Computers and Operations Research 117 (2020) 104892

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

# Dynamic aircraft recovery problem - An operational decision support framework

J. Vink[a,b], B.F. Santos[a,*], W.J.C. Verhagen[a,c], I. Medeiros[d], R. Filho[d]

[a] Air Transport and Operations, Faculty of Aerospace Engineering, Delft University of Technology, the Netherlands
[b] ORTEC B.V., The Netherlands USA
[c] Aerospace Engineering and Aviation, School of Engineering, RMIT University, Australia
[d] Embraer S.A., Brazil

ARTICLE INFO

*Article history:*
Received 31 March 2019
Revised 12 January 2020
Accepted 13 January 2020
Available online 15 January 2020

*Keywords:*
Airline disruption management
Dynamic aircraft recovery
Irregular airline operations

ABSTRACT

This paper presents a new approach for solving the recovery of the airline schedule when disruptions have occurred. The goal is to develop an operational tool that provides the airline with a solution in less than one minute. The proposed recovery model uses a heuristic that iteratively solves selections of the airline's fleet in order to quickly converge to a good solution. An initial solution is always presented in seconds, after which potential reductions of disruption cost are investigated. The schedule is modeled as a set of parallel time-space networks, using an integer linear programming. The model is solved dynamically; a recovery solution is found whenever a disruption occurs and subsequent disruptions are solved based on the previously found solution. Aircraft maintenance schedules and passenger itineraries are modeled, while crew concerns are indirectly taken into consideration to avoid major disruptions caused by the recovery solution. The approach presented in this paper can be applied on heterogeneous fleets and to both point-to-point and (multi) hub-and-spoke airlines. The performance of the selection heuristic is discussed using a case study on the network of an airline operating in the United States. This case study shows that the selection heuristic can find a globally optimal solution in 90% of the disruption instances tested, within 22 s on average. This corresponds to 4% of the time needed to compute the optimal solution using the entire fleet.



## 1. Introduction

The airline industry encounters disrupted flight schedules on a daily basis. Delays, inclement weather, temporary closures of airports or airspaces, unscheduled maintenance, all of these can be the cause of severe disruptions to the airline's flight schedule. Such disruptions affect the scheduled routing of aircraft, but can also upset crew schedules and passenger itineraries. The entire process of recovering the schedule after disruptions occur is defined as disruption management. EUROCONTROL (2019) has established that in Q1 of 2019, 19.2% of all flights in Europe suffered from delays. Furthermore, 42% of the total generated delay minutes have a *reactionary* cause, meaning the delay is the result of an aircraft's late arrival from a previous flight. This emphasizes the need for disruption management systems.

For a disruption management system to be suitable for real-time implementation at an airline, it should be reliable and provide solutions quickly. This is because in actual operations multiple disruptions occur throughout the day and decisions need to be made in a matter of minutes, after new disruption information becomes available. We define this iterative process of re-solving the airline's schedule every time new disruption information is made available as *dynamic recovery*. In this process, the resolution of new disruptions starts from the solution for the previous disruption event. Previous decisions(s) can be revoked if it improves the solution and under the constraint that there is still time to change operations.

Despite the interest of many researchers in the airline recovery problem, few studies consider this inherent dynamic aspect of the problem. To the best of the authors' knowledge, the works from Bratu and Barnhart (2006), Vos et al. (2015), and Marla et al. (2017) are the only exceptions. Nevertheless, none of these models are suitable for operational use by airlines. To enable such use, three aspects must be addressed: first, the model shall provide good solutions at the fleet level in one or two minutes, as demanded by most airline operations controllers; second, the model should be able to handle a large set of possible disruptions; and third, the model shall always provide a solution, regardless of

* Corresponding author.
*E-mail address:* b.f.santos@tudelft.nl (B.F. Santos).

https://doi.org/10.1016/j.cor.2020.104892
0305-0548/

the eventual infeasibility of the problem from the operational point of view (e.g., an aircraft may get stuck during the day at an airport closed due to adverse weather conditions, never reaching the home base of the airline as required to resume operations the next day).

