Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 95 (2026) 169–176

Transportation Research Procedia

www.elsevier.com/locate/procedia

Euro Working Group on Transportation Annual Meeting 2025 - EWGT2025

Automated repair of input data for fleet assignment in a digital airline twin

Milena Röhrs[a,*], Sarah Albrecht[a], Thorsten Ehlers[a], Klaus Lütjens[a]

[a]*Institute of Air Transport, German Aerospace Center (DLR), Blohmstr. 20, 21079 Hamburg, Germany.*

Abstract

Digitalization has great potential for making air transport more efficient and sustainable, e.g. by holistic optimization in order to improve the usage of resources. This applies in particular to processes within an airline. In this context, an automated decision-making tool with a high degree of integration and compatibility to different business strategies could enable airlines, politicians, and research institutions to assess various scenarios. For example, it could help determine how new aircraft types can be integrated into an airline's fleet or how flight demand can be met with limited resources. For this purpose, a digital airline twin (DAT) is currently being created by the German Aerospace Center (DLR). One major difficulty is that the input data, such as airline schedules, which the DAT typically receives from databases, is often neither complete nor flawless. Hence, there is a need for automatic procedures for dealing with these imperfect data sets. This study presents two algorithms for making schedules periodic through the addition of repositioning legs. One approach is more profit-driven while the other one aims to stay as close to the original schedule as possible. Comparing and evaluating both, the second algorithm turned out to be the better option for the DAT as the necessary computational effort is smaller and the results are at least as good as those of the first algorithm.

Keywords: Data analytics; automated data repair; decision-support tool; fleet assignment; digital twin

1. Introduction

In the 21st century the aviation sector is put to the test by challenges such as the climate crisis, increasing demand for flights and staff shortages. New decision support frameworks for airlines, researchers and politicians are crucial for navigating these challenges. One such framework is the digital airline twin (DAT) which is being designed and implemented at the German Aerospace Center (DLR). Given airline-specific input data, the DAT will automatically execute several integrated and/or sequential optimization procedures. Those procedures represent all relevant airline planning processes starting with the question of which airports to include in the network and ending with the actual flight execution. During execution, the results of the long-term planning processes are used as inputs for subsequent

* Corresponding author. Tel.: +49 40 2489641 225.
E-mail address: milena.roehrs@dlr.de

2352-1465
10.1016/j.trpro.2026.02.022

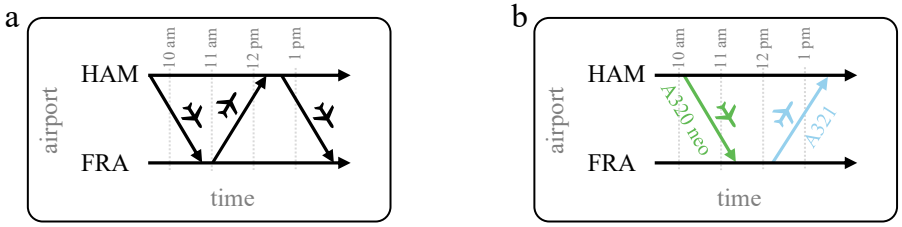

Fig. 1. Time-space networks of imbalanced schedules: (a) wrong schedule interval/missing return flight; (b) inconsistent labelling.

planning steps. This way, the DAT will provide a deeper understanding of both the interaction of planning processes and a variety of use cases. One specific planning process is called fleet assignment. It involves determining the most suitable aircraft type to operate each flight leg, considering e.g. passenger demand, expected revenue, aircraft capacity, and operational costs as discussed by Sherali et al. (2006). Here, a flight leg is defined by its origin airport, departure time, destination airport, and arrival time. The fleet assignment tool should be able to take collections of flight legs, called schedules, from external databases as input. This real-world data often comes with the challenge of being inconsistent, incomplete or in other ways not fulfilling the models requirements.

For example, schedules with assigned aircraft types are commonly required to be operable periodically, meaning they repeat at consistent intervals (e.g., every x days). This periodicity simplifies planning fleet-dependent processes, such as maintenance, crew management, and gate allocation, as discussed for a daily schedule by Gopalan and Talluri (1998). Periodicity necessitates the initial fleet distribution among the airports to be restored by the end of the schedule. In the following, schedules that fulfil this criterion for all airports and all aircraft types will be called 'balanced'.

