Solving the Aircraft Disassembly Scheduling Problem

Charles Thomas[1*] and Pierre Schaus[1*]

[1*]Institute of Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), UCLouvain, Place Sainte Barbe 2/L5.02.01, Louvain-la-Neuve, 1348, Belgium.

*Corresponding author(s). E-mail(s): charles.thomas@uclouvain.be; pierre.schaus@uclouvain.be;

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

Acknowledgments

Charles Thomas contribution 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.

1 Introduction

The air transport industry is a growing sector that has rebounded since the COVID-19 pandemic. Despite a recent slowing down, the growth of the sector is projected to continue in the future with an annual production of around 2000 new aircrafts per year (IATA (2025)). While the global number of planes in activity is increasing, so too is the number of aircrafts that reach their end of life (EOL) phase. Dealing with these aircrafts in an economically beneficial way while limiting their environmental impact is an important challenge for the air transport industry.

Recent studies (see Asmatulu, Overcash, and Twomey (2013); Habib et al. (2025); Sabaghi, Cai, Mascle, and Baptiste (2016); Scheelhaase, Müller, Ennen, and Grimme (2022)) indicate that, while aircraft recycling provides environmental and economic advantages, the resulting profit margins are strongly influenced by the condition of the recycled aircraft. The value of an EOL aircraft usually

ranges between USD 1 million and USD 3 million. However, this value is conditional to the state of the aircraft and whether it is in order of maintenance. Currently, most of the value come from the reselling of parts extracted during the disassembly process, of which engine components are the most valuable and may account up to 80% of the value recovered.

While valuable in theory, materials are more difficult reuse due to the mixing of several components into alloys and the increased use of polymer materials. Current recycling processes for both metals and polymers require a substantial amount of energy and must deal with the presence of impurities such as undesirable elements from paints and isolation, which make their recycling costly.

Nevertheless, improvements in the recycling processes as well as environmental and regulatory considerations are driving airlines towards investing in the recycling of EOL aircrafts. Current market analyses (Fortune Business Insights (2025); Global Market Insights (2024); KPMG (2024)) estimate the global aircraft recycling market at USD 5 to 7 billion with 400 to 450 aircrafts dismantled annually. This market is projected to grow to around USD 14 billion by 2034 with more than 1000 aircraft expected to be retired annually.

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. Disassembly tasks mostly consist in the removal of pieces and materials from the aircraft but also include checks, activation and deactivation of systems and preparation work such as opening access panels and setting up scaffoldings. Tasks have variable durations going from 15 minutes to 16 hours. The order in which tasks are performed may be restricted by precedence constraints.

Tasks each require a given number of technicians that must be assigned in addition to the scheduling. Some tasks have certification requirements, which limit the technicians that may be assigned. Some technicians may also be unavailable at some times during the scheduling process. Additionally, place is limited in some parts of the aircraft which constrains the number of technicians able to work simultaneously in these locations and thus tasks that can be performed concurrently. Finally, the balance of the aircraft must be maintained during the whole disassembly process, which puts additional constraints on the order in which tasks where a certain quantity of mass is removed are done. The objective of the problem is to minimize the *makespan*, i.e. the time at which the last task of the schedule ends.

The problem studied in this paper is based on a real use case provided by an industrial partner. The data used for the experiments is based on the disassembly of a Boeing B737 plane, which consists in around 1500 tasks to schedule.

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 Constraint Programming (CP) approach presented in Thomas and Schaus (2024), a Mixed Integer Programming (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.

Subsection 1.1 discusses related work on similar scheduling problems arising in a dismantling context. Section 2 presents the problem and gives a small example. The CP model is presented and explained in Section 3. The MIP model is shown in Section 4. Section 5 presents the experiments done and their results. Finally, Section 6 offers some closing remarks as well as possible avenues for future research.

1.1 Related work

The aircraft disassembly scheduling problem is a variation of the Resource Constrained Project Scheduling Problem (RCPSP) (Brucker, Drexl, Möhring, Neumann, & Pesch, 1999; Özdamar & Ulusoy, 1995), 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 & Johnson, 1975). Several variants of the problem exist (Hartmann & Briskorn, 2022). The closest one to our 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, Feydy, and Schutt (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, Artigues, Lopez, Mönch, and Basini (2023), a combination of heuristic and metaheuristic approaches is used to solve a variant of the MSPSP with 50 tasks.

Dismantling an aircraft is the opposite of assembling it. Pucel and Roussel (2024) developed a constraint programming model for assembly line balancing of aircrafts with workstations, work zones within workstations, and cumulative resources shared across workstations. This model represents an extension of RCPSP with spatial divisions (workstations and zones) where capacity constraints limit concurrent operations in each spatial unit. More generally, the capacity of certain locations within the aircraft—expressed as the maximum number of technicians who can work simultaneously in a given area—can be interpreted as a standard renewable resource, and represents a special case of spatial resources (see De Boer (1998)).

Regarding the balancing constraints, the chemical tanker scheduling and tank allocation problem is also concerned with maintaining ship stability and trim equilibrium (see Vilhelmsen, Larsen, and Lusby (2016)). Le, Roussel, and Lecoutre (2025) introduced resource-constrained assembly line balancing for aircraft manufacturing where resource utilization should remain relatively balanced across production periods to ensure a stable workflow. Cargo optimization are also problems embedding center of gravity and load distribution constraints as core requirements (see Limbourg, Schyns, and Laporte (2012)). More generally the balance constraints in RCPSP problems can be modeled using cumulative or storage resources (see Neumann and Schwindt (2003)).

