Solving the Aircraft Disassembly Scheduling Problem

Charles Thomas[a,1], Pierre Schaus[a]

[a] *UCLouvain, Place Sainte Barbe 2/L5.02.01, Louvain-la-Neuve, 1348, Belgium*

Abstract

Dismantling aircrafts reaching their end of life is a complex endeavour that is necessary in terms of sustainability but yields small income margins for air transport companies. An efficient scheduling of the disassembly procedure is thus crucial to ensure the profitability of the process and incentivize practice. This is a large scheduling problem that involves thousands of tasks and many different constraints: Extracting parts that are destined to be reused requires technicians with specific certifications and equipment. Extraction operations might be subject to precedence relations. Furthermore, the aircraft must be kept balanced during the whole process. Finally, some of the locations of the aircraft have a limited space that caps the number of technicians able to work there concurrently. This article presents the problem in details and proposes two approaches to solve the problem: a Constraint Programming model and a MIP model. The models are tested on instances of varying sizes involving up to 1450 tasks which are based on real operational data provided by an industrial partner.

Keywords: Scheduling, Aircraft Disassembly, Constraint Programming, Resource Constrained Project Scheduling, Combinatorial Optimization

1. Introduction

The air transport industry is a growing sector that has rebounded since the COVID-19 pandemic. While the global number of planes in activity is increasing, so too is the number of aircrafts that reach their end of life. Dealing with these aircrafts in an economically beneficial way while limiting their environmental impact is an important challenge for the air transport

Email addresses: charles.thomas@uclouvain.be (Charles Thomas), pierre.schaus@uclouvain.be (Pierre Schaus)

[1]Work financed by the Planum project

industry. In this context, the Planum research project aims at establishing an industry around the dismantling and treatment of end of life aircrafts in Wallonia (Belgium). As part of this project, efforts have been made to improve the efficiency of the dismantling process. One of these research areas concerns the scheduling of the disassembly tasks.

The dismantling of an aircraft consists of the following steps: First, parts and elements that can be reused or individually recycled are removed from the aircraft. Additionally, several pollutants also need to be removed in this phase. Once this is done, the carcass of the aircraft remains. It is then cut and shredded into small material pieces that are then sorted and treated to be either recycled or disposed of.

This paper studies the scheduling of the disassembly tasks that occur in the first half of this process, up to the sectional cutting and shredding of the aircraft. Many pieces and components are removed during this disassembly phase which can lead to around 1500 tasks to schedule for a medium sized plane such as the Boeing B737. Additionally, technicians must be assigned to each tasks and several other considerations must be taken into account such as the balance of the aircraft or specific certifications needed in order for some parts to be allowed to be reused on other aircrafts.

This article is an extension of the work presented in Thomas and Schaus (2024). The aircraft disassembly scheduling problem is presented with additional details. In addition to the CP approach presented in Thomas and Schaus (2024), a MIP approach is proposed. Both approaches are then compared on several instances generated based on real data from an industrial partner. Several variations of the problem are considered where some constraints are deactivated selectively to examine their impact.

1.1. outline

Section 2 presents the problem and gives a small example. Section 3 discusses related work on similar scheduling problems arising in a dismantling context. The CP model is presented and explained in Section 4. The MIP model is shown in Section 5. Section 6 presents the experiments done and their results. Finally, Section 7 offers some closing remarks as well as possible future research insights.

2. Problem

The Aircraft Dismantling Problem (ADP) consists in scheduling the set of tasks performed during the disassembly of the aircraft.

Technicians must be assigned to tasks following strict requirements: Each task involves a specific number of technicians. For tasks relative to the retrieval of parts that are to be reused, at least one technician in the team must have a specific certification. Some technicians may have periods of unavailability during the project span.

In addition, space is limited in some parts of the plane which limits the number of technicians able to work concurrently in these places. For example, a cargo hold may accommodate at most two or three technicians at the same time which limits the number of tasks that can be done in parallel there. Thus, the plane is divided into locations that each have an occupancy limit corresponding to the maximum number of technicians allowed to work there at the same time.

There are precedences between some operations as retrieving some parts my require the prior removal of other parts or the setup of dedicated equipment. Additionally, some groups of tasks must be performed at specific points in the dismantling process. For example, when the aircraft is received, a series of tasks consisting of several integrity and performance tests are done prior to any dismantling operation. These constraints are also introduced as precedences.

Finally, the plane must be kept balanced during the whole disassembly process by ensuring that the difference of mass between its extremities does not overstep given thresholds. To do so, two balance axis are considered: The first axis opposes the front of the aircraft with its rear. The second balance axis opposes both wings of the aircraft. At any time of the planning, the difference of mass between the front and the rear of the aircraft as well as the difference between the left and right wings must not exceed given thresholds.

The objective is to minimize the total time taken by the whole extraction process. This is represented by a *makespan* value that corresponds to the time step at which the last operation finishes.

2.1. Formal definition

In formal terms, the problem is defined as such: The set of all tasks to perform is denoted $\mathcal{T}$. Each task $i \in \mathcal{T}$ is defined by the following elements: a duration needed to perform the task d_i, a location l_i where the task takes place, an occupancy τ_i, which corresponds to the number of technicians mobilized by the task, a mass removed m_i, a set of precedences $\mathcal{P}_i$ referencing other tasks that must end before the start of the task and a set of requirements needed to perform the task $\mathcal{Q}_i$. Each requirement $q \in \mathcal{Q}_i$ of this set is a pair (c_{iq}, n_{iq}) where $c_{iq} \in \mathcal{C}$ is the skill (or certification) required and n_{iq} indicates the number of technicians with this skill needed.

Note that some tasks do not have a specific location or span several locations. Thus their l_i attribute may be empty. Such tasks either do not count towards the occupancy constraint or mobilise the whole plane or technical team, in which case, the precedences are set up so that the task is done at a specific step in the dismantling process and no other task may be done at the same time. Furthermore, some tasks may not have a mass value or specific requirements, in which case, these fields are empty.

All available technicians are part of the set $\mathcal{R}$. Each technician $j \in \mathcal{R}$ is associated with a set of skills $\mathcal{C}_j$ and a set of unavailabilities $u \in \mathcal{U}_j$ consisting of time windows $[s_{ju}, e_{ju}]$ when the technician is not available.

A set of locations $\mathcal{L}$ contains all the locations where operations can take place. Each location $l \in \mathcal{L}$ is associated to a capacity k_l that indicates the maximum number of technicians that can work simultaneously in this location and optionally a zone z_l which corresponds to one of the balance zones of the aircraft. There are four balance zones in total: `Aft` and `Fwd` which correspond to the rear and front of the aircraft and `Left` and `Right` which correspond to the wings.

Two global parameters: B^{af} and B^{lr} indicate the maximum difference of mass allowed at any point in the planning between the `Aft` and `Fwd` zones and the `Left` and `Right` zones respectively.

