Comparing MIP and CP for the Job Shop Scheduling Problem

Julian Nunez Nova, Kirill Savin, Tanzim Hossain

Faculty of Engineering, University of Porto (FEUP)

Abstract. The Job Shop Scheduling Problem (JSP) is a classical NP-hard optimization problem with significant relevance in manufacturing and production planning. In this work, we study and compare two exact optimization approaches for solving the JSP: Mixed-Integer Programming (MIP) and Constraint Programming (CP). Both models are implemented using IBM ILOG CPLEX Studio and evaluated on benchmark instances from the Fisher and Thompson dataset. We analyze solution quality, resolution time, and model scalability, highlighting the strengths and limitations of each approach.

Keywords: Mixed-Integer Programming · Constraint Programming

1 Introduction

The Job Shop Scheduling Problem (JSP) consists of scheduling a set of jobs, each composed of a sequence of operations, on a set of machines. Each operation must be processed on a specific machine for a fixed duration, and machines can handle at most one operation at a time. The objective is typically to minimize the makespan, defined as the completion time of the last operation.

Due to its combinatorial nature, the JSP is known to be NP-hard. Several exact and heuristic approaches have been proposed, among which Mixed-Integer Programming (MIP) and Constraint Programming (CP) are widely used. This paper aims to compare these two modeling paradigms in terms of solution quality and computational performance.

1.1 Mixed-Integer Programming (MIP) Model

We model the Job Shop Scheduling Problem (JSP) with start-time variables and Big-M disjunctive constraints to enforce machine capacity. The objective is to minimize the makespan.

Sets and indices. $J = \{1, \ldots, n\}$ (jobs), $K = \{1, \ldots, m\}$ (operations per job, m machines total).

Parameters. $M_{j,k} \in \{1, \ldots, m\}$ is the machine required by operation (j, k); $p_{j,k} \in \mathbb{Z}_{\geq 0}$ is its processing time. A valid Big-M is:

$$M = \sum_{j \in J} \sum_{k \in K} p_{j,k}.$$

Conflict set (operations on same machine, different jobs).

$$\mathcal{P} = \Big\{ (j_1, k_1, j_2, k_2) \;\Big|\; j_1 < j_2,\ M_{j_1,k_1} = M_{j_2,k_2} \Big\}.$$

Decision variables. $t_{j,k} \in \mathbb{Z}_{\geq 0}$ start time of operation (j,k), $C_{\max} \in \mathbb{Z}_{\geq 0}$ makespan, $y_{j_1,k_1,j_2,k_2} \in \{0,1\}$ for each $(j_1,k_1,j_2,k_2) \in \mathcal{P}$, where $y_{j_1,k_1,j_2,k_2} = 1$ means (j_1,k_1) precedes (j_2,k_2).

MIP model:

$$\begin{aligned}
\min \quad & C_{\max} && && (1)\\
\text{s.t.} \quad & t_{j,k+1} \geq t_{j,k} + p_{j,k} && \forall j \in J,\ k = 1, \ldots, m-1 && (2)\\
& t_{j_1,k_1} + p_{j_1,k_1} \leq t_{j_2,k_2} + M(1 - y_{j_1,k_1,j_2,k_2}) && \forall (j_1,k_1,j_2,k_2) \in \mathcal{P} && (3)\\
& t_{j_2,k_2} + p_{j_2,k_2} \leq t_{j_1,k_1} + M y_{j_1,k_1,j_2,k_2} && \forall (j_1,k_1,j_2,k_2) \in \mathcal{P} && (4)\\
& C_{\max} \geq t_{j,m} + p_{j,m} && \forall j \in J && (5)\\
& t_{j,k} \in \mathbb{Z}_{\geq 0} && \forall j \in J,\ k \in K && (6)\\
& y_{j_1,k_1,j_2,k_2} \in \{0,1\} && \forall (j_1,k_1,j_2,k_2) \in \mathcal{P}, && (7)\\
& C_{\max} \in \mathbb{Z}_{\geq 0}. && && (8)
\end{aligned}$$

2 Constraint Programming Model

The Job Shop Scheduling Problem is also formulated using Constraint Programming (CP), where operations are modeled as interval variables with fixed durations.

Sets and indices. Let

$$J = \{1, \ldots, n\}, \qquad K = \{1, \ldots, m\}, \qquad \mathcal{M} = \{1, \ldots, m\}.$$

Parameters.

$$M_{j,k} \in \mathcal{M}, \qquad p_{j,k} \in \mathbb{Z}_{\geq 0}, \quad \forall j \in J,\ k \in K.$$

Decision variables.

$$O_{j,k} \text{ is an interval variable of fixed size } p_{j,k}, \quad \forall j \in J,\ k \in K,$$

$$C_{\max} \in \mathbb{Z}_{\geq 0}.$$

CP model.