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, Ignacio, and Coelho (2022) propose an integer programming approach to schedule aircraft engine assembly lines that also involves workers with several skills on up to 100 tasks. In Niu, Xue, Zhong, Qiu, and Zhou (2023) the authors propose an approach to schedule technicians on short-term aviation maintenance processes (up to 48h). In Camelot, Baptiste, and Mascle (2013); Dayi, Afsharzadeh, and Mascle (2016); Srinivasan and Gadh (1999); Zhong, Youchao, Ekene Gabriel, and Haiqiao (2011) 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, Xirouchakis, and Zust (2002), a disassembly problem with capacity constraints is studied. The stochastic aspects of disassembly processes are studied in Bentaha, Battaïa, and Dolgui (2013) and Tian, Zhou, and Chu (2013). In Edis (2021); Hübner, Gerhards, Stürck, and Volk (2021); Kizilay (2022); Zwingmann, Ait-Kadi, Coulibaly, and Mutel (2008) several MILP and CP models are proposed to solve disassembly problems but are only able to solve instances up to 150-300 tasks, while our problem requires solving a variant of the RCPSP with up to 1500 tasks.

2 Problem

This section presents the aircraft disassembly problem, introduces a formal definition of the problem (Subsection 2.1) and illustrates the problem with an example (Subsection 2.2).

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, which 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 $\mathcal{P}_i$ referring to 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 may not have a mass value or specific requirements, in which case, these fields are empty. Furthermore, 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 mobilize the whole plane or technical team. In the latter case, these tasks must occur at a specific phase in the dismantling process, which is enforced by precedence constraints.

For example, after the reception of the aircraft, there is a series of basic maintenance checks, followed by a test run of the engines SKYbrary (2025). This procedure consist in a chain of tasks: fueling the plane, towing it to the test area, performing the power run, towing it back and purging its tanks. These tasks mobilize all the plane and must be performed in this strict order as enforced by precedences. Afterwards, the actual disassembly can start. Additional precedence constraints are used to make sure that any task that is part of the basic maintenance check must be done before the fueling of the plane, which marks the start of the test run procedure. Similarly, any disassembly task must be done after the purging of the tank at the end of the procedure.

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 that corresponds to one of the balance zones of the aircraft. There are four balance zones in total: `Aft` and `Fwd` that correspond to the rear

and front of the aircraft and `Left` and `Right` that 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.

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.

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.

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.

3 CP Model

This Section presents the constraint programming model proposed to solve the problem. Subsection 3.1 provides a brief background on constraint programming and introduces several modeling concepts used in the model. The variables used in the model are introduced in Subsection 3.2. Subsection 3.3 presents the model and details the constraints used.

3.1 Background

Constraint Programming (CP) is a powerful paradigm for modeling and solving complex scheduling problems (see Laborie, Rogerie, Shaw, and Vilím (2018)). Modern CP solvers provide specialized modeling constructs that enable the natural representation of scheduling problems through *optional interval variables* and *cumulative functions* introduced in Laborie and Rogerie (2008); Laborie, Rogerie, Shaw, Vilím, and Katai (2012).

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

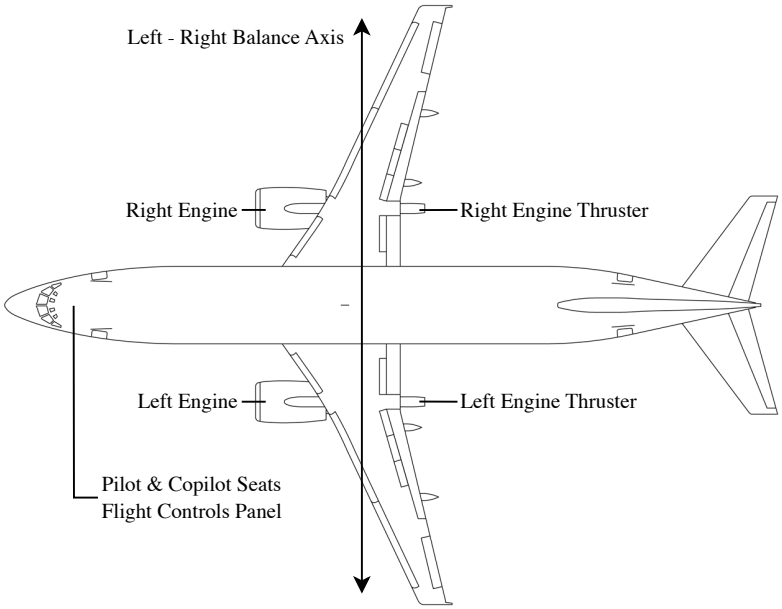

Fig. 1: Location of the tasks of Example 2.2

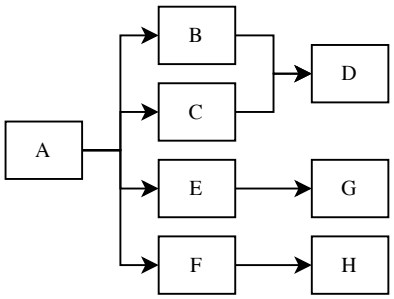

Fig. 2: Precedences between the tasks of Example 2.2

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

An *interval variable* represents a time interval during which an activity occurs. Each interval variable a is characterized by three attributes: a start time s_a, an end time e_a, and a duration d_a, where the relationship $s_a + d_a = e_a$ holds. An *optional interval variables* enables the interval variable to be present or absent state. Several

