Multi-Ressource Scheduling with Setup Times: An Application Case to the Textile Industry*

Alexandre Mercier-Aubin[1], Claude-Guy Quimper[1], and Jonathan Gaudreault[1]

Université Laval, Québec, QC, G1V 0A6, Canada

Abstract. In the textile industry, the looms are now automatic, but what is left to be automated is the scheduling process. This paper is about the scheduling of the looms and workers doing the setup between two jobs. We explain the problem, tools, methodologies, and constraints to solve this NP-Hard problem. As of now, this is a work in progress.

Keywords: Looms · Scheduling · Constraint Programming · Textile

1 Introduction

1.1 Problem to Solve

The company financing this project is a commercial textile manufacturer. In one of their factories, they own nearly 90 automated looms. Each loom has a set of possible configurations and a different speed for each specific configuration. There are approximately 800 tasks (or jobs) to assign to the looms on a horizon of two weeks, each having different compatibility with the loom configurations. Once the tasks are matched to a loom and a configuration, they need to be scheduled. Each task has a priority and a due date. The objective function of the scheduling will be to maximize the total tardiness weighted by the priority of each task. The higher a priority is, the more penalized it will be for being late. At the moment, this is done manually by the staff. Note that this scheduling problem is quite similar to the NP-Complete resource constraint scheduling problem with the addition of setup times [1]. The objective of the master's degree related to this paper is to automate the process of scheduling. Techniques such as linear programming (LP) and constraint programming (CP) will be used to try and tame the issue.

The whole project splits into three sections: planning, scheduling, and simulation. These topics will be detailed in the next sections, but the focus is set on the scheduling since this is the topic of my master thesis. Other parts will be done with collaborators.

The related master's degree has begun on July 8th, 2019, therefore no significant discovery will be presented at this point. This is mostly an overview of the project and what has been done so far.

* Supported by Mitacs

1.2 Tools

As of now, empirical testing has been done on the performances of different techniques for scheduling. According to Wen-Yang Ku et al.[4], CP offers great performances for scheduling problems.

One of the available tools for modeling mathematical problems is MiniZinc [2]. For a given model, multiple solvers can be used. This gives us the ability to test LP and CP without hassle. Yet, knowing that CP is empirically better, it keeps the ability to focus on it by using solver specific features. The point here is that the planning part and scheduling part of this project can be modeled in the same language and use the forces of both LP and CP according to which performs better.

2 Methodology

2.1 Planning

In the planning phase, we assign all tasks in T to a loom in L. Multiple tasks can be assigned to a loom. A loom $l \in L$ can take different configurations in C_l. Should task i be assigned to loom l, the loom must take the configuration $c_{il} \in C_l$. A loom is allowed to change configurations only once during the scheduling horizon. We call this configuration change a *major setup*. The configuration before the major setup is c_l. The planning model decides what will be the new configuration c'_l of each loom. The choice induces a setup time D_l that is function of c_l and c'_l. The next two capacities are also considered. The loom capacity Q_l is the amount of time a loom can be used on the horizon. The human resource capacity Q_r is the amount of time an engineer r can spend for all major setups. There are minor setups between each task on a loom, but these minor setups are omitted and are rather delegated to the scheduler.

2.2 Scheduling

Once the planning phase completed, the task i is assigned to loom $A[i]$. We also know which task must be executed before or after the major setup. It is imperative to set the order of the tasks. A minor setup is a "short" setup that might last between 5 minutes to 3 hours. These setups do not change the configuration of the looms. When there is no major setup between two jobs, there must be a minor setup. In our model, the variable $N[i]$ represents the task that executes next to i. This information is necessary to compute the minor setup times. The duration of a minor setup is function of the task i and the following task $N[i]$. A minor setup is decomposed into an ordered sequence of $|P|$ multiple tasks each executed by a person of a profession in P. There is one setup task per profession in P and the tasks are ordered such they are executed by professions $1, 2, \ldots, |P|$ in that order. The duration of these tasks is given by the parameters $d_{i,N[i],p}$ for $p \in P$. There are Q_p people of profession p. Each profession can be seen as a cumulative resource of capacity Q_p. The schedule needs to ensure that not too

many minor setups occur simultaneously in order not to overload the human resources. A different profession and resources are used for the major setups.

A task i has an earliest starting time est_i, a latest completion time lct_i, and a duration d_i. The task i can only execute, without interruption, within the time window $[\text{est}_i, \text{lct}_i)$. Let S_i be the starting times of task i, $S_{i,p}$ be the starting times of the setup task following task i executed by profession p. The duration of this task depends on the next task. The time required for profession p to execute its setup task from task i to task j is given by the parameter $d_{i,j,p}$.

The scheduler is used every week over a horizon $t_{\max}$ equal to two weeks. Some tasks, already scheduled from the last execution of the solver, are left untouched. This means that the parameters est_i and lct_i form a tight execution window for these tasks.

A task i has a due date e_i and a priority r_i. A task is allowed to miss its due date at a cost of r_i for every unit of time beyond this due date. The goal is to minimize the total tardiness: $\sum_i \max(0, S_i + p_i - e_i)$.