In this paper we address these aspects. We present a new modeling approach that has been designed to be implemented in practice, solving the aircraft routing and flight schedule parts of the airline recovery problem in practice. This problem of solving the aircraft routing and flight schedule together is denoted as the Aircraft Recovery Problem (ARP). Passengers and aircraft maintenance schedule recovery are considered in the model by modeling passengers' re-accommodation possibilities and considering individual aircraft maintenance requirements. The latter aspect requires aircraft to be modeled individually. This greatly increases the model's computational complexity. Therefore the problem is divided and solved sequentially, such that the problem size remains manageable and solutions are provided in a matter of seconds. Finally, crew scheduling concerns are indirectly modeled, limiting the impact of aircraft and schedule changes on the crew schedules.

This study provides new insights into three aspects:

1. We address the practical challenge of developing a decision support model suitable for operational use – the model provides good solutions within one minute and slack variables are used to guarantee that the problem is always feasible.
2. We propose an expedited way to incorporate connecting passengers in the Aircraft Recovery Problem (ARP) without increasing the complexity of the resulting model.
3. We approach the Aircraft Recovery Problem (ARP) according to the dynamic aspect of the problem, demonstrating how much we under-estimate the disruption costs when considering fully information in a static version of the problem.

This paper is structured as follows. First existing literature is reviewed in Section 2. The proposed modelling approach is discussed in Section 3, followed by the explanation of the proposed rule-based selection heuristic in Section 4. A case study is presented in Section 5 that is used to demonstrate and validate the methodology as it could be applied in practice, followed by a study of the robustness of the results, presented in Section 6. Conclusions are presented in Section 7.

## 2. Literature overview

The development of modeling techniques for airline disruption management has frequently been addressed in research. Previous efforts were focused on specific aspects of airline disruption management, or addressed the combination of multiple aspects in one model. This section first discusses research with a focus on the recovery of aircraft routing, after which the literature that includes other aspects of the recovery problem is presented. A chronological overview of the evolution of modeling developments to address the Aircraft Recovery Problem (ARP) is shown in Fig. 1.

### 2.1. Aircraft recovery

Teodorović and Guberinić (1984) were among the first to address the Aircraft Recovery Problem (ARP), solving a problem for eight flights with the objective to minimize passenger delay. The problem was modeled as a connection network, as defined by Clausen et al. (2010). This work was extended in Teodorović and Stojković (1990) and Teodorović and Stojković (1995). Also in the 90's, researchers started to model the Aircraft Recovery Problem (ARP) using time-space networks. A time-space network is a transformation of the Aircraft Recovery Problem (ARP) in a two-dimensional plane where one axis time intervals are represented and the other airport stations. Jarrah et al. (1993) developed two successive shortest path models, one considering the delay of flights and another considering flight cancellation as a recovery technique. Yan and Yang (1996) incorporated both cancellations and delays of flights. Delays are incorporated using sliding arcs; that is, copies of the flight arcs at later departure times. The model is formulated as an integer programming problem. The model was tested on a single fleet of 17 aircraft and 39 flights. Thengvall et al. (2000) was the first to propose parallel time-space networks for a multi-fleet situation, using a mixed integer programming (MIP) approach. A separate time-space network is created for each sub-fleet. Protection arcs are included as an incentive for the model to adhere to the initial flight schedule. Vos et al. (2015) proposed parallel time-space networks for each individual aircraft so that aircraft-specific constraints could be incorporated. This model is solved *dynamically*, meaning that disruptions are solved as they become known, and subsequent disruptions are solved based on the previously found recovery solution. The model was tested on a fleet of 43 aircraft serving 53 destinations, for which solutions were found in about 10–15 min.

Alternative to the time-space network representation, time-band networks were introduced by Bard et al. (2001) to model the Aircraft Recovery Problem (ARP). These are similar to time-space networks, in which only the arcs corresponding to flights are created and nodes represent all activities at an airport within a certain time interval (the time-band). This node consolidation is used to reduce the size of the time-space network. For examples on time-band networks, see Eggenberg et al. (2010) and Hu et al. (2015). Both works used parallel network structures to model individual aircraft. The former work used separate networks for each aircraft to include maintenance constraints while solving the Aircraft Recovery Problem (ARP). The authors used a column generation approach to solve disruptions for a fleet of 10 aircraft within 30 seconds. Hu et al. (2015) modeled disruptions in a fleet of 178 aircraft split over multiple sub-fleets, for which solutions were found within approximately three minutes for five-minute time-bands. The authors considered re-accommodation options for passengers on disrupted flights, but passenger connections at a hub airport and aircraft maintenance requirements were not modeled.

