Exact solution of the two-machine flow shop problem with 3 operations

Federico Della Croce[1], Fabio Salassa[1] and Vincent T'kindt[2]

[1] Politecnico di Torino, Italy
[federico.dellacroce,fabio.salassa]@polito.it
[2] University of Tours, Laboratory of Theoretical and Applied Computing (EA 6300), ERL CNRS 7002 ROOT, Tours, France
tkindt@univ-tours.fr

Keywords: Two-machine flow shop with three operations, ILP modeling, Exact approach.

1 Introduction

We consider a two-machine flow shop problem with three operations originally proposed in (Gupta et al. 2004). There is a set of n jobs being available at time zero to be processed on a two-machine flow-shop. Each job i has three operations, where the first operation has processing time a_i and must be performed on the first machine. The third operation has processing time b_i and must be performed on the second machine. Finally, the second operation has processing time c_i and can be performed either on machine 1 immediately after the first operation or on machine 2 immediately before the third operation. The operations of the same job cannot be processed concurrently, nor can any machine process more than one job at a time. We assume that preemption is not allowed, i.e., any operation once started must be completed without interruption. The goal is to minimize the makespan denoted by $C_{\max}$. As mentioned in (Gupta et al. 2004), this problem applies to several situations where a machine-independent setup operation is needed on each job between the two operations. The setup time is job-dependent and both machines are equipped with the required tooling for the setup. Then, the setup of an individual job is performed either while the job is still mounted on the first machine after the completion of the first operation or once the job is mounted on the second machine before the start of the second operation. The problem has strong similarities with the two-machine flow shop problem with common due date and jobs selection considered in (T'kindt et al. 2007) and (Della Croce et al. 2017). By using the extended three-field notation of (T'kindt, Billaut 2006) this latter problem is denoted by $F2|d_i = d$, *unknown* d $|\epsilon(d/n_T)$ where the number of jobs $n - n_T$ to be selected (here n_T is the number of tardy, hence discarded, jobs) is given in advance and the aim is to find the minimum value of d. For problem $F2|d_i = d$, *unknown* d $|\epsilon(d/n_T)$, the best available exact approach is able to solve very large size instances in limited CPU time (less than 30 seconds in the worst case for instances with $n = 100000$).

From every instance of the original 3-operation two-machine flow shop problem, it is possible to generate a special $F2|d_i = d$, *unknown* d $|\epsilon(d/n_T)$ problem as follows. Every job i of the original problem induces two "coupled" incompatible jobs i_1 and i_2 of the jobs selection problem where i_1 has the second operation of i assigned to the first machine, while i_2 has the second operation of i assigned to the second machine. Correspondingly, job i_1 has processing times $\alpha_{i_1} = a_i + c_i$ and $\beta_{i_1} = b_i$, while job i_2 has processing times $\alpha_{i_2} = a_i$ and $\beta_{i_2} = b_i + c_i$. Thus, we reduce to a two-machine flow shop problem with $2n$ jobs where exactly n compatible jobs out of the $2n$ jobs have to be selected.

2 ILP formulation

Consider the $2n$ jobs generated from the original problem as indicated above with processing times α_i (β_i), $1 \leq i \leq 2n$, on machine M_1 (M_2). When the set Ω of selected jobs is fixed, the minimization of the makespan for these jobs can be done in polynomial time by the so-called Johnson's algorithm (Johnson 1954): schedule first the jobs with $\alpha_i \leq \beta_i$ in non-decreasing order of α_i, followed by the jobs with $\alpha_i > \beta_i$ in non-increasing order of β_i. Without loss of generality, let us assume that the $2n$ jobs are indexed according to their position in the Johnson's schedule.