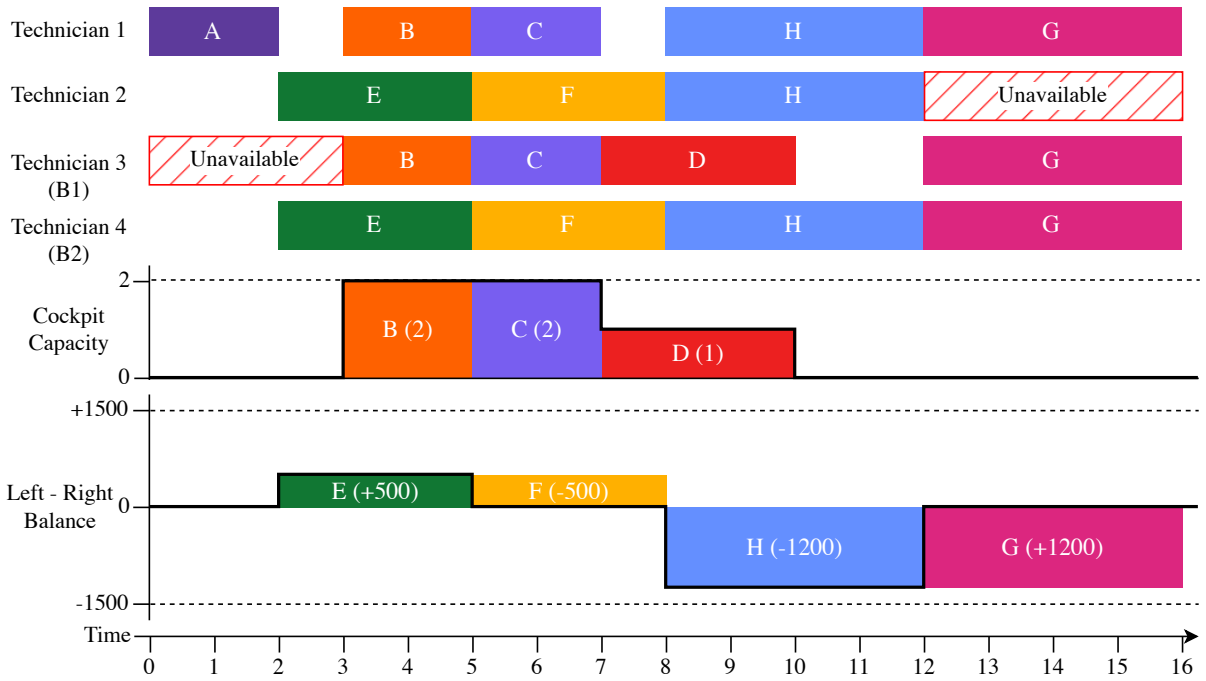

Fig. 3: Illustration of a solution for Example 2.2

global constraints are defined on optional interval variables:

- The constraint $\mathit{alternative}(a, \{a_1, \ldots, a_n\})$ ensures that if interval a is present, then exactly one interval from $\{a_1, \ldots, a_n\}$ is present, and that a starts and ends together with the chosen interval. If a is absent, all intervals in $\{a_1, \ldots, a_n\}$ are absent.
- The constraint $\mathit{alternative}(a, \{a_1, \ldots, a_n\}, c)$ ensures that if a is present, exactly c intervals from $\{a_1, \ldots, a_n\}$ are present, with all selected interval having the same start and end as a.
- The $\mathit{noOverlap}(\{a_1, \ldots, a_n\})$ constraint enforces that no two present intervals in the set overlap in time.

Cumulative functions provide a powerful mechanism for modeling resource usage over time (see Laborie et al. (2012); Schaus, Thomas, and Kameugne (2025)). A cumulative function is a piecewise constant function $f : \mathbb{Z} \to \mathbb{Z}$ that represents the aggregated contribution of activities to a resource level at each time point. Cumulative functions are constructed by combining *elementary cumulative function expressions*, which define the contribution of individual interval variables or fixed time intervals. The main elementary cumulative functions, illustrated on Figure 4, are:

- **pulse**(a, h): Represents the contribution of interval a with height h during its execution. If a is present with $a = [s_a, e_a)$, the function equals h for all $t \in [s_a, e_a)$ and 0 otherwise.
- **step**(t_0, h): Represents a constant contribution of height h starting at time t_0 and continuing indefinitely. For all $t \geq t_0$, the function equals h; otherwise it equals 0.
- **stepAtStart**(a, h): Represents a step function that changes by h at the start time of interval a. If a is present with $a = [s_a, e_a)$, the function equals h for all $t \geq s_a$ and 0 otherwise.

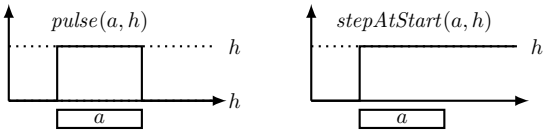

Fig. 4: Elementary cumulative functions.

Cumulative functions can be formed by combining functions using addition and subtraction: $f = \sum_i \epsilon_i \cdot f_i$ where $\epsilon_i \in \{-1, +1\}$ and each f_i is

also a cumulative function. When all interval variables involved in a cumulative function are fixed, the function itself becomes a fixed piecewise constant function. The constraint $0 \leq f \leq C$ (or $f \geq C$) ensures that the cumulative function f does not exceed (or fall below) capacity C at any time point.