There are several reasons why schedules extracted from real data are not always balanced. First is the choice of schedule interval. If, for example, the schedule in Fig. 1a ended at 12:30 pm, it would be balanced. However, since it continues until 2 pm, it is not. Moreover, imbalance can occur because not all schedules are intended to be completely periodic. This is particularly the case when e.g. switching between summer and winter flight schedules. Nevertheless, it can sometimes be observed in the data that in the course of a month many more aircraft of a certain type depart from an airport than arrive while at another airport the opposite trend can be observed. Those implausible trends indicate inadequacies in the data set such as missing flights (see Fig. 1a), e.g. due to maintenance or transfer flights, or inconsistent labelling of the aircraft type (see Fig. 1b), caused e.g. by manual recording. To address this issue, we present two repair algorithms that automatically prepare real-world flight schedules for the fleet assignment process. These algorithms handle mislabeled and missing flights by making minor adjustments to the assigned aircraft types as well as by adding new flight legs as needed.

2. Literature review and contribution

Approaches similar to parts of our repair algorithms have already been applied in the literature. For example, Caetano and Gualda (2010) obtained both a periodic schedule and a fleet assignment at the same time generating repositioning flights if the flight plan otherwise cannot be executed cost-effectively. Similarly, Lohatepanont and Barnhart (2004) presented an algorithm that creates a schedule via deletions and additions of legs to a base schedule integrated in the fleet assignment process. To do so, they differentiated between mandatory and optional legs, a strategy that was also employed in multiple follow-up studies. This is similar to allowing optional teleportations to deal with imbalances as done in the first of our algorithms (A1).

Also, the concept of the deficit function (DF) was used in the past e.g. for inserting deadheading trips for minimizing the size of the required fleet in bus transport and aircraft routing (see Ceder and Stern (1981); Stern and Gertsbakh (2019)). To the authors knowledge, however, it has never been used in an algorithmic sequence for preparing input for fleet assignment as done in this paper (A2). Furthermore, neither of these concepts have been implemented and evaluated in the context of a DAT. And finally, the paper contributes to the current state of knowledge by comparing these two approaches.

3. Methodology

Fig. 2 provides an overview of the steps involved in the two repair algorithms presented by the authors. Both are based on a time-space network (TSN) representation of schedules similar to the one presented by Hane et al. (1995). Such a network (see Fig. 1) is a directed graph. Arrivals and departures are represented by the set of its nodes N. The vertical position i of a node (t, i) represents an element of the set of airports A and the horizontal position t indicates the time within the schedule interval. The possibilities of aircraft to either fly to another airport or to remain stationary are represented by arcs. Due to the different turnaround times, the flight time to another airport depends on the aircraft type $k \in K$ (set of aircraft types).

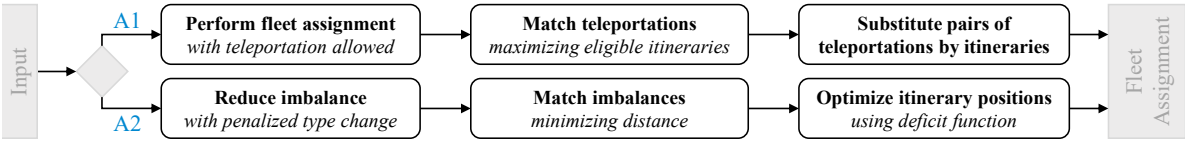

Fig. 2. Sequence of the processing steps in the two alternative repair algorithms (A1 and A2).

3.1. Repair Algorithm 1 (A1)

The basic idea of this algorithm is to first introduce teleportations to the schedule that will later be replaced by realizable flights. For determining the necessary teleportations, the algorithm optimizes a modified version of the basic fleet assignment model (FAM) defined by Hane et al. (1995) with the modification being as follows: A virtual airport called the 'teleport' θ is added to the TSN. Nodes are added to θ whenever a flight event on an unbalanced airport occurs. In analogy to Gu et al. (1994), the (im)balance ξ_i^k with respect to the whole schedule interval $(t_{start}, t_{end}] \subset \mathbb{R}$ (or a smaller time-frame $(t_1, t_2]$) is determined for airport i and aircraft type k by subtracting the aircraft departing during this time from those arriving. Depending on whether the balance is positive or negative, a 'teleport'-arc ranges from the flight event to the new node at the teleport or the other way around. Formally, the set of nodes and edges of the TSN is extended by:

$$\begin{aligned} N^{\text{new}} &:= \{(t, \theta) \mid \exists i \in A, k \in K : \xi_i^k(t_{\text{start}}, t_{\text{end}}) \neq 0 \wedge (t, i) \in N\}, \\ E_{\rightarrow}^{\text{new}} &:= \{((t, i), (t, \theta)) \mid (t, i) \in N \wedge \exists k \in K : \xi_i^k(t_{\text{start}}, t_{\text{end}}) > 0\}, \\ E_{\leftarrow}^{\text{new}} &:= \{((t, \theta), (t, i)) \mid (t, i) \in N \wedge \exists k \in K : \xi_i^k(t_{\text{start}}, t_{\text{end}}) < 0\}. \end{aligned} \tag{1}$$