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

$$\text{s.t.} \quad \text{endBeforeStart}(O_{j,k},\, O_{j,k+1}) \qquad \forall j \in J,\ k = 1, \ldots, m-1 \tag{10}$$

$$\text{noOverlap}(\{O_{j,k} \mid j \in J,\ k \in K,\ M_{j,k} = \mu\}) \quad \forall \mu \in \mathcal{M} \tag{11}$$

$$C_{\max} \geq \text{endOf}(O_{j,m}) \qquad \forall j \in J. \tag{12}$$

3 Experimental Results

3.1 Test Instances and Limitations

Experiments were conducted using benchmark instances from the JSPLIB GitHub repository [1], which contains the Fisher and Thompson job-shop scheduling dataset. The `ft06` instance was specifically used to evaluate both MIP and CP formulations.

The MIP model was solved for `ft06` but failed on `ft10` due to CPLEX Community Edition size limits. The Big-M MIP model introduces binary sequencing variables and disjunctive constraints for each pair of conflicting operations, causing the model size to grow rapidly with the instance size.

In contrast, the CP formulation remains solvable for `ft06` thanks to its use of interval variables and global constraints `noOverlap`, which provide a more compact representation and better scalability under limited solver resources.

3.2 Key Performance Indicators

The following key performance indicators (KPIs) were analyzed:

- Optimal makespan
- Resolution time
- Time to first feasible solution
- Model size and search effort

3.3 Comparison Between MIP and CP

Table 1: Performance comparison for instance `ft06`

Metric	MIP	CP
Makespan	61	61
Solve time (s)	0.23	0.07
Time to first feasible (s)	0.11	0.06
Nodes / Branches	184	140,983

Both MIP and CP reach the optimal makespan for the `ft06` instance, confirming equivalent solution quality; however, their performance metrics reveal distinct ways of handling the problem's combinatorial complexity. The CP model finds feasible solutions faster and achieves shorter overall solve times due to strong constraint propagation enabled by global constraints such as `noOverlap`, which efficiently prune infeasible schedules early in the search. Although CP explores a much larger number of branches, each branching decision is computationally inexpensive, allowing rapid traversal of the search space. In contrast, the MIP formulation explores far fewer nodes, but each node requires solving a linear relaxation, resulting in higher computational cost per decision and longer solve times. These results indicate that the complexity of the Job Shop Scheduling Problem is strongly influenced by the modeling approach, with CP distributing complexity across many lightweight decisions and MIP concentrating it into fewer but more expensive optimization steps.

3.4 Schedule Visualization

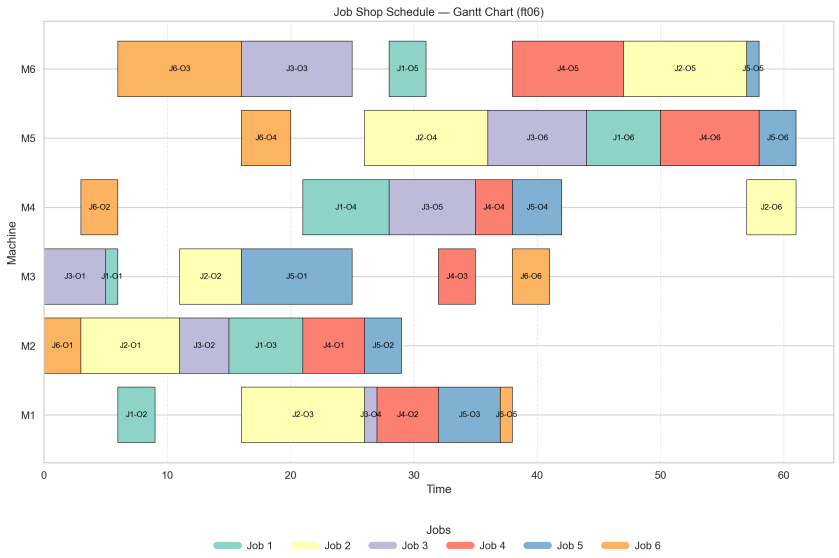

Fig. 1: Gantt chart for the `ft06` instance obtained with the CP model

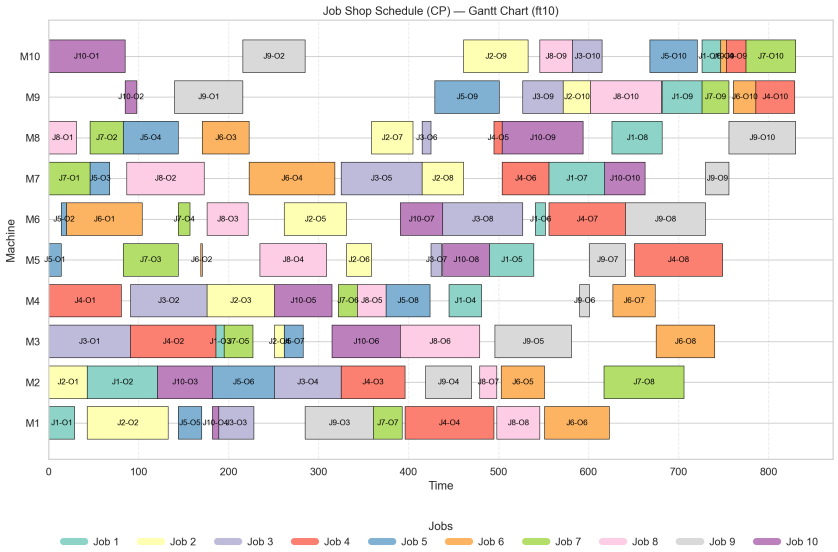

Fig. 2: Gantt chart for the `ft10` instance obtained with the CP model

The Gantt chart illustrates the execution of operations across machines over time, confirming that no machine conflicts occur and that all job precedence constraints are respected.