Let d be the unknown common due date, or equivalently the makespan of the selected jobs. Let us associate to each job i a binary variable x_i that indicates if job i is selected or not. A first ILP model is as follows.

$$\min d \tag{1}$$

$$\alpha_1 x_1 + \sum_{i=1}^{2n} \beta_i x_i \leq d \tag{2}$$

$$\sum_{i=1}^{2n} \alpha_i x_i + \beta_{2n} x_{2n} \leq d \tag{3}$$

$$\sum_{i=1}^{j} \alpha_i x_i + \sum_{i=j}^{2n} \beta_i x_i \leq d, \quad \forall j = 2, ..., 2n-1 \tag{4}$$

$$x_i + x_k = 1 \;\; \forall i, k \;\; \textit{incompatible} \tag{5}$$

$$x_i \in \{0, 1\} \;\; \forall i \in 1, ..., n \tag{6}$$

Here, constraints (2–4) are critical-path constraints which define the value of d. Notice that d is always determined by the sum of the processing times of jobs $1, .., j$ on the first machine plus the sum of the processing times of jobs $j, .., 2n$ on the second machine where j depends on the selected early jobs and therefore constraints (2–4) consider all possible values of j with $1 \leq j \leq 2n$. Notice that in the critical path constraints (2–4), we explicited constraint (2) corresponding to $j = 1$ and constraint (3) corresponding to $j = 2n$. Constraints (5) represent the incompatibility constraints between each pair of coupled jobs so that there will be exactly n early (selected) jobs. Finally, constraints (6) indicate that the x_i variables are binary.

Due to the presence of constraints (4) that generate $O(n^2)$ nonzeroes in the constraints matrix, the above model is limited in size as it induces an out-of-memory status of the solver if problems with several thousands of variables are considered.

As mentioned in (Della Croce et al. 2017), there exists an equivalent ILP formulation with $O(n)$ nonzeroes in the constraints matrix that can be obtained by introducing variables $y_j = \sum_{i=1}^{j} \alpha_i x_i + \sum_{i=j}^{2n} \beta_i x_i$ and constraints $y_i = y_{i-1} - \beta_{i-1} x_{i-1} + \alpha_i x_i, \;\; \forall i \in 2, .., 2n$.

$$\min d \tag{7}$$

$$y_1 = \alpha_1 x_1 + \sum_{i=1}^{2n} \beta_i x_i \tag{8}$$

$$y_{2n} = \sum_{i=1}^{2n} \alpha_i x_i + \beta_{2n} x_{2n} \tag{9}$$

$$y_i = y_{i-1} - \beta_{i-1} x_{i-1} + \alpha_i x_i \;\; \forall i = 2, ..., 2n-1 \tag{10}$$

$$y_i \leq d, \;\; \forall i = 1, ..., 2n \tag{11}$$

$$x_i + x_k = 1, \ \ \forall i, k \ \ incompatible \tag{12}$$

$$x_i \in \{0, 1\} \ \ \forall i \in 1, ..., n, \ \ y_i \geq 0 \ \ \forall i \in 1, ..., n \tag{13}$$

Let us denote by ILP_c the above ILP. Interestingly enough, the addition of the incompatibility constraints makes the problem much more difficult both for CPLEX 12.9 solver applied to ILP_c and to the constraint generation approach of (Della Croce et al. 2017) adapted in order to incorporate the incompatibility constraints. We tested both solution approaches on a Computer Intel i5 @1.6 GHz and 8 G of RAM. We considered a standard distribution of processing times with a_i, b_i and c_i uniformly distributed in the range [1...100] and tested 10 distinct instances for each problem size considering a CPU time limit of 60 seconds per instance. With this distribution, CPLEX 12.9 solver applied to model (7–13) already failed to solve to optimality one instance with 200 jobs, while the constraint generation approach of (Della Croce et al. 2017) was limited to 600 jobs runnning out of time on one instance with 700 jobs. We remark however that constraints (8–9) in ILP_c can be modified as follows where $\alpha_{[\min_2]}$ and $\beta_{[\min_2]}$ indicate the second smallest processing time on the first and the second machine respectively.

$$y_1 = \alpha_1 x_1 + \alpha_{[\min_2]}(1 - x_1) + \sum_{i=1}^{2n} \beta_i x_i \tag{14}$$

$$y_{2n} = \sum_{i=1}^{2n} \alpha_i x_i + \beta_{[\min_2]}(1 - x_{2n}) + \beta_{2n} x_{2n} \tag{15}$$

Indeed, if $x_1 = 1$, then constraint (14) coincides with constraint (8), while if $x_1 = 0$, then the critical path on the first selected job has processing time on the first machine not inferior to $a_{[\min_2]}$. Similar consideration holds with respect to constraint (15) taking into account the critical path on the last selected job and its processing time on the second machine. At the time of the conference we will also discuss how $a_{[\min_2]}$ and $b_{[\min_2]}$ can be increased without loss of optimality. In the reminder we denote by ILP_{ic} the improved ILP formulation that substitutes in ILP_c constraints (8–9) with constraints (14–15).