The objective is to minimize the makespan under the following constraints:

1. All the technicians needed for a task must be allocated during its whole duration.
2. A technician cannot be allocated to different operations at the same time;

3. A technician cannot be scheduled during its unavailabilities;
4. Precedences between tasks must be respected;
5. All the requirements needed for a task must be met.
6. The capacity k_l of a location must not be overloaded at any time;
7. The difference of mass between the `Aft` and `Fwd` zones cannot overstep the balance parameter B^{af} at any time during the planning;
8. The difference of mass between the `Left` and `Right` zones cannot overstep the balance parameter B^{lr} at any time during the planning;

Note that for balance constraints, we consider that the weight change induced by a task happens at the start time of the task.

2.2. Example

Let us consider a small example with 8 dismantling tasks: Task A consists in emptying the fuel tanks of the plane. It must be done at the start of the dismantling process, before the other tasks. Tasks B and C consist in removing the Pilot and Copilot seats in the cockpit of the plane. Two technicians are required for each of these tasks. Task D involves removing the flight controls panel in the cockpit. The pilot and copilot seats must be removed before in order for the technician to access this panel. This part can be reused and thus the task requires a technician with a B1 certification. Tasks E and F consist in removing the thrusters on the left and right engine respectively. At least one technician with the B2 certification must be in the team for both of these tasks. Finally, tasks G and H deal with the removal of the left and right engines. Each of them requires a team of 3 technicians, including 1 with a B2 certification. The thrusters must be removed prior to the removal of the engines. The tasks and their characteristics are shown in Table 1.

Task	Description	l_i	d_i	τ_i	$\mathcal{Q}_i$	m_i	$\mathcal{P}_i$
A	Empty Fuel Tanks		2	1			
B	Rem. Pilot Seat	Cockpit	2	2			A
C	Rem. Copilot Seat	Cockpit	2	2			A
D	Rem. Flight Controls Panel	Cockpit	3	1	(B1,1)		B,C
E	Rem. L. Engine Thruster	L. Wing	3	2	(B2,1)	500	A
F	Rem. R. Engine Thruster	R. Wing	3	2	(B2,1)	500	A
G	Rem. L. Engine	L. Wing	4	3	(B2,1)	1200	E
H	Rem. R. Engine	R. Wing	4	3	(B2,1)	1200	F

Table 1: Tasks of Example 2.2

We have a team of 4 technicians. Technician 2 is unavailable in the time interval $[12, H]$. Technician 3 has a B1 certification and is unavailable in the time interval $[0, 3]$. Technician 4 has a B2 certification.

Only the cockpit location has a capacity constraint, allowing at most two technicians to be present simultaneously. The balance difference along the left-right axis must not exceed 1500. Figure 1 shows the location of the tasks. Note that Task A (Empty Fuel Tanks) has no location. Figure 2 shows the precedences between tasks.

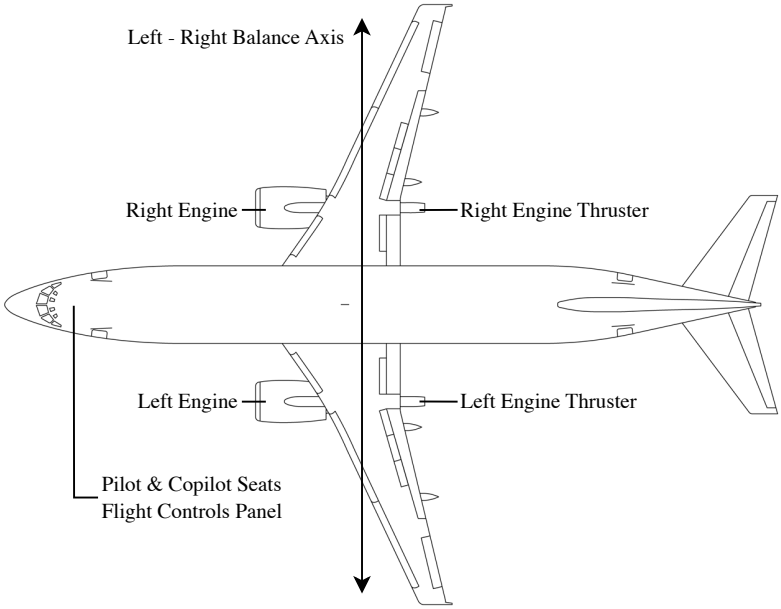

Figure 1: Location of the tasks of Example 2.2

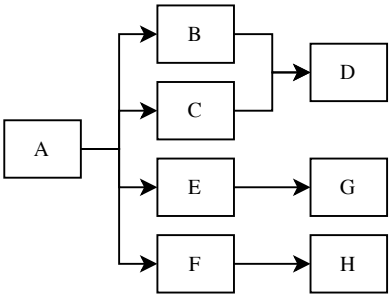

Figure 2: Precedences between the tasks of Example 2.2

A possible solution is shown in Table 2. Its makespan is 16. This is an optimal solution. Figure 3 illustrates the solution. The top part shows

the assignations of the technicians. The middle part shows the occupancy of the cockpit over time. Tasks B, C and D cannot be done simultaneously due to the occupancy constraint. The bottom part shows the evolution of the balance over the left right axis during the dismantling. Note that we must alternate between tasks on the left and right of the plane to satisfy this balance constraint.

Task	s_i	e_i	technicians
A	0	2	1
B	3	5	1,3
C	5	7	1,3
D	7	10	3
E	2	5	2,4
F	5	8	2,4
G	12	16	1,3,4
H	8	10	1,2,4

Table 2: Possible solution for Example 2.2

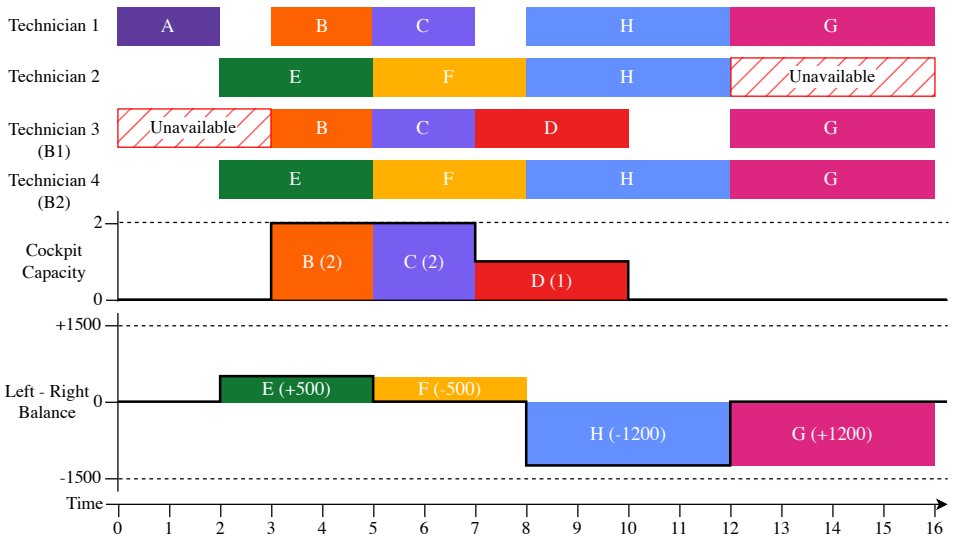

Figure 3: Illustration of a solution for Example 2.2

3. Related work

The Aircraft Dismantling Scheduling problem is a variation of the Resource Constrained Project Scheduling Problem (RCPSP) Özdamar and Ulusoy (1995); Brucker et al. (1999) which consists in scheduling a series