2.3 Simulation

The main goal of the simulation is to evaluate the performance of the generated schedules in a context with hazards such as breakdowns. The evaluated criteria are the setup times, job operation times, how does the plan cope with breakdowns, etc. Every loom and human resource is modeled and scaled to the production chain. This way, the workers' movements are accounted for.

This simulation model will be used to test the planning/scheduling pipeline. It will also be used by the employees to predict the impact of a strategic decision on the lines of production.

Since the project only started recently, the scheduling model presented in the next section does not take into account the simulation nor the stochastic aspects of problem.

3 Scheduling Model

3.1 Constraints

Two constraints are particularly useful to model our scheduling problems: the Cumulative constraint and the Disjunctive constraint. The Cumulative constraint [5] is useful when for every time points, the resource usage has to be less than or equal to the resource capacity. For instance, it will ensure that no more than Q_p setup tasks are simultaneously executed by people of profession p. This constraint comes with efficient filtering algorithms [6].

The Disjunctive constraint is a specialization of the Cumulative constraint with a unit resource capacity. It prevents two tasks to overlap in time. The Disjunctive can constrain the tasks assigned to a loom l to execute at distinct time. To reduce the number of tasks subject to the Disjunctive constraint, a task and its setup tasks are merged into one longer task.

The constraint Circuit$(N[1], \ldots, N[n])$ is satisfied if the sequence $N[1]$, $N[N[1]]$, $N[N[N[1]]]$, ... contains all integers between 1 and n. Consider a graph with vertices $1..n$ and arcs $\{(i, j) \mid j \in \text{dom}(N[i])\}$. The vector N encodes a Hamiltonian circuit. To turn our problem into a circuit, we add a dummy task to N that follows the last task of each loom but precedes its first task.

Recall that the loom l has for initial configuration c_l, the tasks that require this initial configuration must execute before the major setup that occurs at time S_l.

$$\forall l \in L, \forall i \in \{i \mid A[i] = l \wedge C_{il} = c_l\} \ : \ S_i \leq S_l. \tag{1}$$

The tasks that do not require the initial configuration require the second configuration and must execute after the major setup whose duration is D_l.

$$\forall l \in L, \forall i \in \{i \mid A[i] = l \wedge C_{il} \neq c_l\} \ : \ S_i \geq S_l + D_l. \tag{2}$$

The minor setup tasks are executed in order of profession and all setup tasks must be executed between a pair of tasks i et N_i.

$$\forall i \in T, \forall p \in P \ : \ D_{i,p} = d_{i,N_i,p} \tag{3}$$

$$\forall i \in T, \forall p \in 1..|P| - 1 \ : \ S_{i,p} + D_{i,p} \leq S_{i,p+1} \tag{4}$$

$$\forall i \in T \ : \ S_{i,|P|} + D_{i,|P|} \leq S_{N[i]} \tag{5}$$

3.2 Objective function and branching heuristic

The objective function is to minimize the total tardiness weighted by priority.

$$\min \sum_{i \in T} R_i \cdot \max(0, E_i - (S_i + D_i)) \tag{6}$$

To accelerate the process of finding a solution, MiniZinc allows the use of heuristics. Coupled with the Chuffed[7] specific priority search [3], it renders possible the ability to guide the solver with a search value different than the branching value.

The second heuristic is a bit more complex. First, we solve small TSP instances using a TSP solver, one for each loom. The cities are the tasks and the distances the setup times. Therefore, the optimal circuit minimizes the setup times. The resulting circuits are sent to MiniZinc and used as an initial solution for local search that, at each iteration, reassign 30% of the solution.

4 Conclusion

The model is still under development and it would be premature to publish any result. We hope to obtain good solutions, i.e. solutions better than the ones produced by a human.

Once we obtain a working model, we will evaluate the robustness of our solution by using a simulator. From there, we hope to come up with a second version of the model that produces solutions robust to breakdowns. It might also be interesting to investigate the constraints unavailable in MiniZinc such as the WeightedCircuit. The transitions between setups can be represented as a weighted graph, therefore making this constraint quite valuable.

References

1. Garey, M.R., Johnson, D.S.: Computers and intractability: A guide to the theory of NP-Completeness. First Canadian Edition edition. W H Freeman & Co, Canada (1979)
2. MiniZinc Handbook, https://www.minizinc.org/doc-2.3.0/en/index.html. Last accessed 7 Jul 2019
3. Feydy, T., Goldwaser, A., Schutt, A., Stuckey, P.J., Young, K.D.: Priority search with MiniZinc, Data61, CSIRO
4. Ku, W.Y., Beck, C. J.: Mixed integer programming models for job shop scheduling: A computational analysis, Computers & Operations Research, **73**, 165–173 (2016)
5. Aggoun, A., Beldiceanu, N.: Extending chip in order to solve complex scheduling and placement problems, Mathematical and computer modelling, **17**,57–73, 1993.
6. Schutt, A., Feydy, T., Stuckey, P.J.: Explaining Time-Table-Edge-Finding Propagation for the Cumulative Resource, CPAIOR, **16**,234-250, 2013.
7. Chu, G. G. (2011). Improving combinatorial optimization. PhD thesis, Engineering, Computer Science and Software Engineering, The University of Melbourne.