As for all other airports, the nodes of the teleport are ordered by their time t, ground arcs are added between neighboring nodes and a wraparound arc connects the last with the first node. The additional arcs $E^{\text{new}} := E_{\rightarrow}^{\text{new}} \cup E_{\leftarrow}^{\text{new}}$ allow for relocating aircraft at every arrival and departure during a profit-driven reassignment. The usage of E^{new} indicates when and where aircraft are missing to be able to execute the resulting assignment. Later, repositioning flights are added at these positions. While flight arcs need to be covered by exactly one aircraft, a teleport arc $e \in E^{\text{new}}$ can be used for all $k \in K$ by any non-negative number of aircraft, described by the variable $p_{e,k} \in \mathbb{N}_0$. This additional aircraft flow must be considered in the balance constraint (see Eq. 3b). To ensure that teleportations are used only when necessary, their utilization is penalized. The penalty per aircraft and teleportation is determined by multiplying a base penalty ρ (e.g. chosen as 1 million €) by a factor γ_e, that linearly increases from one to two during the schedule interval for $e \in E_{\rightarrow}^{\text{new}}$ and decreases the other way around for $e \in E_{\leftarrow}^{\text{new}}$. This progress dependent penalty factor reduces the symmetry of the search space and ensures that the aircraft are withdrawn from airports as early as possible and brought back as late as possible. By doing so, the length of the teleportation interval, which is defined in Eq. 2, is increased. A large teleport interval is beneficial when later substituting the teleportations with realistic repositioning flights (see Fig. 3).

$$h : E_{\rightarrow}^{\text{new}} \times E_{\leftarrow}^{\text{new}} \to \mathbb{R}, \quad (((t_1, i), (t_1, \theta)), ((t_2, \theta), (t_2, i))) \mapsto \max(0, t_2 - t_1) \tag{2}$$

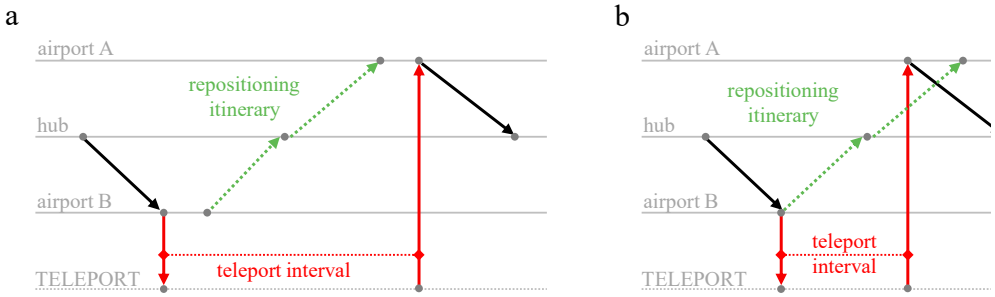

Fig. 3. (a) Eligible pair of teleportations that can be substituted by a repositioning itinerary within the teleport interval. (b) Teleport interval is smaller than the time necessary for the itinerary.

Now, let Y be the set of all ground arcs, F the set of all flight legs, $Z_l \subseteq K$ the set of aircraft types able to fly the leg $l \in F$, IN_n the set of incoming arcs of node $n \in N$, OUT_n the set of outgoing arcs of node n, CG the wrap around ground arcs, $CL(k)$ the wrap around flying arcs with turnaround for the aircraft type k and M^k the number of aircraft of type $k \in K$ available in the fleet. The cost of a aircraft type k assigned to leg l is given by the parameter $c_{l,k}$. The resulting optimization problem is given by:

$$\min \quad \sum_{l \in F} \sum_{k \in Z_l} f_{l,k} \cdot c_{l,k} + \sum_{e \in E^{\text{new}}} \sum_{k \in K} p_{e,k} \cdot \gamma_e \cdot \rho$$

$$\text{s.t.} \quad \sum_{k \in Z_l} f_{l,k} = 1 \quad \forall l \in F \quad \text{(Cover constraint)}, \tag{3a}$$