of tasks consuming several resources under precedence constraints. The objective is to find a feasible schedule that minimizes the makespan of the tasks. This problem is NP-complete Garey and Johnson (1975). Several variants of the problem exist Hartmann and Briskorn (2022). The closest one to our current problem is probably the Multi-Skill Project Scheduling Problem (MSPSP) introduced in Bellenguez and Néron (2004). It consists in scheduling tasks and assigning workers with different skill levels to them. It is essentially a relaxed version of the Aircraft Disassembly Scheduling problem without the capacity and balance constraints. In Young et al. (2017), the authors use a CP model to solve several instances of the MSPSP with up to 60 tasks, 19 workers and 15 different skills. In Polo-Mejía et al. (2023), a combination of heuristic and metaheuristic approaches is used to solve a variant of the MSPSP with 50 tasks.

Other publications are related to the problem studied in this paper: In Shan et al. (2017), the authors propose a genetic algorithm to solve an aircraft assembly RCPSP. The authors of da Matta Oliveira Borsato Pinhão et al. (2022) propose an integer programming approach to schedule aircraft engine assembly lines which also involves workers with several skills on up to 100 tasks. In Niu et al. (2023) the authors propose an approach to schedule technicians on short-term aviation maintenance processes (up to 48h). In Srinivasan and Gadh (1999); Zhong et al. (2011); Camelot et al. (2013); Dayi et al. (2016) different approaches are studied to solve problems linked to aircraft disassembly by finding optimal sequences to access specific components based on spatial and geometrical data. Several CP approaches have also been proposed for problems linked to disassembly scheduling: In Lee et al. (2002), a disassembly problem with capacity constraints is studied. The stochastic aspects of disassembly processes are studied in Bentaha et al. (2013) and Tian et al. (2013). In Zwingmann et al. (2008); Edis (2021); Kizilay (2022) several MILP and CP models are proposed to solve disassembly problems but are only able to solve instances up to 150 tasks, while our problem requires solving a variant of the RCPSP with up to 1500 tasks.

4. CP Model

4.1. Variables

The CP model uses conditional interval variables Laborie and Rogerie (2008); Laborie et al. (2012). Each of these variables represents an interval

of time which can be either present or absent. They are used to represent tasks in the model.

Each task $i \in \mathcal{T}$ is modeled with an interval variable $a_i \in \mathcal{A}$ characterized by a start s_i, an end e_i and a duration d_i which is fixed to the duration of the corresponding task in the data. The start and end are initialized to $s_i = [0, H - d_i]$ and $e_i = [d_i, H]$ respectively. These interval variables are always present.

Note that since tasks require technicians with specific skills (certifications), and technicians can have multiple skills as well as their own unavailability, an approach that exploits symmetries by grouping technicians into a single capacitated resource is not feasible. Therefore, the model also needs to assign technicians to each task. This is also represented by interval variables. For each task $i \in \mathcal{T}$, for all technicians $j \in \mathcal{R}$, an optional interval variable ω_{ij} represents the possible assignation of the technician to the task. The initial domain of these interval variables corresponds to the whole planning horizon ($[0, H]$). Task interval variables are always set to present, while the assignment interval variables are optional. The unavailabilities of the technicians are also modeled as interval variables v_{ju} which are set to the time windows corresponding to the unavailabilities.

4.2. Constraints

The model is written as:

$$\text{minimize } \max_{a_i \in \mathcal{A}}(e_i) \tag{1}$$

subject to

$$alternative(a_i, \{\omega_{ij} \mid j \in \mathcal{R}\}, \tau_i) \qquad (i \in \mathcal{T}) \tag{2}$$

$$noOverlap(\{\omega_{ij} \mid i \in \mathcal{R}\} \cup \{v_{ju} \forall u \in \mathcal{U}_j\}) \qquad (j \in \mathcal{R}) \tag{3}$$

$$e_p \leq s_i \qquad (i \in \mathcal{T}, p \in \mathcal{P}_i) \tag{4}$$

$$alternative(a_i, \{\omega_{ij} \mid j \in \mathcal{R} \wedge c_{iq} \in \mathcal{C}_j\}, x_{iq}) \qquad (i \in \mathcal{T}, q \in \mathcal{Q}_i) \tag{5}$$

$$o_l = \sum_{a_i \in \mathcal{A} | l_i = l} pulse(a_i, \tau_i) \qquad (l \in \mathcal{L}) \tag{6}$$

$$0 \leq o_l \leq k_l \qquad (l \in \mathcal{L}) \tag{7}$$

$$b^{af} = step(0, B^{af}) + \sum_{a_i \in \mathcal{A} | z_{l_i} = \texttt{Aft}} stepAtStart(a_i, m_i) + \sum_{a_i \in \mathcal{A} | z_{l_i} = \texttt{Fwd}} stepAtStart(a_i, -m_i) \tag{8}$$

$$b^{lr} = step(0, B^{lr}) + \sum_{a_i \in \mathcal{A} | z_{l_i} = \texttt{Left}} stepAtStart(a_i, m_i) + \sum_{a_i \in \mathcal{A} | z_{l_i} = \texttt{Right}} stepAtStart(a_i, -m_i) \tag{9}$$

$$0 \leq b^{af} \leq B^{af} \cdot 2 \tag{10}$$

$$0 \leq b^{lr} \leq B^{lr} \cdot 2 \tag{11}$$

Constraint (2) ensures that exactly τ_i technicians are selected for each task $i \in \mathcal{T}$. Constraint (3) ensures that each technician is assigned to a single task at the same time by enforcing that no overlap occurs between optional intervals of the technician. This constraint also ensures that technicians are not assigned during their non-availability periods by also considering unavailability intervals u_{ju} in the set of intervals which must not overlap. Precedence constraints ensure that preceding activities are finished when an activity starts (4).

Skill requirements. The skill requirements are enforced by constraint (5). It ensures that the number of technicians that possess a skill needed for a task and are assigned to the task is higher or equal to the number required with this skill. The variable x_{iq} is a cardinality variable whose initial domain is $[n_{iq}, |\mathcal{R}_{iq}|]$ where $\mathcal{R}_{iq} = \{j \in \mathcal{R} \mid c_{iq} \in \mathcal{C}_j\}$ is the set of all technicians that posses the skill required by the requirement q of the task i. Using a cardinality variable is necessary in order to allow more than the number of technicians with the skill required to be assigned to the task.

Locations capacity. Occupancy and balance constraints are modelled using cumulative functions. For occupancy constraints, each location in the aircraft $l \in \mathcal{L}$ is modelled by a cumulative function o_l (6) which tracks the number of technicians working in this location. This cumulative function is linked to the task activities taking place at this location and must not overstep the capacity of the location k_l (7).