Hence, we can then successfully adapt to our problem the constraint generation approach proposed in (Della Croce et al. 2017) according to the scheme depicted in Algorithm 1. There, we denote by $F2^{3op}$ our problem formulated according to the ILP_{ic} model and by $F2^{3op}_{rel}$ its relaxation induced by the elimination of constraints (10) and considering constraints (11) only for $i = 1, 2n$.

Algorithm 1 Contraint Generation Algorithm

```
1: End=$False$
2: while !End do
3:     Solve $F2^{3op}_{rel}$: $\bar{x}$ is its solution and $OPT(F2^{3op}_{rel})$ its value
4:     Compute $d(\bar{x})$ the optimal value of the ILP of $F2^{3op}$ with added constraints $x = \bar{x}$
5:     if $(d(\bar{x}) = OPT(F2_{rel}))$ then
6:         End=$True$
7:     else
8:         Let $\mathcal{C}$ be the constraint giving $d(\bar{x})$ in the ILP of $F2^{3op}$ for $\bar{x}$
   //       ($\mathcal{C}$ is the most violated constraint)
9:         Add $\mathcal{C}$ to $F2^{3op}_{rel}$
10:    end if
11: end while
12: return $\bar{x}$ as the optimal solution of $F2^{3op}$
```

Algorithm 1 is a constraint generation approach solving initially problem $F2^{3op}_{rel}$ and then considering a separation procedure adding to the relaxation any inequality of the original formulation that is violated by the current solution. We tested both CPLEX 12.9 solver applied to model ILP_{ic} and Algorithm 1 on instances generated according to the same distribution considered above (10 instances for each problem size). The related results are depicted in Table 1 where it is shown that Algorithm 1 is clearly superior and solves within 60 seconds instances with up $n = 30000$. Detailed computational results on several other different distributions will be presented at the conference.

n	CPLEX		Algorithm 1	
	t_{avg}	t_{max}	t_{avg}	t_{max}
1000	0.6	1	0.1	1
4000	9.3	13	0.6	1
7000	24.1	36	1.2	2
10000	CPU limit		3.7	5
15000	CPU limit		6.2	9
20000	CPU limit		10.1	13
25000	CPU limit		24.3	35
30000	CPU limit		25.9	43
35000	CPU limit		CPU limit	

Table 1. Comparing CPLEX applied to model ILP_{ic} and Algorithm 1

Acknowledgements

This work has been partially supported by "Ministero dell'Istruzione, dell'Università e della Ricerca" Award "TESUN-83486178370409 finanziamento dipartimenti di eccellenza CAP. 1694 TIT. 232 ART. 6".

References

Della Croce, F., Koulamas, C., T'kindt, V.: A constraint generation approach for two-machine shop problems with jobs selection. *European Journal of Operational Research*, 259: 3, 898–905 (2017).

Gupta, J.N.D., Koulamas, C.P., Kyparisis, G.J., Potts,C.N.,Strusevich, V.A.: Scheduling Three-Operation Jobs in a Two-Machine Flow Shop to Minimize Makespan. *Annals of Operations Research*, 129: 1-4, 171–185 (2004).

Johnson, S.: Optimal two and three stage production schedules with set-up time included. *Naval Research Logistics Quarterly*, 1, 61–68 (1954).

Tkindt, V. , Billaut, J.-C.: Multicriteria scheduling: Theory, models and algorithms (2nd edition). Heidelberg, Germany: Springer-Verlag (2006).

T'kindt, V., Della Croce, F., Bouquard, J.L.: Enumeration of Pareto Optima for a Flowshop Scheduling Problem with Two Criteria. *INFORMS Journal on Computing*, 19: 1, 64–72 (2007)