$$\sum_{l \in \text{IN}_n \cap F} f_{l,k} + \sum_{a \in \text{IN}_n \cap Y} y_{a,k} + \sum_{e \in \text{IN}_n \cap E^{\text{new}}} p_{e,k} - \sum_{l \in \text{OUT}_n \cap F} f_{l,k} - \sum_{a \in \text{OUT}_n \cap Y} y_{a,k} - \sum_{e \in \text{OUT}_n \cap E^{\text{new}}} p_{e,k} = 0 \quad \forall n \in N, \forall k \in K \quad \text{(Balance constraint)}, \tag{3b}$$

$$\sum_{a \in CG} y_{a,k} + \sum_{i \in CL(k)} f_{i,k} \leq M^k \quad \forall k \in K \quad \text{(Count constraint)}. \tag{3c}$$

Here, the variable $f_{l,k} \in \{0, 1\}$ indicates if aircraft type $k \in Z_l$ is assigned to leg $l \in F$ and the variable $y_{g,k} \in \mathbb{N}_0$ describes the number of aircraft of type $k \in K$ standing at ground arc $g \in Y$. The constraints fulfil coverage, balance and consistency with the available fleet size while minimizing assignment costs just as their counterparts in Hane et al. (1995). The main difference is that the objective is appended by a term that penalizes the use of teleportation.

After the solution of this optimization problem is obtained, the chosen teleportations are transformed into flight legs that are consistent with the airlines historic schedules. To do so, for each aircraft type k a complete bipartite graph G_k (see Fig. 4a) is created. The first set of nodes N_{orig} is constructed by creating for each $e \in E^{\text{new}}_{\rightarrow}$, with $e = ((t, i), (t, \theta))$, as many nodes associated with airport i and time t as aircraft of type k are withdrawn from i via e (given by $p_{e,k}$). The second set N_{dest} is created analogously from the set $E^{\text{new}}_{\leftarrow}$. This way, the number of nodes in G_k associated with an airport i equals $|\xi^k_i(t_{start}, t_{end})|$ of the reassigned schedule. An airport appearing in N_{orig} has a surplus of aircraft of type k (positive imbalance) at the end of the schedule and being in the second set indicates a deficit (negative imbalance). Airports that are balanced regarding k do not appear in G_k at all. The way the graph is created induces a function from $N_{\text{orig}} \times N_{\text{dest}}$ to $E^{\text{new}}_{\rightarrow} \times E^{\text{new}}_{\leftarrow}$. By combining this map with h (see Eq. 2), each origin-destination pair has an associated teleport interval length. Additionally, the algorithm calculates the shortest path between those pairs in the airlines spatial network. This network consists of airports as nodes, historical legs as edges and their distance as edge weight. Note here that the resulting shortest paths are saved. Part of them are later used as the artificial flights that have to be added to overcome the imbalance of aircraft. For each origin-destination pair, the algorithm computes the minimal flight time required to operate its shortest-path itinerary using an aircraft of type k. This minimal flight time

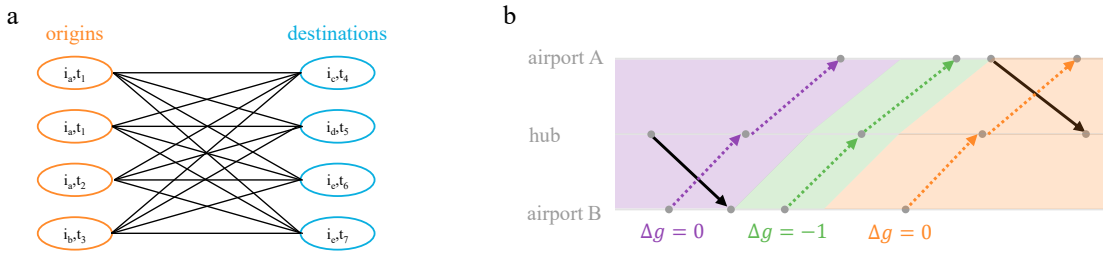

Fig. 4. (a) Bipartite graph G_k with $k \in K$ for matching positively and negatively imbalanced airports. (b) Net effect that adding the same itinerary at different positions of the schedule has on the size of the required fleet.