Balance. For balance constraints, two cumulative functions are used: The function b^{af} (8) models the difference of mass between the front and rear of the aircraft (the `Aft` and `Fwd` zones). The function b^{lr} (9) does the same for the left and right wings (the `Left` and `Right` zones). When some weight is removed in one of these balance zones, it is either added to or subtracted from the relevant cumulative function.

For example, if an operation removes 30 units of weight on the left wing of the aircraft, this amount will be added to the cumulative function b^{lr} while an operation that removes weight on the right wing will have this weight subtracted from the b^{lr} function. We use *stepAtStart* functions to add these weights to the balance cumulative functions at the start time of the tasks. In order to avoid having to deal with negative cumulative functions, these are shifted by the amount of tolerated mass difference (B^{af} or B^{lr}). These cumulative function thus start at the tolerated mass difference (B^{af} or B^{lr}) and must at all time be included between zero and twice the tolerated mass difference value (10), (11).

5. MIP Model

The MIP model proposed to solve the Aircraft Dismantling Problem is based on the On/Off Event-based MIP Formulation (OOE) for the RCPSP from Brucker and Knust (2012). This formulation is independent of the time span of the project and instead scales with the number of tasks. The intuition is to discretize the time into events which correspond to the starts of tasks. Constraints that must be enforced at any time can then be decomposed and enforced at each event.

In this model, technicians non-availabilities are modelled as additional tasks with fixed time windows to which the unavailable technicians are assigned. This set of additional tasks is denoted $\mathcal{U}$. The set of all tasks is thus $\mathcal{T}' = \mathcal{T} \cup \mathcal{U}$. We consider $|\mathcal{T}'|$ events $e \in \mathcal{E}$ which correspond to the starting times of tasks. Additionally, a global planning horizon H is computed based on the sum of all tasks duration.

5.1. Variables

The binary variables z_{ie} for $i \in \mathcal{T}', e \in \mathcal{E}$ indicate if task i is processed during event e:

$$z_{ie} = \begin{cases} 1 & \text{if task } i \text{ is processed during event } e \\ 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}'; e \in \mathcal{E}) \qquad (12)$$

Additionally, auxiliary variable $z_{i,-1} = 0$ for $i \in \mathcal{T}'$ are used to deal with the case when $e = -1$ in some constraints.

The continuous variables t_e for $e \in \mathcal{E}$ represent the time at which events occur. The continuous variable t_{max} corresponds to the makespan objective.

The continuous variables h_{ie} for $i \in \mathcal{T}, e \in \mathcal{E}$ are intermediate variables that are used for the computation of the makespan:

$$h_{ie} = \begin{cases} t_e + d_i & \text{if task } i \text{ starts at event } e \\ \leq 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}; e \in \mathcal{E}) \qquad (13)$$

The binary variables x_{ij} for $i \in \mathcal{T}', j \in \mathcal{R}$ correspond to the assignment of technician j to task i:

$$x_{ij} = \begin{cases} 1 & \text{if technician } j \text{ is assigned to task } i \\ 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}) \qquad (14)$$

The binary variables y_{ije} for $i \in \mathcal{T}', j \in \mathcal{R}, e \in \mathcal{E}$ are event assignment variables:

$$y_{ije} = \begin{cases} 1 & \text{if technician } j \text{ is assigned to task } i \\ & \text{during event } e \\ 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}, e \in \mathcal{E}) \qquad (15)$$

These variables are used in some constraints and make the link between the variables z_{ie} and x_{ij}.

The binary variables a_{ie} for $i \in \mathcal{T}, e \in \mathcal{E}$ indicate if task i starts at event e:

$$a_{ie} = \begin{cases} 1 & \text{if task } i \text{ starts at event } e \\ 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}; e \in \mathcal{E}) \qquad (16)$$

Note that, as these variables are used for balance constraints which only apply for original tasks, they are created for the set of original tasks $\mathcal{T}$.

Finally, the continuous variables b_e^{af} and b_e^{lr} for $e \in \mathcal{E}$ track the balance of the aft-forward and left-right axis respectively and are used for balance constraints. Two auxiliary variable $b_{-1}^{af} = 0$ and $b_{-1}^{lr} = 0$ are used for the case $e = -1$.

5.2. Constraints

The MIP model is written as:

$$\text{minimize } t_{max} \tag{17}$$

subject to

$$t_0 = 0 \tag{18}$$

$$t_e \geq 0 \qquad (e \in \mathcal{E}) \tag{19}$$

$$t_e - t_{e-1} \geq 0 \qquad (e \in \mathcal{E} \setminus \{0\}) \tag{20}$$

$$z_{i,-1} = 0 \qquad (i \in \mathcal{T}') \tag{21}$$

$$z_{ie} \in \{0, 1\} \qquad (i \in \mathcal{T}'; e \in \mathcal{E}) \tag{22}$$

$$\sum_{e \in \mathcal{E}} z_{ie} \geq 1 \qquad (i \in \mathcal{T}') \tag{23}$$

$$\sum_{e'=0}^{e-1} z_{ie'} - e(1 - (z_{ie} - z_{i,e-1})) \leq 0 \qquad (i \in \mathcal{T}'; e \in \mathcal{E} \setminus \{0\}) \tag{24}$$

$$\sum_{e'=e}^{N-1} z_{ie'} - (N - e)(1 + (z_{ie} - z_{i,e-1})) \leq 0 \qquad (i \in \mathcal{T}'; e \in \mathcal{E} \setminus \{0\}) \tag{25}$$

$$t_f - t_e - d_i \begin{pmatrix} z_{ie} - z_{i,e-1} - \\ (z_{if} - z_{i,f-1}) \end{pmatrix} \geq -d_i \qquad (i \in \mathcal{T}'; e, f \in \mathcal{E} \mid e < f) \tag{26}$$

$$h_{ie} \leq H(z_{ie} - z_{i,e-1}) \qquad (i \in \mathcal{T}, e \in \mathcal{E}) \tag{27}$$

$$h_{ie} \leq t_e + d_i \qquad (i \in \mathcal{T}, e \in \mathcal{E}) \tag{28}$$

$$h_{ie} \geq t_e + d_i - H(1 - (z_{ie} - z_{i,e-1})) \qquad (i \in \mathcal{T}, e \in \mathcal{E}) \tag{29}$$

$$t_{max} \geq 0 \tag{30}$$

$$t_{max} \geq h_{ie} \qquad (i \in \mathcal{T}; e \in \mathcal{E}) \tag{31}$$

$$y_{ije} \leq x_{ij} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \tag{32}$$

$$y_{ije} \leq z_{ie} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \tag{33}$$

$$y_{ije} \geq x_{ij} + z_{ie} - 1 \qquad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \tag{34}$$

$$y_{ije} \geq 0 \qquad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \tag{35}$$

$$\sum_{j \in \mathcal{R}} y_{ije} = \tau_i z_{ie} \qquad (i \in \mathcal{T}'; e \in \mathcal{E}) \tag{36}$$