### 2.2. Integrated recovery

With the increase in computer performance over the last decade, more aspects of airline operations have been integrated in recovery models. In this section we discuss some of these integrated models.

Bratu and Barnhart (2006) presented a delay management model to define which outbound flight to delay and which to cancel. The authors considered passenger connections and assumed the presence of reserve crews to guarantee crew schedule feasibility. In this work, the fleet has been modeled as a single commodity, without differentiating aircraft in the fleet. Two versions of the passenger delay model were investigated. An initial version assumed an approximation of the delay costs according to an algorithm previously developed by the authors. The second version explicitly modeled passenger disruptions and the recovery options. Despite the approximation assumed in the first model, the authors mentioned a computation time of 4.5 min for a problem with 302 aircraft of four different aircraft types. The computational times of the second version of the model were up to 25 times longer. More recently, Santos et al. (2017) also addressed the airline delay management model to cope with disruptions in a hub airport. In this study the authors modeled passenger connectivity at a hub airport, integrating airport capacity management when solving the flight

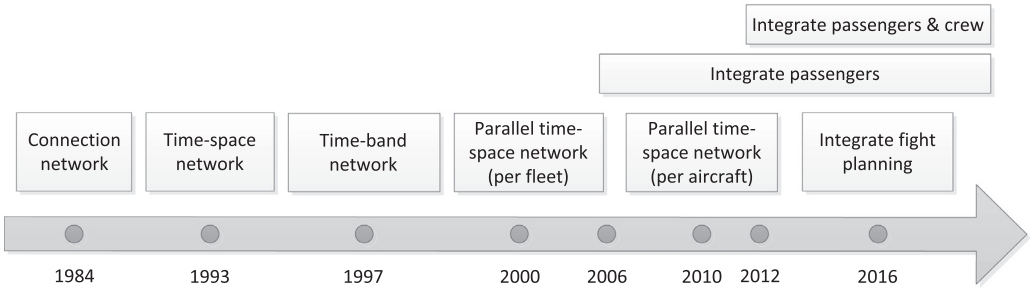


**Fig. 1.** Timeline of developments in modeling the aircraft recovery problem.

rescheduling problem. Although the authors referred to the operational use of the tool by an airline, they reported computational times of five to six minutes to solve for time windows of 180 min, using a rolling horizon approach.

Bisaillon et al. (2011) developed a large neighborhood search heuristic to solve the Aircraft Recovery Problem (ARP) while including passenger itineraries. Solutions were found within a 10 min time limit, and the model won the ROADEF 2009 Challenge. Inconsistency and penalty costs are applied for decisions that are possible but not preferable, for example when a passenger's itinerary cannot be restored. This model was improved in Sinclair et al. (2014) and Sinclair et al. (2016). The focus of these papers was on improving the solution quality, assuming fixed running times of five or 10 min. A heuristic method was also developed by Jozefowiez et al. (2013) as part of the ROADEF 2009 Challenge. The authors proposed a three-phase heuristic to solve the schedule, aircraft and passenger rescheduling problems. By comparing their results with the results previously published, the authors showed that their heuristic method was capable of obtaining the best-found solution on more than half of the large instances, while not taking more than four minutes.

Models that integrate aircraft, passenger and crew recovery have also been developed. Considering these three aspects at the same time greatly increases the complexity of the problem. Crew schedules for example, are highly regulated and schedule changes can still have effects multiple days after the disruption. Petersen et al. (2012) were the first to present results for a fully integrated model. A Benders' decomposition is used to solve the master problem (flight schedule) and three subproblems (aircraft recovery, crew recovery and passenger re-accommodation). The integrated approach finds equal or lower cost solutions compared to the traditional approach where the subproblems were solved sequentially. Also, the integrated approach is capable of finding solutions for which the traditional approach does not find a feasible solution. The improved performance in terms of solution cost comes at the price of increased computational time. For larger disruptions, the time required to find a solution often exceeds 20 min. This makes such a model unsuitable for real-time operational use. Maher (2015) presents a column and row generation approach for solving an integrated recovery model. Big-M constraints are used to ensure passengers are re-accommodated if possible, but these are not formulated as hard constraints to prevent model infeasibility. Only single leg passenger itineraries are considered. Computation times of 20 min are reported.