is compared with the teleport interval size defined above. A pair $m \in N_{\text{orig}} \times N_{\text{dest}}$ is considered eligible if the teleport interval exceeds the flight time along the shortest path, as in this case the aircraft would arrive at the destination airport on time. The weight of an eligible edge is defined as 1 and weights of other edges equal zero. The algorithm computes a complete matching with maximum weight on this graph. The matching determines which itineraries to add to the schedule. Being complete implies that the number of chosen itineraries starting from $i \in A$ equals the number of surplus aircraft at i. Same goes for the number of itineraries arriving at a certain destination airport and its aircraft deficit. Thus, adding the itineraries creates a balanced schedule. A maximum matching in this graph corresponds to a maximum number of eligible pairs. Eligible matches are added to the schedule in the middle of the teleport interval (see Fig. 3a). Non-eligible itineraries are added right at the time of the outbound teleportation (see Fig. 3b).

3.2. Repair Algorithm 2 (A2)

A second, alternative repair procedure, that excludes the influence of assignment dependent cost of operation (ADCO), base revenue and teleportation, is presented in the following. The aircraft types are only differentiated by their IATA-codes instead of more detailed characteristics. This decreases the number of variables and simplifies the implementation. Furthermore, we can define a very simple dissimilarity measure: The dissimilarity $\tilde{c}_{l,k}$ is zero if the IATA code of the aircraft type formerly assigned to the leg l and the IATA-code of $k \in K$ are the same. If their first difference is in the first, second or third letter, their dissimilarity is set to be 1, 0.5 or 0.25 respectively. The teleport-arcs of the network flow problem of A1 are substituted by a source and a sink to deal with imbalance: Two nodes are added to the set of nodes N, the source $\eta_\uparrow$ and the sink $\eta_\downarrow$, which are connected by one arc. For each airport $a \in A$, one arc is added from the source to the first node ϕ_a and another arc from the last node λ_a to the sink. The source-sink flow is penalized by a factor of 30. This penalty, together with the overall dissimilarity, forms the objective function.

$$\min \quad \sum_{l\in F}\sum_{k\in K} f_{l,k}\cdot\tilde{c}_{l,k} + \sum_{k\in K} s_k \cdot 30$$

$$\text{s.t.} \quad \sum_{k\in Z_l} f_{l,k} = 1 \quad \forall l \in F \quad \text{(Cover constraint)}, \tag{4a}$$

$$\sum_{l\in \text{IN}_n\cap F} f_{l,k} + \sum_{e\in \text{IN}_n\cap Y} y_{e,k} - \sum_{l\in \text{OUT}_n\cap F} f_{l,k} - \sum_{e\in \text{OUT}_n\cap Y} y_{e,k} + \delta_{n,\phi_a}\cdot \text{fsv}_{k,a} - \delta_{n,\lambda_a}\cdot \text{ttv}_{k,a} = 0 \quad \text{with } n=(a,t),\ \forall n \in N\setminus\{\eta_\uparrow,\eta_\downarrow\},\ \forall k\in K \quad \text{(Balance constraint)}, \tag{4b}$$

$$\sum_{k\in K} s_k + \sum_{i\in CL(k)} f_{i,k} \le M^k \quad \forall k \in K \quad \text{(Count constraint)}, \tag{4c}$$

$$s_k - \sum_{a\in A} \text{fsv}_{a,k} = 0 \quad \forall k \in K \quad \text{(Source balance)}, \tag{4d}$$

$$\sum_{a \in A} \text{ttv}_{a,k} - s_k = 0 \quad \forall k \in K \quad \text{(Sink balance)}, \tag{4e}$$

$$\begin{aligned} \text{ttv}_{k,a} &\leq \text{ttuv}_{k,a} \cdot M_k, \\ \text{fsv}_{k,a} &\leq \text{fsuv}_{k,a} \cdot M_k, \quad \forall k \in K, \forall a \in A \quad \text{(Sink \& source usage constraints)}. \\ \text{ttuv}_{k,a} + \text{fsuv}_{k,a} &\leq 1 \end{aligned} \tag{4f}$$

The variables denoted as in A1 were extended by ones that describe the sink and source flow: $\text{fsuv}_{k,a} \in \{0,1\}$ and $\text{ttuv}_{k,a} \in \{0,1\}$ indicate whether there is any flow from source or to sink for a given airport $a \in A$ and aircraft type $k \in K$. $\text{fsv}_{k,a} \in \mathbb{R}_{\geq 0}$ and $\text{ttv}_{k,a} \in \mathbb{R}_{\geq 0}$ represent how many aircraft come from the sink and go to the source, respectively. Finally, $s_k \in \mathbb{N}_0$ counts how many aircraft of type $k \in K$ flow through the source node in total. Coverage of every flight, fleet type balance and upper bounds for the number of used aircraft are enforced just as in A1 (see Eq. 4a - 4c). The symbol δ indicates the Kronecker delta and ensures that the flow from the source and the flow to the sink is considered for the first and last node of each airport. The balance constraints for source node and sink node are listed separately for emphasis and clarity (see Eq. 4d and 4e, respectively). The additional source and sink usage constraints (see Eq. 4f) ensure that for each type and airport aircraft can either get drained to the sink or flow in from the source.