3.2 Variables

Each task $i \in \mathcal{T}$ is modeled with an interval variable $a_i \in \mathcal{A}$. The start and end are initialized to $s_i = [0, H - d_i]$ and $e_i = [d_i, H]$ respectively where the horizon can be set to the sum of all tasks duration. 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 assignment of technician j to task i. 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 υ_{ju} that are set to the time windows corresponding to the unavailabilities.

3.3 Constraints

The model is written as:

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

subject to

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

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

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

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

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

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

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

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

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

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

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 υ_{ju} in the set of intervals that 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 than 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) that 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 functions 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, as ensured by constraints (10) and (11).

4 MIP Model

This section presents the MIP model proposed to solve the problem. Mixed Integer Programming (MIP) is a commonly used approach to model and solve combinatorial optimization problems. As mentioned in Section 1.1, MIP approaches have been successfully applied on similar problems.

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 Koné, Artigues, Lopez, and Mongeau (2011) in which we introduce the balance constraints as well as the resource assignment variables to capture technicians' skills and unavailabilities constraints.

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 modeled as additional tasks with fixed time windows to which the unavailable technicians are assigned. This set of additional tasks is denoted by $\mathcal{U}$. The set of all tasks is thus $\mathcal{T}' = \mathcal{T} \cup \mathcal{U}$. Let $\mathcal{E} = \{0, 1, \ldots, |\mathcal{T}'| - 1\}$ denote the index set of events e, which correspond to starts of a tasks. We further define $\mathcal{E}^0 = \mathcal{E} \setminus \{0\}$.

Variables are introduced in Subsection 4.1 while Subsection 4.2 details the constraints. Finally, Subsection 4.3 explains how the model from Koné et al. (2011) was adapted for the specific constraints of this problem and Subsection 4.4 discusses its complexity.

4.1 Variables

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

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

Additionally, auxiliary variables $z_{i,-1} = 0$ 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 binary variables x_{ij} for $i \in \mathcal{T}', j \in \mathcal{R}$ correspond to the assignment of technician j to task i:

$$x_{ij} = \begin{cases} 1 & \text{if technician } j \text{ is assigned to task } i \\ 0 & \text{otherwise} \end{cases} \qquad (i \in \mathcal{T}'; j \in \mathcal{R}) \quad (13)$$

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

$$y_{ije} = \begin{cases} 1 & \text{if technician } j \text{ is assigned to task } i \\ & \text{during event } e \\ 0 & \text{otherwise} \end{cases}$$

$$(i \in \mathcal{T}'; j \in \mathcal{R}, e \in \mathcal{E}) \quad (14)$$

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:

$$a_{ie} = \begin{cases} 1 & \text{if task } i \text{ starts at event } e \\ 0 & \text{otherwise} \end{cases}$$
$$(i \in \mathcal{T}'; e \in \mathcal{E}) \quad (15)$$

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 $e = -1$.

4.2 Constraints

The MIP model is written as:

$$\text{minimize } t_{max} \quad (16)$$

subject to

$$t_0 = 0 \quad (17)$$

$$t_e - t_{e-1} \geq 0 \quad (e \in \mathcal{E}^0) \quad (18)$$

$$z_{i,-1} = 0 \quad (i \in \mathcal{T}') \quad (19)$$

$$a_{ie} \geq z_{ie} - z_{i,e-1} \quad (i \in \mathcal{T}'; e \in \mathcal{E}) \quad (20)$$

$$a_{ie} \leq 1 - z_{i,e-1} \quad (i \in \mathcal{T}'; e \in \mathcal{E}) \quad (21)$$

$$\sum_{e \in \mathcal{E}} z_{ie} \geq 1 \quad (i \in \mathcal{T}') \quad (22)$$

$$\sum_{e'=0}^{e-1} z_{ie'} - e(1 - (z_{ie} - z_{i,e-1})) \leq 0 \quad (i \in \mathcal{T}'; e \in \mathcal{E}^0) \quad (23)$$

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

$$t_f - t_e - d_i(z_{ie} - z_{i,e-1} - (z_{if} - z_{i,f-1})) \geq -d_i \quad (i \in \mathcal{T}'; e, f \in \mathcal{E} \mid e < f) \quad (25)$$

$$t_{max} \geq t_e + d_i - H(1 - a_{ie}) \quad (i \in \mathcal{T}, e \in \mathcal{E}) \quad (26)$$

$$y_{ije} \leq x_{ij} \quad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \quad (27)$$

$$y_{ije} \leq z_{ie} \quad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \quad (28)$$

$$y_{ije} \geq x_{ij} + z_{ie} - 1 \quad (i \in \mathcal{T}'; j \in \mathcal{R}; e \in \mathcal{E}) \quad (29)$$

$$\sum_{j \in \mathcal{R}} y_{ije} = \tau_i z_{ie} \quad (i \in \mathcal{T}'; e \in \mathcal{E}) \quad (30)$$

$$\sum_{i \in \mathcal{T}'} y_{ije} \leq 1 \quad (j \in \mathcal{R}; e \in \mathcal{E}) \quad (31)$$