Previously mentioned work did not consider the possibility to use cruise speed control to prevent delays. Arikan et al. (2016) incorporate this option using a conic quadratic mixed integer programming (MIP), because of the nonlinear relation between cruise speed and fuel consumption. The authors argue that passenger delay can be greatly mitigated by using this approach. Marla et al. (2017) consider discrete cruise speed changes to study the relation between fuel burn and delay costs, and show that overall costs can be reduced by allowing cruise speed changes. Using discrete alternative flight plans (associated with different cruise speeds), the problem is formulated as an mixed integer programming (MIP) problem. The departure time of outbound flights can also be re-timed. The objective is to minimize the sum of fuel, delay and cancellation costs. The research focuses on the delay of long haul flights inbound to the hub, as these flights carry passengers that connect to downstream flights. The model integrates passenger delay costs. These are approximated, as it was established that explicitly modeling each connecting passenger results in a formulation of the problem that cannot be solved in a time frame that is acceptable for airlines. When a passenger misses a connecting flight a fixed penalty is applied, assuming that this passenger is re-accommodated in the next bank. Computation time is limited to two minutes, and the mixed integer programming (MIP) gap is evaluated to determine how far solutions are from the optimum solution. The model is tested on an airline operating a hub-and-spoke network with 250 flights. For large inbound delays, exceeding 120 min, authors mention an average mixed integer programming (MIP) gap of about 13–15% when alternative flights plans are included. Without alternative flight plans included, the model finds the optimal solutions in two minutes. Marla et al. (2017) consider the dynamic nature of airline disruption management by considering fixed points in time, just before each arrival and departure wave of the long-haul flights. The airline's network is modeled using parallel time-space networks; a separate network is created for each aircraft. On these networks, maintenance tasks are included if applicable. These tasks are compulsory and cannot be canceled.

The goal of the research in this paper is to develop an operational tool that provides solutions quickly. In consultation with airlines it was established that solutions have to be available within 1 minute to be suitable for operational use. Moreover, an operational tool should always provide the airline with a solution, even if not all airline preferences can be satisfied. The model presented in this paper applies a similar principle as used in Marla et al. (2017), although we do not regard cruise speed changes. We expand the scope of disruptions considered; in addition to arrival delay of flights with connecting passengers as assumed by Marla et al. (2017), we consider flight cancellations and temporary unavailability of aircraft and airports. These are realistic disruptions that cause more severe disturbances than a simple delay of an inbound flight. Severe disruptions may cause conflicts such that certain airline preferences cannot be satisfied, e.g. the overnight-location of a certain aircraft. As our goal is to develop a tool for operational use, we include strategies to cope with such situations. This paper also furthers the techniques that can be used to incorporate connecting passengers in the Aircraft Recovery Problem (ARP) without increasing the complexity of the problem to be solved.

## 3. Modelling approach

In this section we explain the approach followed to address the dynamic Aircraft Recovery Problem (ARP), in which disruptions are solved as they occur throughout the day. For every disruption the recovered schedule resulting from the previous disruptions is used as a starting point. The schedule is to be recovered within a certain time window. At the end of the time window, all aircraft are expected to be at the right location such that scheduled operations can be resumed from this point on. If disruptions are so severe that a requested aircraft location at the end of the time window can not be complied with, we provide the next-best solution: one where the minimal amount of location requests are violated. The violated request is highlighted and the alert is provided to the controller. This human-in-the-loop characteristic is one of the aspects that makes this model an operational tool. Another aspect is the time needed to find a solution. For an airline, it is important to obtain a feasible solution in a very short time. The model provides the first recovery solution within seconds such that the airline can implement decisions quickly, if necessary.

The methodology can be applied to both hub-and-spoke and point-to-point network structures, and to airlines operating a heterogeneous fleet. Four types of disruptions are considered in this work. Namely,

- delay of flights,
- cancellation of flights,
- closure of airports for a period of time, and
- technical problems that make the aircraft unavailable for a period of time.

The next subsection describes the parallel time-space network approach adopted in this work, followed by the formulation of the optimization model. The main ingredients of the model, including the modeling of connecting passengers and the consideration of maintenance requirements, are explained in the following subsections.

### 3.1. Parallel time-space networks

In this work we make use of parallel time-space networks to model each individual aircraft (Fig. 2), as previously proposed by Vos et al. (2015). A separate network is created for each aircraft, which are connected by flight coverage constraints. Time is discretized in homogeneous time-steps. Scheduled flight time is extended to accommodate for the minimum required turnaround time (TAT), and the departure and arrival times are rounded to the nearest time-step. The advantage of individual aircraft networks is that we can use specific *tail numbers*, solving the scheduled recovery problem together with the aircraft routing recovery problem. In addition, aircraft specific constraints, such as maintenance requirements and destination at the end of the day, can be modeled.