In contrast to the first algorithm, the result holds no information about the point in time when spare aircraft are available or when additional aircraft are needed at an imbalanced airport. Instead, the departure times of additional repositioning flights are determined afterwards as described below. During this procedure, the subschedule for each aircraft type k of the reassigned schedule obtained from the optimization above is treated individually. For each airport $i \in A$ the imbalance $\xi_i^k(t_{start}, t_{end})$ is computed. Similar to the second step of A1, a complete bipartite graph is created with as many nodes per airport (on the left side or on the right site depending on the imbalance sign) as its absolute imbalance value. Again, for each combination the shortest path is computed. The key difference to the matching procedure in the previous algorithm lies in the direct use of the resulting path distances as the costs used in a minimal complete matching. Once this matching is obtained, the best points in time at which to add the itineraries corresponding to these matches to the schedule need to be determined. This time corresponds to the horizontal position of the departure in the TSN, see Fig. 4b. For each involved aircraft type, this is done by a greedy algorithm, similar to the one by Ceder and Stern (1981), that successively adds the itineraries: One step of this greedy algorithm consists of adding one itinerary. Let P be the union of the times of arrival or departure at the origin of the itinerary and the times of arrival or departure at the destination of the itinerary minus the flight time. (If we end up with a number $\leq t_{start}$, due to the periodicity of the schedule, we go to t_{end} and subtract what is left.)

The position $t \in P$ is chosen that optimizes the net effect $\Delta g_k(t)$ on the sub-type fleet size g_k that is required to execute the resulting schedule determined as in Gu et al. (1994). First, $n_{i,k}^{schedule}$ is calculated, which corresponds to the number of aircraft required at airport i at the beginning of the schedule. It equals the maximum negative disbalance of all time intervals that start at the beginning of the schedule. The difference of this value after and before adding the itinerary at position t is denoted by $\Delta n_i(t)$ and describes how the number of aircraft that are needed at this airport changes. The only airports that need to be considered here are the origin and the destination of the itinerary. Additionally, the change in the number of aircraft flying at the beginning of the schedule $\Delta\kappa(t)$ needs to be considered. The best position for adding the itinerary is the one that minimizes the total change:

$$\Delta g_k(t) = \Delta n_{origin,k}(t) + \Delta n_{destination,k}(t) + \Delta \kappa_k(t), \tag{5}$$

$$\min_{t \in P} \Delta g_k(t). \tag{6}$$

3.3. Data, computing methods and settings

The repair algorithms were applied to schedules of a large German airline from Sabre GLBL Inc. (2025). Both a one-day and a one-week schedule were chosen for each month in 2023. For the assignment costs $c_{l,k}$, the assignment dependent marginal returns (excluding factors assumed to be constant such as cost of crew and aircraft acquisition)

were used. These were calculated as the difference between assignment dependent cost of operation (ADCO) and revenue. The revenue was calculated based on cumulative revenue curves assuming optimal revenue management. These curves were derived from the base fares provided by Sabre GLBL Inc. (2025), which consist of the ticket prices without fuel surcharges and taxes. The revenue was processed like in Grimme et al. (2021) except that the revenue for the investigated month was distributed among the executed flights. If necessary, the curve was extrapolated to larger aircraft using the average fee of the 5 % least paying passengers. The ADCO was calculated based on the CeRAS Model using the 'TU Berlin DOC Method' (see Thorbeck and Scholz (2013); Risse et al. (2016)) as implemented by Kühlen et al. (2023) only including the costs for fuel and CO_2 as well as fees for landing, ground handling and navigation. The fuel consumption was computed with the DLR Emission Tool by Buchtal and Clococeanu (2024) using BADA3 data by EUROCONTROL (see Nuic et al. (2010)) and additional DLR aircraft design data which was created following the modelling approach presented by Woehler et al. (2020) and Leipold et al. (2025). The average fuel price of 2023 was calculated from data provided by IndexMundi (2025) to be 0.66 €/kg. The fee values recommended by Thorbeck and Scholz (2013) were inflation adjusted (see Federal Statistical Office of Germany (2024)). For the total weight per passenger 100 kg were assumed. The airlines fleet was taken from from Cirium (2025). Payload-range diagrams and turnaround times were taken from the aircraft manuals (see Airbus (2025); Boeing (2025)). Flight durations were were averaged from the historic schedules, if available, or via linear interpolation based on the great-circle distance, if not. For A1, the aircraft were grouped by IATA-code, number of seats and engine type. The base penalty ρ for teleportations was chosen as 1 million €. The optimization problems were solved using Gurobi Optimization, LLC (2025) changing the integrality gap to 0.01 and using presolve aggression 2.