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

$$1 - x_{ij} \leq \sum_{e \in \mathcal{E}} z_{ie} - \sum_{e \in \mathcal{E}} y_{ije} \leq N(1 - x_{ij}) \quad (i \in \mathcal{T}'; j \in \mathcal{R}) \quad (33)$$

$$\sum_{j \in \mathcal{R}} x_{ij} = \tau_i \quad (i \in \mathcal{T}') \quad (34)$$

$$x_{uj} = 1 \quad (j \in \mathcal{R}, u \in \mathcal{U}_j) \quad (35)$$

$$t_e \geq s_{ju} a_{ue} \quad (j \in \mathcal{R}, u \in \mathcal{U}_j, e \in \mathcal{E}) \quad (36)$$

$$t_e \leq s_{ju} a_{ue} + H(1 - a_{ue}) \quad (j \in \mathcal{R}, u \in \mathcal{U}_j, e \in \mathcal{E}) \quad (37)$$

$$z_{pe} + \sum_{e'=0}^{e} z_{ie'} - e(1 - z_{pe}) \leq 1 \quad (i \in \mathcal{T}; p \in \mathcal{P}_i; e \in \mathcal{E}) \quad (38)$$

$$\sum_{j \in \mathcal{R} | c_{iq} \in \mathcal{C}_j} x_{ij} \geq n_{iq} \quad (i \in \mathcal{T}; q \in \mathcal{Q}_i) \quad (39)$$

$$\sum_{i \in \mathcal{T} | l_i = l} \tau_i z_{ie} \leq k_l \quad (l \in \mathcal{L}; e \in \mathcal{E}) \quad (40)$$

$$b_e^{af} = b_{e-1}^{af} + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Aft}} a_{ie} m_i + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Fwd}} a_{ie}(-m_i) \quad (e \in \mathcal{E}) \quad (41)$$

$$b_e^{lr} = b_{e-1}^{lr} + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Left}} a_{ie} m_i + \sum_{i \in \mathcal{T} | z_{l_i} = \texttt{Right}} a_{ie}(-m_i) \quad (e \in \mathcal{E}) \quad (42)$$

$$-B^{af} \leq b_e^{af} \leq B^{af} \quad (e \in \mathcal{E}) \quad (43)$$

$$-B^{lr} \leq b_e^{lr} \leq B^{lr} \quad (e \in \mathcal{E}) \quad (44)$$

Constraints (17) and (18) ensure that the timing of events follow an increasing order: $t_0 \leq t_1 \leq ... \leq t_{N-1}$. Constraints (19) initialize z_{ie} variables while constraints (22) ensure that each task is processed during at least one event. Constraints (20) and (21) link together the task start variable a_{ie} with the task processing variables z_{ie}. Constraints (23) and (24) make sure that each task is processed in a contiguous block of events: Constraints (23) make 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 ($\sum_{e'=0}^{e-1} z_{ie'} \leq 0$). Constraints (24) enforce 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 ($\sum_{e'=e}^{N-1} z_{ie'} \leq 0$).

Constraints (25) enforce the processing time of tasks based on task processing variables z_{ie} and time variables t_e. They ensure 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 than or equal to the duration of task i.

Constraint (26) ensures that the makespan objective (16) is not smaller than the completion time of any task.

Constraints (27) to (29) link together event assignment variables y_{ije} with assignment variables x_{ij} and task processing variables z_{ie} following the relation $y_{ije} = x_{ij} \cdot z_{ie}$. Constraints (30) link the task processing variables z_{ie} with the event assignment variables y_{ije} and ensure that the correct number of technicians is used when a task is processed. Constraints (31) prevent a technician to be concurrently assigned to more than one task. Constraints (32) link assignment variables x_{ij} with event assignment variables y_{ije}. Constraints (33) make 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} has a value of 1 or 0 if $x_{ij} = 0$. Constraints (34) ensure that the correct number of technicians is assigned to each task.

Constraints (35) fix the unavailability tasks to the corresponding technician. Constraints (36) and (37) fix the events of the the unavailability tasks. Constraints (38) set 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.

Skill Requirements

Constraints (39) ensure that requirements are respected: For each requirement $q \in \mathcal{Q}_i$ of each task $i \in \mathcal{T}$, a 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 than or equal to the requested amount n_{iq}.

Locations Capacity

Constraints (40) model 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 (41) and (42) 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:

$$\sum_{i \in \mathcal{T} \mid z_{l_i} = \texttt{Aft/Left}} a_{ie} m_i \quad + \sum_{i \in \mathcal{T} \mid z_{l_i} = \texttt{Fwd/Right}} a_{ie}(-m_i) \tag{45}$$

Finally, constraints (43) and (44) enforce that the balance variables b_e^{af} and b_e^{lr} stay within the balance range at each event e.

4.3 Differences from the OOE Model

A key difference from the model of Koné et al. (2011) is the introduction of the resource assignment variables x_{ij} and y_{ije}, which capture additional constraints related to technicians' skills and unavailabilities.

The inclusion of technician unavailability tasks, modeled as fixed activities that do not contribute to the makespan, also required a careful adaptation of the formulation proposed by Koné et al. (2011). In their model, the makespan constraints can be expressed as

$$t_{\max} \geq t_e + d_i(z_{ie} - z_{i,e-1}) \quad (i \in \mathcal{T},\, e \in \mathcal{E}),$$

but this expression is not directly applicable in our case, since unavailability tasks are also associated with events and time intervals.

Regarding the balancing constraint, it might be tempting—but incorrect—to formulate it as

$$-B^{af} \leq \sum_{i \in \mathcal{T} \mid z_{l_i} = \texttt{Aft}} z_{ie} m_i + \sum_{i \in \mathcal{T} \mid z_{l_i} = \texttt{Fwd}} z_{ie}(-m_i) \leq B^{af} \quad (e \in \mathcal{E}),$$

since, in the case of activities with identical start times, the model could artificially choose event

assignments that satisfy the balance constraints without reflecting a valid configuration. In our formulation, the weight differences are accumulated across start events through the variables a_{ie}, ensuring that the model cannot "cheat" regardless of the event assignments. If a feasible solution exists, then there is at least one corresponding event assignment for which the balance constraints are satisfied.

