Logic-based Benders Decomposition for preemptive Flexible Job-Shop Scheduling

Carla Juvin, Laurent Houssin, Pierre Lopez

Logic-based Benders Decomposition for preemptive Flexible Job-Shop Scheduling

Carla Juvin[1], Laurent Houssin[1,2], and Pierre Lopez[1]

[1] LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France
{carla.juvin,pierre.lopez}@laas.fr
[2] ISAE-SUPAERO, Université de Toulouse, France
laurent.houssin@isae-supaero.fr

Keywords: Flexible job-shop scheduling, preemption, Benders decomposition.

1 Introduction

In this paper, we address the preemptive flexible job-shop scheduling problem (pFJSSP). We propose three exact methods to solve the pFJSSP with makespan minimization objective: a Mixed Integer Linear Programming (MILP) formulation, a Constraint Programming (CP) formulation, and a Logic-Based Benders Decomposition (LBBD) algorithm.

The job-shop scheduling problem (JSSP) is one of the most studied NP-hard optimization problems. It consists of a set of jobs and a set of machines. Each job has a sequence of operations, each of which must be performed on a given machine. In order to satisfy present market, the production environment becomes more and more complex and flexible. The flexible job-shop problem(FJSSP) is an extension of the classical JSSP that allows an operation to be processed on any machine from a set of eligible machines.

The FJSSP has received considerable attention and both metaheuristics and exact methods have been developed to solve this problem (Brandimarte (1993), Shen *et al.* (2018)). Moreover, preemption is another important parameter when dealing with scheduling problems as it can have a positive impact on the objective function. The preemptive job-shop scheduling problem (pJSSP) has received limited attention and most of literature focus on approximation algorithms rather than exact methods. Among the exact methods, Le Pape and Baptiste (1998) develop a constraint programming approach, Ebadi and Moslehi (2013) model the pJSSP with a disjunctive graph and develop a branch-and-bound algorithm. To the best of our knowledge, only one study (Zhang and Yang (2016)) considers both preemption and flexibility for the JSSP. However, the problem addressed in this work has special characteristics (flexible workdays and overlapping in operations) that are not considered here.

Problem statement. An instance of the pFJSSP is defined as a set of n jobs $\mathcal{J} = \{1, \dots, n\}$ and a set of machines $\mathcal{M}$. Each job i consists of a sequence of n_i operations $\mathcal{O}_i = (i_1, \dots, i_{n_i})$. An operation $O_{i,j} \in \mathcal{O}_i$ of a job i must be performed by one of the machine from the set of eligible machines $\mathcal{M}_{i,j} \subseteq \mathcal{M}$. Let $p_{i,j,m}$ denote the processing time of operation $O_{i,j}$ that is processed on machine $m \in \mathcal{M}_{i,j}$. Each machine can process at most one operation at a time and preemption is allowed: the processing of operations can be interrupted and resumed later without penalty. Although an operation can be interrupted, it is assumed that it must be fully processed by one and the same machine.

2 Mathematical and Constraint programming models

Let us introduce a model that can be used as a mixed-integer program to solve the pFJSSP. It is based on a time-indexed formulation proposed by Bowman (1959) to solve the

preemptive job-shop problem. We have adapted it to add the notion of resource flexibility. Let $\mathcal{H} = \{1, 2, 3, \ldots, h\}$ be the time horizon. The (binary) decision variables are defined as follows:

- $x_{i,j,m}$ is equal to 1 if operation $O_{i,j}$ is processed on machine m;
- $y_{i,j,t}$ is equal to 1 if operation $O_{i,j}$ is in process at time t,

and the model:

$$\min C_{\max} \tag{1}$$

$$s.t. \quad \sum_{m \in \mathcal{M}_{i,j}} x_{i,j,m} = 1 \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \tag{2}$$

$$\sum_{t=1}^{h} y_{i,j,t} \geq \sum_{m \in \mathcal{M}_{i,j}} x_{i,j,m} \times p_{i,j,m} \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \tag{3}$$

$$\sum_{t'=t}^{h} y_{i,j,t'} \leq \max_{m \in \mathcal{M}_{i,j}} p_{i,j,m} \times (1 - y_{i,j+1,t}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \setminus \{O_{i,n_i}\},\, t \in \mathcal{H} \tag{4}$$

$$\sum_{i \in \mathcal{J}} \sum_{j=1}^{n_i} x_{i,j,m} \times y_{i,j,t} \leq 1 \quad \forall m \in \mathcal{M},\, t \in \mathcal{H} \tag{5}$$

$$C_{\max} \geq (t+1) \times \sum_{i \in \mathcal{J}} y_{i,n_i,t} \quad \forall t \in \mathcal{H} \tag{6}$$

$$x_{i,j,m} \in \{0, 1\} \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i,\, m \in \mathcal{M}_{i,j} \tag{7}$$

$$y_{i,j,t} \in \{0, 1\} \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i,\, t \in \mathcal{H} \tag{8}$$