4. Results and discussion of model differences

Table 1. Results for 1-day-schedules.

		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
No. of explored nodes	A1	855	1	1471	1	979	1	1	1	84	1	1072	975
	A2	1	1	1	1	1	1	1	1	1	1	1	1
Optimization time [sec]	A1	43	50	65	45	100	47	42	41	74	62	129	102
	A2	0.9	0.9	0.4	0.5	1.1	0.7	1.2	1.4	0.9	1.2	0.5	0.8
No. of teleport legs	A1	3172	3520	3366	3983	4317	3531	3585	3697	4235	4303	3657	3407
No. of new itineraries	A1	26	27	32	16	19	22	17	15	15	23	13	17
	A2	26	27	32	16	19	22	17	15	15	23	13	17
No. of eligible itineraries	A1	20	22	28	10	14	17	14	11	12	18	7	9
No. of switched IATA codes	A2	32	30	23	41	39	31	35	43	39	37	38	35
No. of new legs	A1	41	53	51	31	33	35	30	27	28	37	21	26
	A2	40	47	49	30	32	37	30	27	27	37	17	25
No. of needed aircraft	A1	231	235	256	247	261	284	278	275	285	277	259	258
	A2	217	219	226	236	252	276	274	272	282	265	248	259

Both algorithm succeeded to create balanced schedules. The key metrics for comparison are listed in Tab. 1. Seven general conclusions can be drawn from comparing the algorithms: First, the airline's fleet of 308 aircraft (with varying range and airport compatibility, see Cirium (2025)) sufficed to execute any of the repaired schedules. There was one exception, which was the one-week schedule repaired by A1. This can be explained by the second conclusion, which is as follows: Schedules repaired by A2 require less (or equal) aircraft for execution after fleet assignment. This is directly evident from Tab. 1. A third, more subtle insight is that the algorithms "priorities" are nicely reflected by the observed assignment-change dynamics. A1 aims at coming very close to the final fleet assignment already during repair. As a result, the majority of assignment changes are performed during this step. In contrast, A2 penalizes change in fleet type during repair. Therefore, fleet types are mostly preserved during this step and change later during fleet assignment optimization. Next, the experiments show that A1 is not superior to A2 from an economic perspective. The assignment dependent marginal returns resulting from both algorithms differ by less than 1 %. This is surprising, as A1 is inherently more economically driven. Possible reasons are a fleet assignment stable cost function or the assignment changes in the final fleet assignment.

Moreover, except for the one-day schedule in June, A2 added less or equal legs compared to A1. Interestingly, the number of new itineraries is always the same for both algorithms and did not change even with penalty ρ increased by up to a factor of 10 in one instance. This indicates that the penalty ρ of A1 was high enough to prevent unnecessary teleportations. Hence, the observed difference likely stems from the matching methods: A2's distance minimization favors fewer legs, whereas A1 lacks this driving force. Another key finding is that the computational effort is higher for A1 than for A2. As evident from Tab. 1 and confirmed by the one-week schedule results, A2 already found the optimal solution investigating the models root relaxation. In contrast, for most A1-repairs an extensive branch-and-bound tree needed to be explored. This could originate from the high degree of integrality of A2's root relaxation solution. In all cases, the optimization procedure in A1 took several orders of magnitude longer than in A2. As a final advantage, A2 is easier to implement into the DAT due to its independence from the economic evaluation of assignments.

5. Conclusion and outlook

Due to shorter calculation times while generating (depending on the exact motivation of the repair) equally good or even better results than A1, A2 was chosen as the most suitable candidate for the DAT. However, in scenarios with larger cost variations within the feasible set, A1 might outperform A2. Additionally, future studies should analyze the sensitivity regarding the algorithms parameters. Furthermore, it should be investigated if and how the beneficial effect that A2 has on the required fleet size can be made robust to delays. Also, the performance of a modified A2 in flight cancellation management should be investigated. Those considerations remain subject to further studies.