4.4 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 where $\mathcal{P} = \cup_{i\in\mathcal{T}}\mathcal{P}_i$ is the union of the sets of precedences of each task $i \in \mathcal{T}$. 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.

5 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.

Subsection 5.1 explains how industrial data was used to generate the instances used for the experiments. The experimental setup for the two experiments is detailed in Subsection 5.2. The metric used to compare the models is explained in Subsection 5.2.1 Finally, Subsection 5.3 presents the results of the two experiments.

5.1 Data

The instances used in the experiments are based on real industrial data provided by our partner within the Planum research project. The dataset was collected during the complete dismantling of a Boeing 737NG-600 aircraft, and it describes 1454 disassembly tasks representing the full set of operations performed during the process.

Constructing this dataset required a substantial collaborative effort. Since no structured numerical data were available, the information had to be manually reconstructed from technical documentation and user manuals to extract task precedences, durations, and other relevant attributes. In addition, a detailed labeling of operations was carried out during the actual dismantling to ensure that the model accurately reflects the real industrial process.

Most of the task data originate from this source, with the exception of the mass values used for the balance constraints. Because the industrial partner is still in the process of collecting precise mass information, these values were approximated artificially: a mass between 5 kg and 500 kg was assigned to 214 tasks located in the four balance zones, while the remaining tasks were considered to have no mass impact. The maximum allowable mass difference was set to 300 kg on the Aft–Forward axis and 500 kg 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 5 shows a visualization of some of the tasks features of the instance B737NG600-1454. Tasks are sorted by duration then grouped by

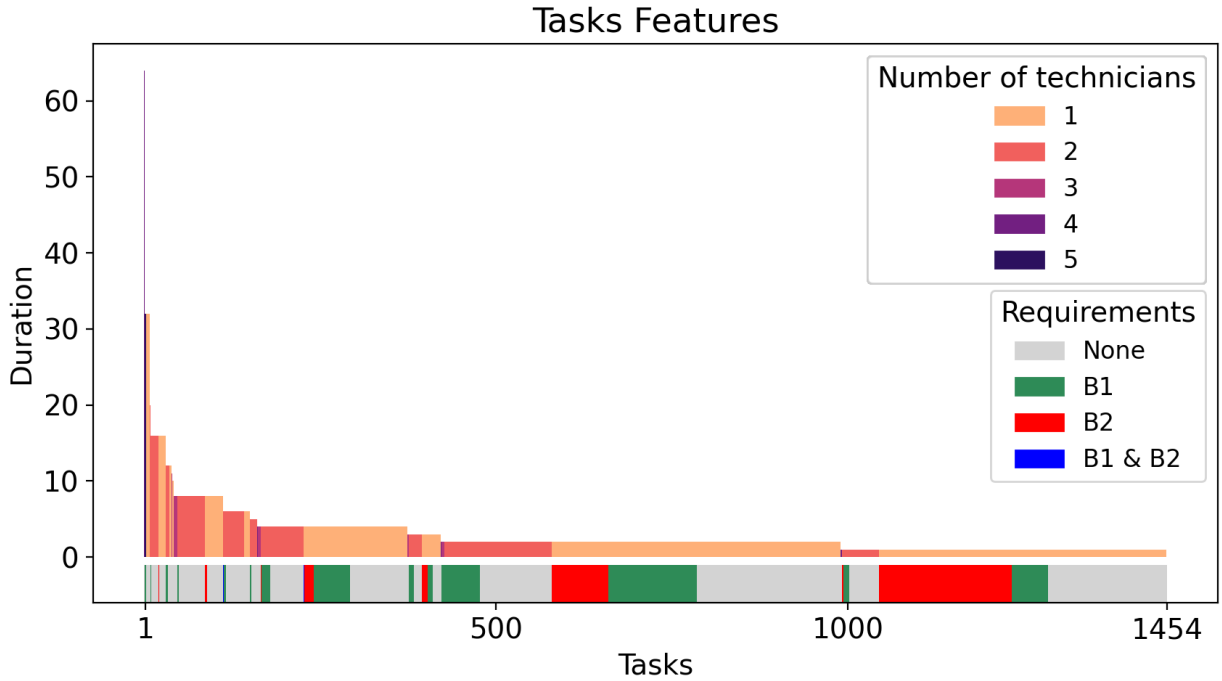

Fig. 5: Durations and number of technicians for the tasks of instance B737NG600-1454

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 approximately 22% of tasks with a B1 requirement, approx. 20% 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.

5.2 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 (denoted by tmt) of 3600 seconds and a maximum memory limit of 8GB.

5.2.1 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, which 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:

$$\gamma(x) = \begin{cases} 0 & \text{if } o(x^*) = o(x) = 0 \\ 1 & \text{if } o(x^*) \cdot o(x) < 0 \\ \frac{|o(x^*) - o(x)|}{max(|o(x^*)|, |o(x)|)} & \text{otherwise} \end{cases} \tag{46}$$

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:

$$p(t) = \begin{cases} 1 & \text{if no incumbent until } t \\ \gamma(x(t)) & \text{otherwise} \end{cases} \tag{47}$$

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:

$$P(T) = \int_{t=0}^{T} p(t)dt = \sum_{i=1}^{I} p(t_{i-1}) \cdot (t_i - t_{i-1}) \tag{48}$$

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 $t_I = T$.