Besides delay options, we assume that flights can be canceled or operated by other aircraft. This means that each aircraft network contains the flight arcs (both on time and delayed) of all other aircraft in the recovery model. The airline needs to define which aircraft types are potential substitutes for each other, providing a swapping matrix similar to the fleet substitution and capacity matrix presented in Hu et al. (2015). The difference in operating costs and the passengers costs resulting from considering aircraft with different seating capacity are reflected in the costs associated with the flight arcs involving aircraft swaps.

### 3.2. Model formulation

In this section we present our recovery model. Notation is introduced first, after which the model formulation is presented. The following sections explain some of the key features of the model in more detail.

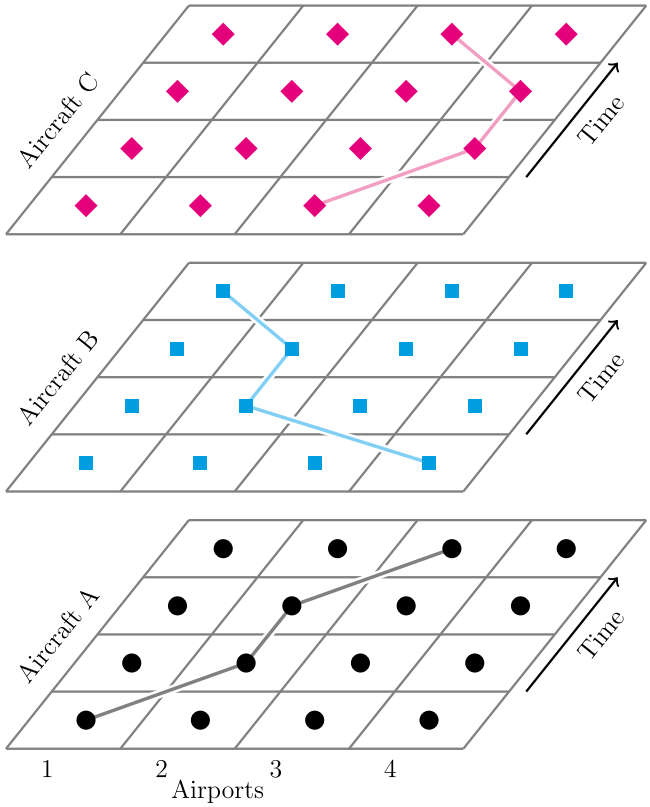


**Fig. 2.** Principle of parallel time-space networks. A separate network is created for each aircraft. Note that only operated flight and ground arcs are illustrated, non selected arcs are not shown for the sake of clarity of the figure.

*Notation*

| **Sets** | |
|---|---|
| $P$ | Aircraft in the (selected) fleet |
| $F$ | Flight arcs allocated to aircraft present in $P$ in the disrupted schedule |
| $F_p$ | Fight arcs currently allocated to aircraft $p$ |
| $D$ | Delayed flight arcs |
| $G$ | Ground arcs associated with $F$ |
| $N$ | Intermediate nodes associated with $F$ |
| $R$ | Maintenance activities associated with $F$ |
| $R_{flex}, R_{fix}$ | Flexible and fixed maintenance activities |
| $E$ | Aircraft types included in $P$ |
| $A$ | Airports associated with $F$ |
| $O(n, p)$ | Arcs originating at node $n$ for aircraft $p$ |
| $T(n, p)$ | Arcs terminating at node $n$ for aircraft $p$ |
| $Z(a, e)$ | Arcs terminating at airport $a$ for aircraft type $e$ |
| $X(p, f)$ | Forbidden flight $f$ and aircraft $p$ combinations |
| $U(r, p)$ | Ground arcs corresponding to fixed maintenance activities $r$ for aircraft $p$ |
| $V(r, p, m)$ | Ground arcs corresponding to flexible maintenance activities $r$ for aircraft $p$ at start time $m$. |