Note that Constraints (5) are nonlinear, but can be easily linearized since variables $x_{i,j,m}$ and $y_{i,j,t}$ are binary, the previous mathematical model thus becoming a MILP model.

Moreover, we propose another formulation of the problem using Constraint Programming. It is based on a formulation proposed by Polo-Mejía *et al.* (2020) to solve the Multi-Skill Project Scheduling Problem with partial preemption. We have adapted it to solve the pFJSSP. We introduce the following decision variables: $I_{i,j}$ represents the interval variable between the start and the end of the processing of operation $O_{i,j}$. Since activities can be performed in multiple machines, we introduce an optional interval variable $mode_{i,j,m}$ for each possible combination of an operation $O_{i,j}$ and a machine m. Each operation $O_{i,j}$ is divided into $p_{i,j,m}$ parts of unit duration, the optional variable $part_{i,j,k,m}$ representing the interval during which the k^{th} part of the operation $O_{i,j}$ is processed if it is executed on machine m.

$$\min C_{\max} \tag{9}$$

$$s.t. \quad C_{\max} \geq I_{i,n_i}.end \quad \forall i \in \mathcal{J} \tag{10}$$

$$endBeforeStart(I_{i,j}, I_{i,j+1}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \setminus \{i_{n_i}\} \tag{11}$$

$$endBeforeStart(part_{i,j,k,m}, part_{i,j,k+1,m}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i,\, m \in \mathcal{M}_{i,j},\, k \in 1, \ldots, p_{i,j,m} - 1 \tag{12}$$

$$span(mode_{i,j,m}, part_{i,j,k,m} : \forall k \in 1, \ldots, p_{i,j,m}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i,\, m \in \mathcal{M}_{i,j} \tag{13}$$

$$alternative(I_{i,j}, mode_{i,j,m} : \forall m \in \mathcal{M}_{i,j}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \tag{14}$$

$$presenceOf(mode_{i,j,m}) \Rightarrow presenceOf(part_{i,j,k,m}) \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i,\, m \in \mathcal{M}_{i,j},\, k \in 1, \ldots, p_{i,j,m} \tag{15}$$

$$noOverlap(part_{i,j,k,m} : \forall i \in \mathcal{J},\, j \in 1, \ldots, n_i,\, k \in 1, \ldots, p_{i,j,m}) \quad \forall m \in \mathcal{M} \tag{16}$$

3 Logic-based Benders decomposition

A logic-based Benders decomposition approach (Hooker (2007)) is proposed to solve the problem under consideration. It consists in dividing the problem into a machine assignment master problem and a preemptive job-shop scheduling subproblem.

The master problem is an assignment problem, each operation needs to be assigned to a machine. The decision variable $x_{i,j,m}$ is equal to 1 if operation $O_{i,j}$ is processed on machine m. We propose the following MILP model:

$$\min C_{\max} \tag{17}$$

$$\sum_{m \in M} x_{i,j,m} = 1 \quad \forall i \in \mathcal{J},\, O_{i,j} \in \mathcal{O}_i \tag{18}$$

$$\text{Benders' cuts} \tag{19}$$

$$x_{i,j,m} \in \{0,1\} \tag{20}$$

However, at first iteration as there is no Benders cut, there is no link between assignment variables $x_{i,j,m}$ and the objective value $C_{\max}$. That is why we propose to include a subproblem relaxation in the master problem. The subproblem relaxation is based on the following three ideas:

1. The makespan is at least equal to the sum of the processing times of the operations of the same job.
2. The makespan is at least equal to the sum of the processing times of the operations assigned to the same machine.
3. Let consider a subset of operations assigned to the same machine. The makespan is at least equal to the following quantity: the minimum release date of the subset plus the sum of the processing times of the operations of the subset plus the minimum delivery time of the job of the last processed operation of this subset.