5.3 Computational results

Models comparison

This first experiment consists 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 $P(tmt)$ shows the primal integral (computed with $tmt = 3600$); 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

Table 3: Results on the problem with all constraints

Instance		CP			MIP		
Name	$obj*$	$P(tmt)$	obj	$t*$	$P(tmt)$	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

the number of tasks. Additional experiments have shown that this behavior stays the same on 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 6 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.

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. One possible explanation is the characteristics of the instance, which consists of many small tasks with short durations, each requiring only one or two technicians. This 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.

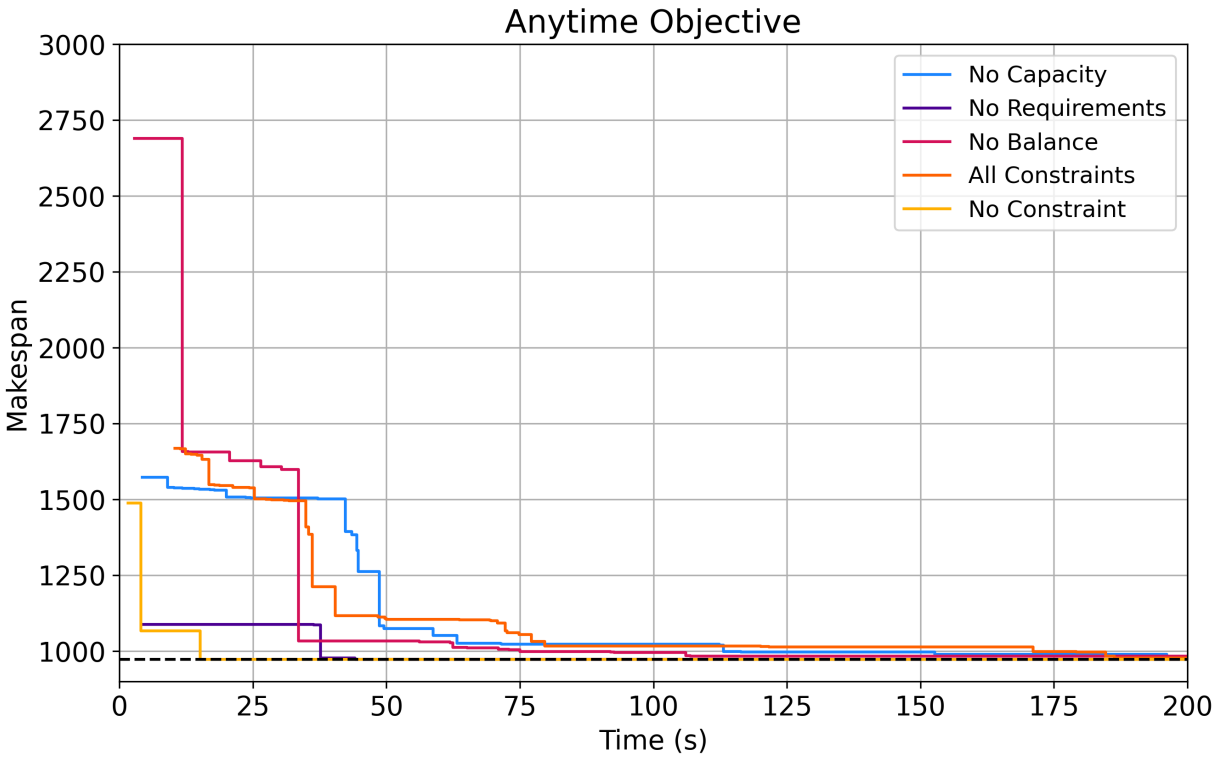

Fig. 6: Anytime performances of the CP approach on the instance B373NG600-1454

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 dominate the version with all constraints. The lower impact of these constraints can again be explained by the large number of small tasks, which offer a lot of flexibility in the scheduling.

6 Conclusion

We presented the aircraft disassembly scheduling problem. This problem, which consists in scheduling 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 in some parts of 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 CP model vastly outperforms the MIP model and is able to find feasible solutions for 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.

6.0.1 Future Work

A possible future research avenue would be to investigate metaheuristic methods to solve the problem, as these methods have been successfully applied to variants of the RCPSP (Laurent, Deroussi, Grangeon, & Norre, 2017; Mischek & 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, Gefflot, Beck, et al. (2001).

References

Asmatulu, E., Overcash, M., Twomey, J. (2013). Recycling of aircraft: State of the art in 2011. *Journal of Industrial Engineering*, *2013*(1), 960581,

Bellenguez, O., & Néron, E. (2004). Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. *International conference on the practice and theory of automated timetabling* (pp. 229–243).

Bentaha, M.L., Battaïa, O., Dolgui, A. (2013). Chance constrained programming model for stochastic profit–oriented disassembly line balancing in the presence of hazardous parts. *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* (pp. 103–110).

Berthold, T. (2013). Measuring the impact of primal heuristics. *Operations Research Letters*, *41*(6), 611-614, https://doi.org/https://doi.org/10.1016/j.orl.2013.08.007

Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, *112*(1), 3–41, https://doi.org/10.1016/S0377-2217(98)00204-5

Camelot, A., Baptiste, P., Mascle, C. (2013). Decision support tool for the disassembly of reusable parts on an end-of-life aircraft. *Proceedings of 2013 international conference on industrial engineering and systems management (iesm)* (pp. 1–8).

da Matta Oliveira Borsato Pinhão, J., Ignacio, A.A.V., 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, https://doi.org/10.1016/j.cie.2022.108652

Davenport, A.J., Gefflot, C., Beck, J.C., et al. (2001). Slack-based techniques for robust schedules. *Proceedings of the sixth european conference on planning (ecp-2001)* (pp. 7–18).

