

If such a schedule exists, it should be like the schedule shown in Table 3 (the order of jobs U_1 and U_2 can be arbitrary).

Denote by A_1 the set of indices for the jobs which are scheduled on P_0 in the interval $\langle B, 2B\rangle$, and by A_2 for the jobs on the interval $\langle 3B, 4B\rangle$. As the lengths of both intervals are equal to B, it holds $\sum_{i \in A_1} a_i = \sum_{i \in A_2} a_i = B$, and the result of the partition problem is YES; otherwise it is NO. $\square$

Comment. It can be proved the NP-hardness in strong sense for HF2a. Proof is based on reduction of 3-partition problem on HF2a.

3-partition problem. Given a positive integer B and a multiset A of positive integers $A = \{a_1, a_2, \ldots, a_p\}$ with $p = 3n$, $\sum_{i=1}^{p} a_i = nB$ and $B/4 < a_i < B/2$ for $i = 1, 2, \ldots, p$. Does there exist a partition of A into 3 element sets $\{A_1, A_2, \ldots, A_n\}$ such that $\sum_{i \in A_k} a_i = B$ for $k = 1, 2, \ldots, n$?

Lets define HF2a problem: There are jobs: $U_1, U_2, \ldots, U_n$ and $V_1, V_2, \ldots, V_{3n}$ and processors P_0, P_1 and P_2 with:

$$t^0_{U_i} = B, \quad t^1_{U_i} = B, \quad P_{\nu(i)} = P_1, \quad i = 1, 2, \ldots, n,$$

$$t^0_{V_i} = a_i, \quad t^1_{V_i} = 0, \quad P_{\nu(i)} = P_2, \quad i = 1, 2, \ldots, 3n.$$

Proposition 2. Optimal makespan is $C_{\max} = 2nB$ if and only if $\sum_{i \in A_k} a_i = B$ for $k = 1, 2, \ldots, n$, where A_k are a set of $t^0_{V_i} = a_i$ of the jobs V_i scheduled in the interval $\langle (2k-1)B, 2kB\rangle$.

P r o o f. The optimal makespan for the jobs $U_1, U_2, \ldots, U_n$ is $2nB$ and is independent on ordering those jobs. There are time windows on the P_0 in this optimal schedule in the form $\langle (2k-1)B, 2kB\rangle$, $k = 1, 2, \ldots, n$. The optimal makespan all jobs $U_i, i = 1, 2, \ldots, n$ and $V_j, j = 1, 2, \ldots, 3n$ remains to be $2nB$ only if there is possible schedule all jobs V_j into those time windows on processor P_0 and follows it there exists 3-partition of the set A. $\square$

4. HEURISTIC METHOD

Due to NP hardness, it will be useful to use a heuristic method in case a huge number of jobs and processors. The proposed heuristic method constructs the order of jobs on the first stage in successive steps in the form $(J_{\pi(1)}, J_{\pi(2)}, \ldots, J_{\pi(n)})$, where $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ is a permutation of the numbers $(1, 2, \ldots, n)$. Next job is chosen on the base of the following aspects:

α) the job is dedicated to the processor P_k for which the lower bound lb_k of the completion time of all jobs dedicated to this processor is maximal,

β) the processing time of the job on the processor chosen in α) is minimal.

Let us denote $\overline{S_k}$ a set of jobs dedicated to processor P_k, which has not been placed in the resulting order, lb_k the lower bound the completion time of all jobs dedicated to processor P_k $(k = 1, 2, \ldots, m)$, T_k^f the release time of processor P_k, T the time at which we can start to process the next job and add it to the result order of jobs.

Step 1. Put $T{:=}0$, $\overline{S_k}{:=}S_k$, $T_k^f{:=}0$, $k = 1, 2, \ldots, m$, $i{:=}1$.

Step 2. Put $lb_k{:=} T + \sum_{j \in \overline{S_k}} t_j^0 + t_j^1$, $k = 1, 2, \ldots, m$.