$$\sum_{i \in \mathcal{T}'} y_{ije} \leq 1 \qquad (j \in \mathcal{R}; e \in \mathcal{E}) \tag{37}$$

$$x_{ij} \leq \sum_{e \in \mathcal{E}} y_{ije} \leq N x_{ij} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}) \tag{38}$$

$$1 - x_{ij} \leq \sum_{e \in \mathcal{E}} z_{ie} - \sum_{e \in \mathcal{E}} y_{ije} \leq N(1 - x_{ij}) \qquad (i \in \mathcal{T}'; j \in \mathcal{R}) \quad (39)$$

$$\sum_{j \in \mathcal{R}} x_{ij} = \tau_i \qquad (i \in \mathcal{T}') \quad (40)$$

$$x_{uj} = 1 \qquad (j \in \mathcal{R}, u \in \mathcal{U}_j) \quad (41)$$

$$t_e \geq s_{ju} z_{ue} \qquad (j \in \mathcal{R}, u \in \mathcal{U}_j, e \in \mathcal{E}) \quad (42)$$

$$t_e \leq \; s_{ju}(z_{ue} - z_{u,e-1}) + H(1 - (z_{ue} - z_{u,e-1})) \qquad (j \in \mathcal{R}, u \in \mathcal{U}_j, e \in \mathcal{E}) \quad (43)$$

$$z_{pe} + \sum_{e'=0}^{e} z_{ie'} - e(1 - z_{pe}) \leq 1 \qquad (i \in \mathcal{T}; p \in \mathcal{P}_i; e \in \mathcal{E}) \quad (44)$$

$$\sum_{j \in \mathcal{R} | c_{iq} \in \mathcal{C}_j} x_{ij} \geq n_{iq} \qquad (i \in \mathcal{T}; q \in \mathcal{Q}_i) \quad (45)$$

$$\sum_{i \in \mathcal{T} | l_i = l} \tau_i z_{ie} \leq k_l \qquad (l \in \mathcal{L}; e \in \mathcal{E}) \quad (46)$$

$$a_{ie} \geq 0 \qquad (i \in \mathcal{T}; e \in \mathcal{E}) \quad (47)$$

$$a_{ie} \geq z_{ie} - z_{i,e-1} \qquad (i \in \mathcal{T}; e \in \mathcal{E}) \quad (48)$$

$$b_e^{af} = b_{e-1}^{af} + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Aft}} a_{ie} m_i + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Fwd}} a_{ie}(-m_i) \qquad (e \in \mathcal{E}) \quad (49)$$

$$b_e^{lr} = b_{e-1}^{lr} + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Left}} a_{ie} m_i + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Right}} a_{ie}(-m_i) \qquad (e \in \mathcal{E}) \quad (50)$$

$$-B^{af} \leq b_e^{af} \leq B^{af} \qquad (e \in \mathcal{E}) \quad (51)$$

$$-B^{lr} \leq b_e^{lr} \leq B^{lr} \qquad (e \in \mathcal{E}) \quad (52)$$

Constraints (18), (19) and (20) ensure that the timing of events follow an increasing order: $t_0 \leq t_1 \leq ... \leq t_{N-1}$. Constraints (21) and (22) initialize z_{ie} variables while constraint (23) ensure that each task is processed during at least one event. Constraints (24) and (25) make sure that each task is processed in a contiguous block of events: Constraint (24) makes sure that when a task is processed at an event e but not the previous one ($z_{ie} - z_{i,e-1} = 1$), it is never processed before ($\sum_{e'=0}^{e-1} z_{ie'} \leq 0$). Constraint (25) enforces that when a task is not processed at an event e but processed at the previous one ($z_{ie} - z_{i,e-1} = -1$), it is never processed after ($\sum_{e'=e}^{N-1} z_{ie'} \leq 0$).

Constraint (26) enforces the processing time of tasks based on task processing variables z_{ie} and time variables t_e. It ensures that for any two events $e, f \in \mathcal{E}$ with $e < f$, task i may start at event e and end at event f only if the difference of time between f and e is larger or equal to the duration of task i.

Constraints (27), (28) and (29) set up the intermediate variables h_{ie} for the makespan calculation based on the time variables and the task processing variables. Constraints (30) and (31) link the makespan objective (17) with the intermediate variables.

Constraints (32) to (35) link together event assignment variables y_{ije} with assignment variables x_{ij} and task processing variables z_{ie} following the relation $y_{ije} = x_{ij} \wedge z_{ie}$. Constraint (36) links the task processing variables z_{ie} with the event assignment variables y_{ije} and ensures that the correct number of technicians is used when a task is processed. Constraint (37) prevents a technician to be concurrently assigned to more than one task. Constraint (38) links assignment variables x_{ij} with event assignment variables y_{ije}. Constraint (39) makes sure that the number of active event assignment variables $\sum y_{ije}$ is equal to either the number of active task processing events $\sum z_{ie}$ if the assignment variable x_{ij} is true or 0 if false. Constraint (40) ensures that the correct number of technicians is assigned to each task.

Constraint (41) fixes the unavailability tasks to the corresponding technician. Constraints (42) and (43) fixes the start and end of the the unavailability tasks. Constraint (44) sets up precedences between tasks: if a precedence task p is processed last at event e, then $\sum_{e'=0}^{e} z_{ie'} = 0$ which implies that task i must be processed later.

Requirements. Constraint (45) ensures that requirements are respected: For each requirement $q \in \mathcal{Q}_i$ of each task $i \in \mathcal{T}$, the constraint ensures that the sum of technicians assigned to the task that have the requested skill $(j \in \mathcal{R} \mid c_{iq} \in \mathcal{C}_j)$ is greater or equal to the requested amount n_{iq}.

Locations capacity. Constraint (46) models the locations capacity constraint of the problem by limiting the cumulative occupation of tasks for each location $l \in \mathcal{L}$ during each event.

Balance constraints. Constraints (47) and (48) link together the task start variable a_{ie} with the task processing variables z_{ie}. Constraints (49) and (50)

set up balance variables b_e^{af} and b_e^{lr}: At each event e, the balance is equal to the balance at the previous event (b_{e-1}^{af} or b_{e-1}^{lr}) plus the balance change at this event which corresponds to the value of mass removed during tasks impacting the balance axis that start at event e which is either added or removed depending on the location of the task:

$$\sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Aft/Left}} a_{ie} m_i + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Fwd/Right}} a_{ie}(-m_i) \tag{53}$$

Finally, constraints (51) and (52) enforce that the balance variables b_e^{af} and b_e^{lr} stay within the balance range at each event e.

5.3. Discussion

While this MIP model avoids a decomposition in time steps which scales poorly on larger problems, it remains costly in terms of variables and constraints with $O(|\mathcal{T}'|^2|\mathcal{R}|)$ binary variables, $O(|\mathcal{T}'|)$ continuous variables and $O(|\mathcal{T}'|^2|\mathcal{R}| + (|\mathcal{P}| + |\mathcal{R}|)|\mathcal{T}'|)$ constraints. Furthermore, the decomposition in events introduces symmetries if several tasks are started at the same time. In this case, several events will share the same time and may be interchangeable.