3.5 KPI Analysis

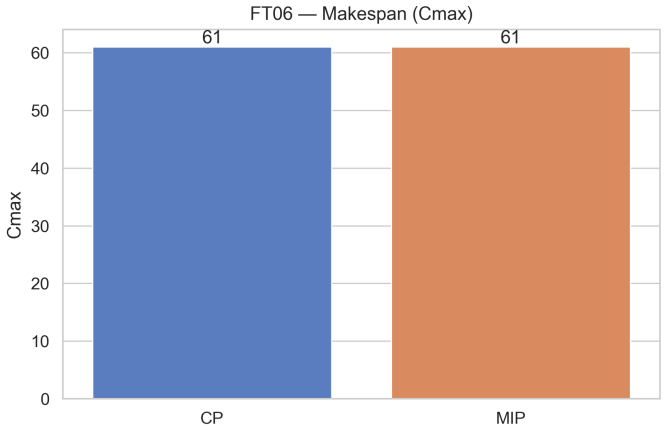

Fig. 3: Makespan comparison for the `ft06` instance

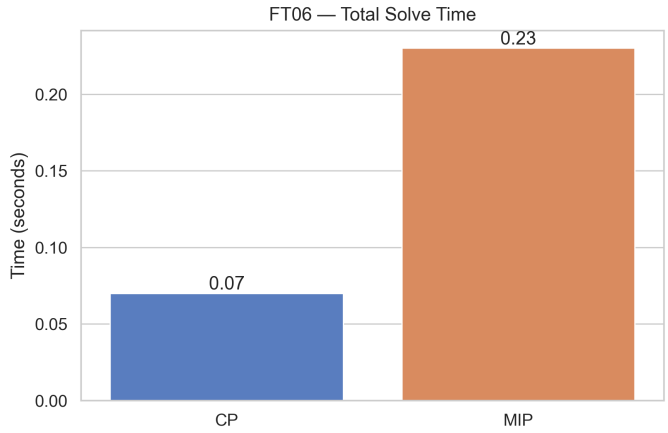

Fig. 4: Solve time comparison for the `ft06` instance

Fig. 5: Model size and search effort comparison for the `ft06` instance

The KPI plots further highlight the performance differences between the two approaches. While both formulations achieve identical solution quality, the CP model benefits from stronger constraint propagation, resulting in faster convergence despite a larger search space.

4 Conclusions

Both MIP and CP successfully obtained optimal solutions for the tested job-shop scheduling instances. Key observations from our comparative analysis include:

– Both MIP and CP achieve optimal solutions for small instances such as `ft06`.

- CP provides faster feasibility detection and solution times due to stronger constraint propagation.
- MIP explores fewer search nodes but incurs higher computational cost per node due to linear programming relaxations.
- CP scales more effectively under limited solver resources (e.g., CPLEX Community Edition).
- The modeling paradigm choice significantly impacts practical problem complexity and solver performance.

While CP demonstrated superior performance in terms of resolution time and modeling simplicity—leveraging global scheduling constraints such as `noOverlap`—the MIP formulation remains valuable for its flexibility and compatibility with a wide range of optimization techniques.

5 Future Work

Future research will extend the experimental evaluation to larger and more diverse benchmark instances to further assess the scalability of MIP and CP formulations. Several promising directions include:

- **Alternative MIP formulations:** Investigating time-indexed and flow-based models, along with tighter linearization techniques, to reduce model size and improve computational performance.
- **Hybrid MIP/CP approaches:** Combining CP's rapid feasibility search with MIP's exact optimization capabilities, using CP to generate high-quality initial solutions and MIP to prove optimality.
- **Enhanced solver tuning:** Evaluating solver behavior under varied parameter settings and licensed configurations to better understand trade-offs between modeling expressiveness, computational efficiency, and solution quality.
- **Larger instance benchmarks:** Extending tests to more challenging instances beyond `ft06` and `ft10` to examine how both methods scale with increasing problem size.

These directions aim to bridge the gaps identified in this study and provide more robust, scalable solutions for complex scheduling problems in both academic and industrial settings.

References

1. Tamaki, H., et al.: JSPLIB: Benchmark instances for job-shop scheduling problems. GitHub repository: https://github.com/tamy0612/JSPLIB
2. IBM ILOG CP Optimizer User's Manual. IBM (2024)
3. IBM ILOG CPLEX Optimization Studio User's Manual. IBM (2024)