At each iteration we obtain a feasible solution $P_h = \{(i,j,m) \,|\, x_{i,j,m} = 1\}$ of the master problem which contains all assignments. Then, the subproblem consists in a pJSSP, which can be solved by two different methods:

- a CP model similar to the one proposed for the pFJSSP, but for which the set of eligible machines $M_{i,j}$ for each operation $O_{i,j}$ is reduced to the machine m assigned by the master problem $(i,j,m) \in P^h$;
- a Branch-and-Bound algorithm proposed by Ebadi and Moslehi (2013).

When the subproblem is solved, we obtain the optimal makespan $C^h_{\max}$ for a given assignment P^h. We can deduce the following cut:

$$C_{\max} \geq C^h_{\max}\Big(1 - \sum_{(i,j,m) \in P^h} (1 - x_{i,j,m})\Big) \tag{21}$$

At each iteration, this cut is added to the master problem as a Benders cut, and the master problem is resolved with this new constraint.

4 Numerical results and Conclusions

For computational tests, we use CPLEX for solving the MILP model and CP Optimizer for the CP model. The computation time was limited to 10 minutes. We use classical instances for the FJSSP[3]. Table 1 shows the results. The first column shows the benchmarks

Table 1. Number of pFJSSP instances proved to be optimal within 10 min CPU

Benchmark	MILP	CP	LBBD (sp:CP)	LBBD (sp:B&B)
Brandimarte (15)	0	5	9	9
Hurink edata (66)	1	2	9	24
Hurink rdata (66)	1	1	1	20
Hurink vdata (66)	1	2	7	25
DPPaulli (18)	0	0	0	0
ChambersBarnes (21)	0	0	0	0
Kacem (4)	3	3	4	4
Fattahi (20)	8	12	13	17

under study (and the number of instances they contain), the following ones the number of instances solved to optimality using the methods described in the previous sections.

According to our computational experiments, CP method is superior to the MILP one and solves more instances to optimality. On the other hand, we can see that the pFJSSP benefits from the decomposition since LBBD methods are more efficient compared to the others. More specifically, solving the subproblem using Branch-and-Bound (fifth column) is faster than using CP (fourth column) and thus enables to solve a greater number of instances. We also notice that for the most difficult benchmarks (DPPaulli and Chambers-Barnes) no instance could be solved optimally by any of these methods within 10 minutes.

Conclusion. We have proposed three methods to solve the pFJSSP where the objective is the minimisation of the makespan: a mathematical model, a constraint programming model, and a logic-based Benders decomposition algorithm. Numerical results have shown that the MILP becomes inefficient for difficult instances. Also, our logic-based Benders decomposition outperforms mathematical programming and constraint programming to find optimal solutions. Our current work consists in improving the solving methods, in particular the decomposition ones to further increase the number of solved instances.

References

Brandimarte P., 1993, "Routing and scheduling in a flexible job shop by tabu search", *Annals of Operations Research*, Vol. 41, pp. 157-183.

Bowman E.H., 1959, "The schedule-sequencing problem", *Operations Research*, Vol. 07, pp. 621-624.

Ebadi A. and Moslehi G., 2013, "An optimal method for the preemptive job shop scheduling problem", *Computers & Operations Research*, Vol. 40, pp. 1314-1327.

Hooker J., 2007, "Planning and scheduling by logic-based Benders decomposition", *Operations Research*, Vol. 55, pp. 588-602.

Le Pape C. and Baptiste P., 1998, "Resource constraints for preemptive job-shop scheduling", *Constraints: An international journal*, Vol. 03, pp. 263-287.

Polo-Mejía O., Artigues C., Lopez P., and Basini V., 2020, "Mixed-integer/linear and constraint programming approaches for activity scheduling in a nuclear research facility", *International Journal of Production Research*, Vol. 58, pp. 7149-7166.

Shen L., Dauzère-Pérès S. and Neufeld J. S., 2018, "Solving the flexible job shop scheduling problem with sequence-dependent setup times", *European Journal of Operational Research*, Vol. 265, pp. 503-516.

Zhang J. and Yang, J. 2016, "Flexible job-shop scheduling with flexible workdays, preemption, overlapping in operations and satisfaction criteria: an industrial application", *International Journal of Production Research*, Vol. 54, pp. 4894-4918.

[3] `http://opus.ub.hsu-hh.de/volltexte/2012/2982/` – Last accessed October 2021