6. Experiments

This section presents the experiments done with both models and their results. Two experiments are considered: The first one compares both approaches on the set of all instances. The second one compares the anytime behavior of the CP approach on several variations of the problem where specific constraints are deactivated to examine their impact on the problem.

6.1. Data

The instances used in the experiments are based on data provided by an industrial partner from the Planum research project. It was collected during the full dismantling of a Boeing 737NG-600 aircraft. It consists in a list of 1454 tasks that are performed as part of the aircraft disassembly.

Most of the data comes from this source with the exception of the mass values used for the balance constraints. Indeed, the industrial partner is still in the process of collecting this data and thus, the generated instances

were completed with arbitrary mass values. A mass value between 5kg and 500kg is assigned to 214 tasks situated in the four balance zones. The other tasks have a mass impact of 0. The maximum difference of mass allowed is 300kg on the Aft - Forward axis and 500kg on the Left - Right axis.

Two different skills are considered which correspond to certifications needed in the air transport industry: B1 and B2. Task durations are calculated based on the industrial data. A unit of time corresponds to 15 minutes which is the smallest time precision observed in the real world data. There are 13 different locations with capacities varying between 2 and 10. An additional dummy location with an infinite capacity is used for a few tasks that either apply to the whole plane or for which the location is not relevant.

The instances used in the experiments were created based on this dataset. The instance B737NG600-1454 corresponds to the whole set of tasks. 15 smaller instances of various sizes were generated based on this instance by removing some of the tasks. The tasks removed are selected randomly. The instances are named with the following convention: `B737NG600-<number of tasks>.json`.

Each instance uses the same set of technicians with 7 technicians available. Among them, one has the B1 certification, one has the B2 certification, one has both B1 and B2 certifications and the four others have no certification. Some unavailability periods are randomly assigned to the technicians.

Figure 4 shows a visualization of some of the tasks features of the instance B737NG600-1454. Tasks are sorted by duration, number of technicians required and skill requirements. The height of the bars in the top part of the graph show the duration of each task. The colors in the top part show the number of technicians required. The colors in the bottom part show the skill requirements.

We can see that the vast majority of tasks have a duration under 10 and require no more than one or two technicians. There is one single longest task that has a duration of 64 and requires 4 technicians. There is $\tilde{2}2\%$ of tasks with a B1 requirement, $\tilde{2}0\%$ with a B2 requirement and 2 tasks with both B1 and B2. The remaining tasks have no requirement.

An anonymized version of the instances as well as the models and results is made available at the following repository: `https://github.com/cftmthomas/AircraftDisassembly`.

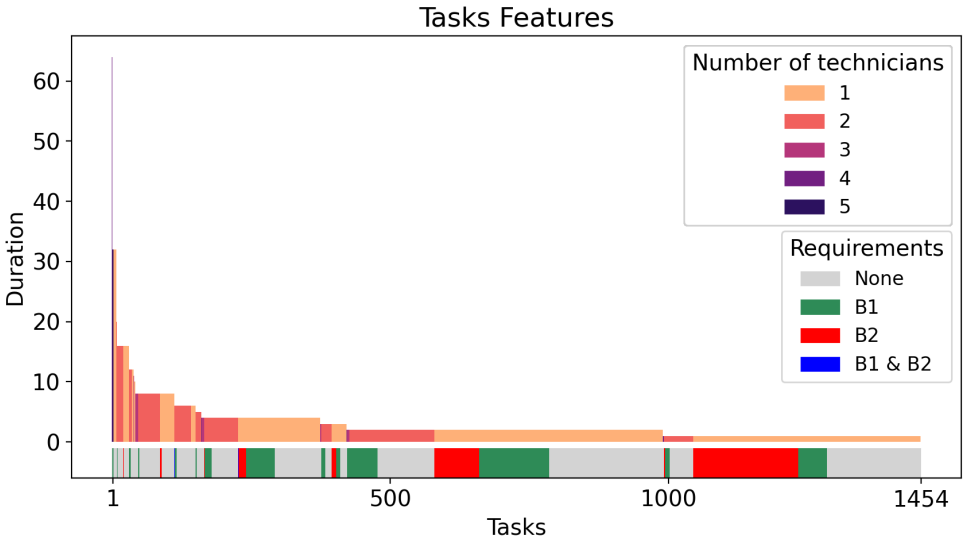

Figure 4: Durations and number of technicians for the tasks of instance B737NG600-1454

6.2. Experimental setup

We compare the CP and MIP approaches on the set of all instances. Additionally, we compare the performances of the CP approach on the base problem with all its constraints as well as several relaxations where some of the constraints have been deactivated in order to examine their impact on the problem difficulty:

Requirements relaxation. The first relaxation consists in discarding the certifications requirements for all tasks. For the CP model, this consists in removing constraint (5).

Capacity relaxation. This relaxation consists in deactivating the locations capacities constraints. Constraints (6) and (7) are deactivated in the CP model.

Balance relaxation. The third relaxation consists in deactivating the balance constraints for both balance axis. For the CP model, this consist in removing constraints (8), (9), (10) and (11).

Requirements, capacity and balance relaxation. The last relaxation considered combines the three previous ones by discarding the requirements, locations capacity and balance constraints. It is the closest to a classical

RCPSP with the only difference being that technicians can be unavailable during some parts of the scheduling period.

The CP Model is implemented in CP Optimizer with the default search used. This search consists in an Adpative LNS (ALNS) approach Laborie et al. (2018) enhanced with a failure directed search to prove optimality. The MIP model is implemented with CPLEX mixed integer optimizer with its default search used for the experiments. Both solvers are part of IBM's CPLEX optimization suite whose version 22.1.1 was used in the experiments.

Experiments were conducted on a Linux server with 2 processors Intel(R) Xeon(R) E5-2687W (40 threads total) and 128 GB of RAM. Both approaches were limited to a single thread for each run with a time limit of 3600 seconds and a maximum memory limit of 8GB.

6.2.1. Performance measurement

We compare the results using the primal integral method proposed in Berthold (2013). Intuitively, this performance metric consists in measuring the area under the anytime objective curve during the whole search. In order to compare performances over different instances, the primal integral is computed based on the primal gap that normalizes an objective value based on the optimum or best known objective. Given a solution x, an optimal (or best known) solution x^* and an objective function $o()$, the primal gap $\gamma \in [0, 1]$ is defined as:

$$\gamma(x) = \begin{cases} 0 & \text{if } |o(x^*)| = |o(x)| = 0 \\ 1 & \text{if } o(x^*) \cdot o(x) < 0 \\ \frac{|o(x^*) - o(x)|}{max(|o(x^*)|, |o(x)|)} & \text{otherwise} \end{cases} \tag{54}$$

The primal gap thus corresponds to the ratio between the distance of the current objective to the best objective $|o(x^*) - o(x)|$ and the largest absolute value between the two. This ratio tends to 1 if the current objective tends to ∞ and reaches 0 if $|o(x^*)| = |o(x)|$.