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

De Boer, R. (1998). *Resource-constrained multi-project management* (Unpublished doctoral dissertation). PhD thesis, University of Twente, The Netherlands.

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

Fortune Business Insights (2025). *Commercial aircraft disassembly, dismantling and recycling market size, share & covid-19 impact analysis.* (https://www.fortunebusinessinsights.com/commercial-aircraft-disassembly-dismantling-and-recycling-market-103584 [Accessed : 31 october 2025])

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

Global Market Insights (2024). *Aircraft recycling market - by aircraft, type, by material & forecast, 2025 - 2034.* (https://www.gminsights.com/

industry-analysis/aircraft-recycling-market [Accessed : 31 october 2025])

Habib, M.A., Subeshan, B., Kalyanakumar, C., Asmatulu, R., Rahman, M.M., Asmatulu, E. (2025). Current practices in recycling and reusing of aircraft materials and equipment. *Materials Circular Economy*, *7*(1), 1–36,

Hartmann, S., & 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, https://doi.org/10.1016/j.ejor.2021.05.004

Hübner, F., Gerhards, P., Stürck, C., Volk, R. (2021). Solving the nuclear dismantling project scheduling problem by combining mixed-integer and constraint programming techniques and metaheuristics. *Journal of Scheduling*, *24*(3), 269–290,

IATA (2025). *Global outlook for air transport - june 2025.* (https://www.iata.org/en/iata-repository/publications/economic-reports/global-outlook-for-air-transport-june-2025/ [Accessed : 31 october 2025])

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,

Koné, O., Artigues, C., Lopez, P., Mongeau, M. (2011). Event-based milp models for resource-constrained project scheduling problems. *Computers & Operations Research*, *38*(1), 3–13,

KPMG (2024). *Circularity in flight.* (https://kpmg.com/ie/en/insights/aviation/circularity-in-flight-fs-aviation.html [Accessed : 31 october 2025])

Laborie, P., & Rogerie, J. (2008). Reasoning with conditional time-intervals. *Flairs conference* (pp. 555–560).

Laborie, P., Rogerie, J., Shaw, P., 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., 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*, 111–143,

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

Le, D.A., Roussel, S., Lecoutre, C. (2025). Aircraft resource-constrained assembly line balancing with learning effect: A constraint programming approach. *31st international conference on principles and practice of constraint programming (cp 2025)* (pp. 25–1).

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

Limbourg, S., Schyns, M., Laporte, G. (2012). Automatic aircraft cargo load planning. *Journal of the Operational Research Society*, *63*(9), 1271–1283,

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

Neumann, K., & Schwindt, C. (2003). Project scheduling with inventory constraints. *Mathematical Methods of Operations Research*, *56*(3), 513–533,

Niu, B., Xue, B., Zhong, H., Qiu, H., Zhou, T. (2023). Short-term aviation maintenance technician scheduling based on dynamic task disassembly mechanism. *Information Sciences*, *629*, 816-835, https://doi.org/10.1016/j.ins.2023.01.137

Polo-Mejía, O., Artigues, C., Lopez, P., Mönch, L., 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,

Pucel, X., & Roussel, S. (2024). Constraint programming model for assembly line balancing and scheduling with walking workers and parallel stations. *30th international conference on principles and practice of constraint programming.*

Sabaghi, M., Cai, Y., Mascle, C., Baptiste, P. (2016). Towards a sustainable disassembly/dismantling in aerospace industry. *Procedia CIRP*, *40*, 156–161,

Schaus, P., Thomas, C., Kameugne, R. (2025). Implementing cumulative functions with generalized cumulative constraints. *arXiv preprint arXiv:2508.01751*, ,

Scheelhaase, J., Müller, L., Ennen, D., Grimme, W. (2022). Economic and environmental aspects of aircraft recycling. *Transportation Research Procedia*, *65*, 3–12,

Shan, S., Hu, Z., Liu, Z., Shi, J., Wang, L., 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,

SKYbrary (2025). *Aircraft ground running.* (https://skybrary.aero/articles/aircraft-ground-running [Accessed : 10 november 2025])

Srinivasan, H., & Gadh, R. (1999). Selective disassembly of components with geometric constraints. *International design engineering technical conferences and computers and information in engineering conference* (Vol. 19746, pp. 571–579).

Thomas, C., & Schaus, P. (2024). A constraint programming approach for aircraft disassembly scheduling. *International conference on the integration of constraint programming, artificial intelligence, and operations research* (pp. 211–220).

Tian, G., Zhou, M., 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,

Vilhelmsen, C., Larsen, J., Lusby, R. (2016). A heuristic and hybrid method for the tank allocation problem in maritime bulk shipping. *4OR*, *14*(4), 417–444,

Young, K.D., Feydy, T., Schutt, A. (2017). Constraint programming applied to the multi-skill project scheduling problem. *Principles and practice of constraint programming: 23rd international conference, cp 2017, melbourne, vic, australia, august 28–september 1, 2017, proceedings 23* (pp. 308–317).

Zhong, L., Youchao, S., Ekene Gabriel, O., 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., Mutel, B. (2008). Optimal disassembly sequencing strategy using constraint programming approach. *Journal of Quality in Maintenance Engineering*, *14*(1), 46–58,

Özdamar, L., & Ulusoy, G. (1995). A survey on the resource-constrained project scheduling problem. *IIE Transactions*, *27*(5), 574-586, https://doi.org/10.1080/07408179508936773