Step 3. Choose processor P_k for which:

a) $\overline{S_k}$ not empty,

b) $T_k^f \leq T$,

c) lb_k is maximal for k satisfying a), b).

Step 4. Chose j from $\overline{S_k}$ for which time t_j^0 is minimal.

Step 5. Put $\pi(i){:=}j$, $\overline{S_k}{:=}\overline{S_k} - \{j\}$, $T_k^f{:=} T + t_j^0 + t_j^1$,

$T{:=}\max\{T + t_j^0, \ \min\{T_l^f;\ l = 1, 2, \ldots, m,\ \overline{S_l} \neq \emptyset\}\}$.

Step 6. If $i < n$ then $i{:=}i + 1$, go to step 2, else stop.

The set of batches on the Table 1 was solved by the proposed heuristic method, the result order of batches is:

$$\pi = (14, 17, 7, 18, 6, 8, 9, 10, 3, 13, 11, 1, 19, 12, 4, 2, 5, 20, 21, 22, 23, 24, 25, 26, 15, 27, 16)$$

and the makespan is 3256.28, which is value close to the optimal value.

5. NUMERICAL EXPERIMENTS

The model and heuristic method were tested on problems PR1, …, PR10 that had been proposed to be different as to time values of adjustment and production, size and a number of jobs dedicated to a processor. In Table 4 there are presented results obtained on PC 2.1 GHz and CPLEX 11.0, model and heuristics experiments are compared.

Table 4. Numerical experiments – results.

(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)
PR1	27	10	1	11	3254.4	3256.28		19 sec.
PR2	27	10	1	11	3293.4	3293.4		3.7 sec.
PR3	20	10	2	2	3760	3760		0.4 sec.
PR4	20	10	1	4	5147	5147		8.55 sec.
PR5	30	5	2	9	10620	10620*	6970	3600 sec.*
PR6	30	5	4	9	10639	10620*	8850	3600 sec.*
PR7	30	5	6	6	8720	8710		268 sec.
PR8	30	5	6	6	33185	33185*	7162	3600 sec.*

Comments:

(a) . . . the problem name
(b) . . . n a number of jobs
(c) . . . m a number of machines
(d) . . . the minimum number of jobs dedicated to one processor
(e) . . . the maximum number of jobs dedicated to one processor
(f) . . . the makespan of the heuristic solution
(g) . . . the makespan of the optimal solution in CPLEX11.0, in case * the best value of makespan obtained by interruption of the computation (after 1 hour)
(h) . . . the lower bound of the makespan obtained by interruption of the computation
(i) . . . the computer time of the solution in CPLEX11.0 on PC 2.1 GHz
* . . . computation was interrupted
PR1 . . . the case study problem
PR2 . . . minor changes in time values in PR1
PR3 . . . two jobs are dedicated to each machine
PR4 . . . a number of jobs dedicated to one machine is gradually being increased
PR5 – PR8 . . . a total number of jobs n is increased
PR8 . . . the adjusting time is greater than production time of jobs.

The solution of these problems obtained with use of heuristics seems to be satisfactory, especially in case of large-sized problems.

The optimal solution was obtained in problems PR1, PR2 and PR4. In case of problems PR5, PR6, PR8 the computation was interrupted after 1 hour. Thus, an optimal solution was not obtained, the best solution and the lower bound of the makespan is shown in Table 4. Hence the value of the makespan of the heuristic method cannot be discussed in this case. For problems PR1 and PR7 the heuristic method did not provide the optimal solution (the same result can be shown for smaller problems, e. g. for the case of two processors and three jobs). The solution obtained by the heuristic method is worse than the best solution obtained after the interruption of computation in the problem PR7.

6. HYBRID FLOW-SHOP WITH ADJUSTMENT WITH R ADJUSTERS HF2a(R)

We will suppose $1 < R < m$, where R is a number of adjusters, i. e. a number of adjusters is less than a number of machines on the second stage. In case $R \geq m$ each machine is equipped with an adjuster and therefore the makespan is independent on jobs scheduling.