For an experimental run where several improving solutions have been found at given points in time, the primal gap can be used to compute a

primal gap step function $p(t)$ which is defined as:

$$p(t) = \begin{cases} 1 & \text{if no incumbent has been found until time } t \\ \gamma(x(t)) & \text{otherwise} \end{cases} \tag{55}$$

where $x(t)$ is the incumbent solution at time t. The function $p(t)$ starts at 1 until a first solution has been found and decreases to reach 0 once the optimum or best known solution has been found.

The primal integral $P(T)$ for a time $T \in [0, t_{max}]$ is the integral of the primal gap function from 0 to T:

$$P(T) = \int_{t=0}^{T} p(t)dt = \sum_{i=1}^{I} p(t_{i-1}) \cdot (t_i - t_{i-1}) \tag{56}$$

where $t_0 = 0$, $t_i \in [0, T]$ for $i \in 1, \ldots, I$ denotes the time points at which a solution has been found and $t_I = T$.

6.3. Computational results

Models comparison. This first experiment consist in comparing both approaches on all instances for the full problem with all its constraints.

Table 3 presents the results for the problem with all constraints. The columns *name* and obj^* indicate the name and best known solution for each instance. For each approach, the column $P(t_{max})$ shows the primal integral (computed with $t_{max} = 3600$); the column *obj* shows the best solution obtained and the column t^* indicates the time to prove optimality if the approach was able to. The character "`-`" indicates a timeout and the character "`x`" indicates that the approach ran out of memory. As we can see, the CP approach vastly outperforms the MIP approach which is only able to solve small sized instances, up to 30 tasks. The CP approach is able to prove optimality for instances up to 30 tasks but stagnates until timeout for the larger instances.

The MIP approach is only able to find a solution on the four smallest instances and to solve only two of them to optimality. Furthermore, it runs out of memory (8G per thread) on instances of 600 or more tasks. This can be explained by the fact that the MIP model uses a number of constraints and variables that grow quadratically with the number of tasks. Additional experiments have shown that this behavior stays the same on

Table 3: Results on the problem with all constraints

Instance		CP			MIP		
Name	$obj*$	$P(t_{max})$	obj	$t*$	$P(t_{max})$	obj	$t*$
B737NG600-10	64	0.022	64	0.044	58.633	64	179.661
B737NG600-15	64	0.007	64	0.009	238.449	64	463.419
B737NG600-20	65	0.023	65	0.048	1480.616	72	-
B737NG600-30	68	0.043	68	1.914	2453.189	128	-
B737NG600-40	91	0.057	91	-	-	-	-
B737NG600-50	93	0.108	93	-	-	-	-
B737NG600-75	114	0.114	114	-	-	-	-
B737NG600-100	117	0.152	117	-	-	-	-
B737NG600-150	159	0.205	159	-	-	-	-
B737NG600-200	184	0.410	184	-	-	-	-
B737NG600-300	250	1.602	250	-	-	-	-
B737NG600-400	287	1.102	287	-	-	-	-
B737NG600-600	420	7.581	420	-	x	x	x
B737NG600-800	505	15.697	505	-	x	x	x
B737NG600-1200	834	17.789	834	-	x	x	x
B737NG600-1454	973	31.022	973	-	x	x	x

all variations of the problem and in some cases is even worse. The full results of these experiments are available in the repository `https://github.com/cftmthomas/AircraftDisassembly`. Due to its poor performances, the MIP model is not considered in the next experiment.

Anytime performances of the CP approach. This experiment aims at comparing the impact of different constraints on the problem difficulty. To do so, we compare the anytime behavior of the CP approach on several variations of the problem where some constraints are deactivated.

Figure 5 shows the anytime objective value of the CP approach on the largest instance B373NG600-1454 for the different variations considered. We show only the start of the search, from 0 to 200s, as this is the part with the most variation in the objective values. After that time, we observe mostly stagnation with occasional very small improvements of the objective. Eventually, a best known solution is reached which has the same objective value of 973 for all variations of the problem. It is indicated by the horizontal dashed line.

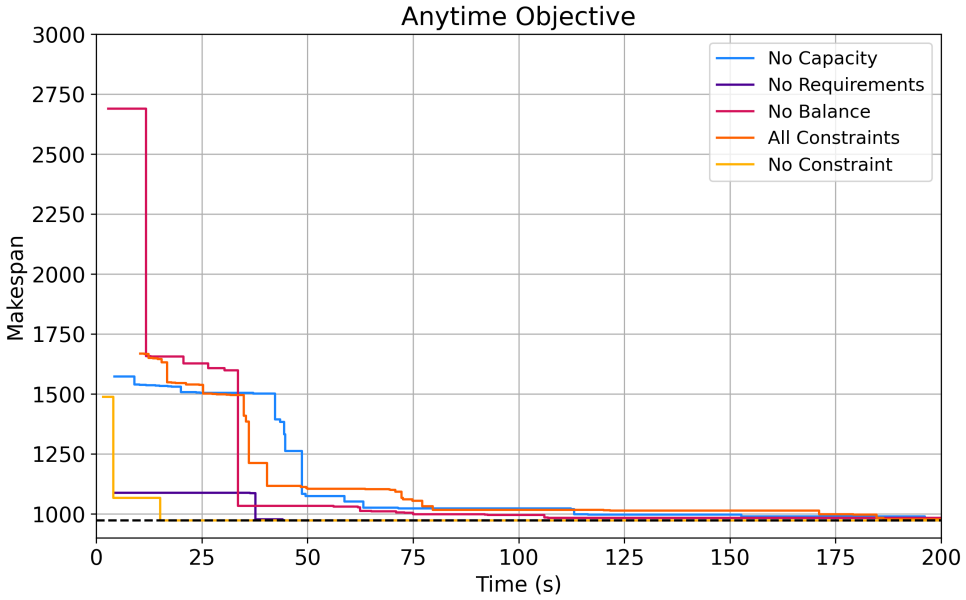

Figure 5: Anytime performances of the CP approach on the instance B373NG600-1454

This stagnation combined with the fact that all the variations have the same objective for their best known solution suggests that the three constraints considered (Requirements, Balance and Capacity) do not impact much the optimal makespan. This can be explained by the characteristics of this instance, which consists of many small tasks with short durations, each requiring only one or two technicians. his provides a high degree of flexibility in accommodating these constraints, which may explain their lack of impact on the makespan.

We observe that deactivating constraints improves the anytime behavior, with the relaxation of the *Requirements* constraint having the most significant impact. This can be attributed to the relatively large number of tasks with specific requirements, coupled with the fact that only three technicians possess the necessary certifications.

The results are less explicit when considering the Capacity or Balance constraint individually. Removing these constraints seems to allow finding a first solution earlier compared to the variation with all constraints. However the anytime behavior on these relaxations does not clearly dominates the version with all constraints. The lower impact of these constraints can again be explained by the large number of small tasks that offer a lot of flexibility in the scheduling.