References

Airbus, 2025. Aircraft characteristics. URL: aircraft.airbus.com. Accessed: 16 Sep 2024.
Boeing, 2025. Airplane characteristics for airport planning. URL: www.boeing.com. Accessed: 16 Sep 2024.
Buchtal, K.A., Clococeanu, M., 2024. Emissionstool - software. German Aerospace Center e.V. (DLR), Institute of Air Transport.
Caetano, D.J., Gualda, N.D.F., 2010. A flight schedule and fleet assignment model, in: Proc. of 12th World Conf. on Transp. Res., Lisboa, Portugal.
Ceder, A., Stern, H.I., 1981. Deficit function bus scheduling with deadheading trip insertions for fleet size reduction. Transp. Sci. 15, 338–363.
Cirium, 2025. Fleets Analyzer. URL: www.cirium.com/products/views/fleets-analyzer/. Accessed: 21 Jul 2024.
Federal Statistical Office of Germany, 2024. Verbraucherpreisindex und Inflationsrate. URL: www.destatis.de. Accessed: 25 Sep 2024.
Gopalan, R., Talluri, K.T., 1998. Mathematical models in airline schedule planning: A survey. Ann. Oper. Res. 76, 155–185.
Grimme, W., Maertens, S., Bingemer, S., 2021. The role of very large passenger aircraft in global air transport – a review and outlook to the year 2050. Transp. Res. Procedia 59, 76–84.
Gu, Z., Johnson, E.L., Nemhauser, G.L., Yinhua, W., 1994. Some properties of the fleet assignment problem. Oper. Res. Lett. 15, 59–71.
Gurobi Optimization, LLC, 2025. Gurobi optimizer reference manual. URL: www.gurobi.com. Accessed: 23 Apr 2025.
Hane, C.A., Barnhart, C., Johnson, E.L., Marsten, R.E., Nemhauser, G.L., Sigismondi, G., 1995. The fleet assignment problem: Solving a large-scale integer program. Mathematical Programming 70, 211–232.
IndexMundi, 2025. Jet fuel - monthly price (euro per gallon). URL: www.indexmundi.com. Accessed: 28 Mar 2025.
Kühlen, M., Lütjens, K., Linke, F., Gollnick, V., 2023. An explanatory approach to modeling the fleet assignment in the global air transportation system. CEAS aeronautical journal 14, 255–269.
Leipold, A., Baier, F., Bauder, U., Blinstrub, J., Buchtal, K.A., Campillo Borras, E., Clococeanu, M., Eckel, G., Ennen, D., Flüthmann, N., Gelhausen, M., Hoff, T., Kühlen, M., Kumar, S., Link, A., Ratei, P., Ruoff, S., Schmid, R., Weber, L., 2025. Development pathways for aviation up to 2070 – study report.
Lohatepanont, M., Barnhart, C., 2004. Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment. Transp. Sci. 38, 19–32.
Nuic, A., Poles, D., Mouillet, V., 2010. BADA: An advanced aircraft performance model for present and future ATM systems. Int. J. Adapt. Control Signal Process. 24, 850–866.
Risse, K., Schäfer, K., Schültke, F., Stumpf, E., 2016. Central Reference Aircraft data System (CeRAS) for research community. CEAS Aeronaut. J. 7, 121–133.
Sabre GLBL Inc., 2025. Sabre Market Intelligence - database. URL: www.sabre.com. Accessed: 23 Apr 2025.
Sherali, H.D., Bish, E.K., Zhu, X., 2006. Airline fleet assignment concepts, models, and algorithms. Eur. J. Oper. Res. 172, 1–30.
Stern, H.I., Gertsbakh, I.B., 2019. Using deficit functions for aircraft fleet routing. Oper. Res. Perspect. 6, 100104.
Thorbeck, J., Scholz, D., 2013. TU Berlin DOC method: A simplified DOC model: JAVA DOC applet. 3rd Symposium on Collaboration in Aircraft Design. URL: www.fzt.haw-hamburg.de. Accessed: 23 Apr 2025.
Woehler, S., Atanasov, G., Silberhorn, D., Fröhler, B., Zill, T., 2020. Preliminary aircraft design within a multidisciplinary and multifidelity design environment. URL: elib.dlr.de/185515/. Accessed: 23 Apr 2025.