At first we modify the mathematical model. Let us denote $P_1^0, P_2^0, \dots, P_R^0$ processors on the first stage. The assignment of jobs to adjusters will be solved. Binary variable y_{ir} is equal to 1 if job J_i is assigned to rth adjuster P_r^0, value 0 otherwise. In the model (1) - (7) the inequality (3) is modified to the following form:

$$t_j \geq t_i + t_i^0 + t_i^1 - M(1-x_{ij}) - M(1-w_{ijr}) \quad i,j = 1,2,\dots,n,\ i \neq j,\ r = 1,\dots,R, \quad (3a)$$

where $w_{ijr} = y_{ir} \cdot y_{jr}$. Furthermore, the conditions of unique assignment of jobs to adjusters have to be added:

$$\sum_{r=1}^{R} y_{ir} = 1, \quad i = 1,2,\dots,n \quad (8)$$

and the conditions for the variable w_{ijr}:

$$x_{ij} + x_{ji} \leq w_{ijr} \quad i,j = 1,2,\dots,n, \quad i \neq j,\ r = 1,2,\dots,R, \quad (9)$$

$$y_{ir} + y_{jr} - 1 \leq w_{ijr} \leq \frac{1}{2}(y_{ir} + y_{jr}) \quad i,j = 1,2,\dots,n,\ i \neq j,\ r = 1,2,\dots,R. \quad (10)$$

Proposition 3. HF2a(R) is NP-hard in strong sense.

P r o o f. Proof is similar to the proof in proposition 1. Let us define jobs $U_1, U_2, V_1, V_2, \dots, V_n$ and jobs $W_1, W_2, \dots, W_{R-1}$,

$$t_{U_i}^0 = B, \quad t_{U_i}^1 = B, \quad P_{U(i)} = P_1, \quad i = 1,2,\dots,n,$$

$$t_{V_i}^0 = a_i, \quad t_{V_i}^1 = 0, \quad P_{V(i)} = P_2, \quad i = 1,2,\dots,3n,$$

with

$$t_{W_i}^0 = 2nB, \quad t_{W_i}^1 = 0, \quad P_{W(i)} = P_3, \quad i = 1,2,\dots,R-1.$$

It holds the optimal makespan $C_{\max} = 2nB$ if and only if the 3-partition of A exists. □

Comment. If there are two adjusters in the problem and case study PR1 (see Table 1) then, using the model proposed above, the optimal makespan would not be lower than in case of one adjuster. Thus adding one adjuster would not decrease the makespan, only the idle time of adjusters will be higher.

7. CONCLUSIONS

The paper describes a case study of job scheduling in a mechanical-engineering production plant. The problem is characterized as a hybrid flow-shop consisting of two stages, where the first stage contains one processor and the second stage contains multiple processors and each job is assigned to one of the second-stage processors. It is a new type of flow-shop in which the first-stage scheduling depends upon the time scheduling of the second stage. This problem has been proved to be NP hard. Both a mathematical model and a heuristic method have been proposed. The case study is solved with the aid of both the model and the heuristic method and the solution achieved represents a 30% reduction of the processing time for the given set of jobs, in comparison with the actual job scheduling used in practice.

ACKNOWLEDGEMENT

This research was supported by GA ČR grant No. GA CR 402/09/0041.

(Received March 11, 2010)

REFERENCES

[1] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz: Scheduling Computer and Manufacturing Processes. Springer–Verlag, Berlin 1996.

[2] Y. Cho, S. Sahni: Preemptive scheduling of independent jobs with release and due times on open, flow and job shops. Oper. Res. *29* (1981), 3, 511–522.

[3] M. Kubzin and A. Strusevich: Planning machine maintenance in two-machine shop scheduling. Oper. Res. *54*, 4, 2006, 789–800.

[4] J. K. Lenstra and A. H. G. Rinnooy Kan: Complexity of scheduling under precedence constraints. Oper. Res. *26* (1978), 1, 22–35.

Jan Pelikán, University of Economics Prague, W. Churchill sq. 4, 130 67 Praha 3. Czech Republic.

e-mail: pelikan@vse.cz