7. Conclusion

We presented the aircraft disassembly scheduling problem. This problem which consists in secheduling a set of dismantling tasks while assigning human resources to them is a variation of the RCPSP. The problem also deals with skill requirements and additional constraints related to the balance and capacity of some parts the aircraft.

We proposed two approaches to tackle the problem. The first one is a constraint programming model using advanced modeling features including conditional task intervals, sequence variables and cumulative functions. The second one is a MIP model.

Experiments were run on 16 instances derived from real data provided by an industrial partner. The experiments show the the CP model vastly outperforms the MIP model and is able to solve instances of up to 1450 tasks. Several variations of the problem were considered where some constraints are removed to examine their impact on the problem difficulty. The results indicate that the requirements constraint (some technicians assigned to some tasks require specific certifications) is the one that affects the most the anytime behavior on the problem.

7.0.1. Future Work

As future work, we would like to investigate metaheuristic methods to solve the problem, as these methods have been successfully applied to variants of the RCPSP Laurent et al. (2017); Mischek and Musliu (2021). Including a possible uncertainty in the duration of the task could be useful, and the solutions can then be made more robust against this uncertainty using the techniques introduced in Davenport et al. (2001).

Acknowledgments

This work was funded by the Walloon Region (Belgium) as part of the Planum project. We thank Sabena Engineering for allowing the diffusion of an anonymized version of the dataset provided.

References

Bellenguez, O. and Néron, E. (2004). Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In *Inter-*

national conference on the practice and theory of automated timetabling, pages 229–243. Springer.

Bentaha, M. L., Battaïa, O., and Dolgui, A. (2013). Chance constrained programming model for stochastic profit–oriented disassembly line balancing in the presence of hazardous parts. In *Advances in Production Management Systems. Sustainable Production and Service Supply Chains: IFIP WG 5.7 International Conference, APMS 2013, State College, PA, USA, September 9-12, 2013, Proceedings, Part I*, pages 103–110. Springer.

Berthold, T. (2013). Measuring the impact of primal heuristics. *Operations Research Letters*, 41(6):611–614.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., and Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41.

Brucker, P. and Knust, S. (2012). *Resource-Constrained Project Scheduling*, pages 117–238. Springer Berlin Heidelberg, Berlin, Heidelberg.

Camelot, A., Baptiste, P., and Mascle, C. (2013). Decision support tool for the disassembly of reusable parts on an end-of-life aircraft. In *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IESM)*, pages 1–8. IEEE.

da Matta Oliveira Borsato Pinhão, J., Ignacio, A. A. V., and Coelho, O. (2022). An integer programming mathematical model with line balancing and scheduling for standard work optimization: A realistic application to aircraft engines assembly lines. *Computers & Industrial Engineering*, 173:108652.

Davenport, A. J., Gefflot, C., Beck, J. C., et al. (2001). Slack-based techniques for robust schedules. In *Proceedings of the Sixth European Conference on Planning (ECP-2001)*, pages 7–18.

Dayi, O., Afsharzadeh, A., and Mascle, C. (2016). A lean based process planning for aircraft disassembly. *IFAC-PapersOnLine*, 49(2):54–59.

Edis, E. B. (2021). Constraint programming approaches to disassembly line balancing problem with sequencing decisions. *Computers & Operations Research*, 126:105111.

Garey, M. R. and Johnson, D. S. (1975). Complexity results for multiprocessor scheduling under resource constraints. *SIAM journal on Computing*, 4(4):397–411.

Hartmann, S. and Briskorn, D. (2022). An updated survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, 297(1):1–14.

Kizilay, D. (2022). A novel constraint programming and simulated annealing for disassembly line balancing problem with and/or precedence and sequence dependent setup times. *Computers & Operations Research*, 146:105915.

Laborie, P. and Rogerie, J. (2008). Reasoning with conditional time-intervals. In *FLAIRS conference*, pages 555–560.

Laborie, P., Rogerie, J., Shaw, P., and Vilím, P. (2018). Ibm ilog cp optimizer for scheduling: 20+ years of scheduling with constraints at ibm/ilog. *Constraints*, 23:210–250.

Laborie, P., Rogerie, J., Shaw, P., Vilím, P., and Katai, F. (2012). Interval-based language for modeling scheduling problems: An extension to constraint programming. *Algebraic Modeling Systems: Modeling and Solving Real World Optimization Problems*, pages 111–143.

Laurent, A., Deroussi, L., Grangeon, N., and Norre, S. (2017). A new extension of the rcpsp in a multi-site context: Mathematical model and metaheuristics. *Computers & Industrial Engineering*, 112:634–644.

Lee, D.-H., Xirouchakis, P., and Zust, R. (2002). Disassembly scheduling with capacity constraints. *CIRP Annals*, 51(1):387–390.

Mischek, F. and Musliu, N. (2021). A local search framework for industrial test laboratory scheduling. *Annals of Operations Research*, 302(2):533–562.

Niu, B., Xue, B., Zhong, H., Qiu, H., and Zhou, T. (2023). Short-term aviation maintenance technician scheduling based on dynamic task disassembly mechanism. *Information Sciences*, 629:816–835.

Polo-Mejía, O., Artigues, C., Lopez, P., Mönch, L., and Basini, V. (2023). Heuristic and metaheuristic methods for the multi-skill project scheduling problem with partial preemption. *International Transactions in Operational Research*, 30(2):858–891.

Shan, S., Hu, Z., Liu, Z., Shi, J., Wang, L., and Bi, Z. (2017). An adaptive genetic algorithm for demand-driven and resource-constrained project scheduling in aircraft assembly. *Information Technology and Management*, 18:41–53.

Srinivasan, H. and Gadh, R. (1999). Selective disassembly of components with geometric constraints. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 19746, pages 571–579. American Society of Mechanical Engineers.

Thomas, C. and Schaus, P. (2024). A constraint programming approach for aircraft disassembly scheduling. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 211–220. Springer.

Tian, G., Zhou, M., and Chu, J. (2013). A chance constrained programming approach to determine the optimal disassembly sequence. *IEEE Transactions on Automation Science and Engineering*, 10(4):1004–1013.

Young, K. D., Feydy, T., and Schutt, A. (2017). Constraint programming applied to the multi-skill project scheduling problem. In *Principles and Practice of Constraint Programming: 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28–September 1, 2017, Proceedings 23*, pages 308–317. Springer.

Zhong, L., Youchao, S., Ekene Gabriel, O., and Haiqiao, W. (2011). Disassembly sequence planning for maintenance based on metaheuristic method. *Aircraft Engineering and Aerospace Technology*, 83(3):138–145.

Zwingmann, X., Ait-Kadi, D., Coulibaly, A., and Mutel, B. (2008). Optimal disassembly sequencing strategy using constraint programming approach. *Journal of Quality in Maintenance Engineering*, 14(1):46–58.

Özdamar, L. and Ulusoy, G. (1995). A survey on the resource-constrained project scheduling problem. *IIE Transactions*, 27(5):574–586.