| **Decision Variables** | |
|---|---|
| $\delta_{F_{p,f}}$ | Binary variable equal to 1 if aircraft $p$ is allocated to flight arc $f$, zero otherwise |
| $\delta_{D_{p,f,d}}$ | Binary variable equal to 1 if aircraft $p$ is allocated to delay arc $f$ with delay $d$, zero otherwise |
| $\delta_{C_f}$ | Binary variable equal to 1 if flight $f$ is canceled, zero otherwise |
| $\delta_{G_{p,g}}$ | Binary variable equal to 1 if aircraft $p$ uses ground arc $g$, zero otherwise |
| $y_{r,m}$ | Binary variable equal to 1 if maintenance activities $r$ starts at time $m$, zero otherwise |
| $s_{a,e}$ | Integer slack variable for aircraft type $e$ balance at airport $a$ at end of time window |
| $s_r$ | Binary slack variable equal to one if maintenance activities $r$ are canceled |
| $s_p$ | Binary slack variable equal to one if the routing of aircraft $p$ is changed when compared with the original routing |

| **Parameters** | |
|---|---|
| $OC_{p,f}$ | Operating cost of aircraft $p$ on flight $f$ |
| $DC_{f,d}$ | Delay cost of flight $f$ for delay $d$ |
| $CC_f$ | Cancellation cost of flight $f$ |
| $GC_n$ | Cost of ground arc $n$ |
| $M_{a,e}$ | Cost of missing an aircraft of type $e$ at airport $a$ at the end of the time window |
| $M_r$ | Cost of canceling maintenance activities $r$ |
| $M_p$ | Cost of changing the routing of aircraft $p$ |
| $B_{n,p}$ | Flow balance at node $n$ for aircraft $p$. 0 at intermediate nodes, 1 at start node |
| | where aircraft $p$ is supplied. |
| $K_{a,e}$ | Number of aircraft of type $e$ demanded at airport $a$ at the end of the time window. |

*Formulation*

$$\text{Minimize: } \sum_{p\in P}\sum_{f\in F} OC_{p,f}\cdot\delta_{F_{p,f}} + \sum_{p\in P}\sum_{f\in F}\sum_{d\in D}(OC_{p,f}+DC_{f,d})\cdot\delta_{D_{p,f,d}} + \sum_{f\in F} CC_f\cdot\delta_{C_f} + \sum_{p\in P}\sum_{g\in G} GC_n\cdot\delta_{G_{p,g}} + \sum_{a\in A}\sum_{e\in E} M_{a,e}\cdot s_{a,e} + \sum_{p\in P} M_p\cdot s_p + \sum_{r\in R} M_r\cdot s_r \quad (1)$$

Subject to:

$$\sum_{p\in P}\left(\delta_{F_{p,f}} + \sum_{d\in D}\delta_{D_{p,f,d}}\right) + \delta_{C_f} = 1 \qquad ,\forall f\in F \quad (2)$$

$$\sum_{g\in G\cap O(n,p)}\delta_{G_{p,g}} - \sum_{g\in G\cap T(n,p)}\delta_{G_{p,g}} + \sum_{f\in F\cap O(n,p)}\delta_{F_{p,f}} - \sum_{f\in F\cap T(n,p)}\delta_{F_{p,f}} + \sum_{f\in F,d\in D\cap O(n,p)}\delta_{D_{p,f,d}} - \sum_{f\in F,d\in D\cap T(n,p)}\delta_{D_{p,f,d}} = B_{n,p}, \quad \forall n\in N,\forall p\in P \quad (3)$$

$$\sum_{g\in G\cap Z(a,e)}\delta_{G_{p,g}} + \sum_{f\in F\cap Z(a,e)}\delta_{F_{p,f}} + \sum_{f\in F,d\in D\cap Z(a,e)}\delta_{D_{p,f,d}} = K_{a,e} - s_{a,e} \qquad ,\forall a\in A,\forall e\in E \quad (4)$$

$$\sum_{f\in F_p}\left(\delta_{F_{p,f}} + \sum_{d\in D}\delta_{D_{p,f,d}}\right) \geq |F_p|\cdot(1-s_p) \qquad ,\forall p\in P \quad (5)$$

$$\sum_{g\in U(r,p)}\delta_{G_{p,g}} \geq |U(r,p)|\cdot(1-s_r) \qquad ,\forall r,p\in R_{\text{fix}} \quad (6)$$

$$\sum_{g\in V(r,p,m)}\delta_{G_{p,g}} \geq |V(r,p,m)|\cdot y_{r,m} \qquad ,\forall r,p,m\in R_{\text{flex}} \quad (7)$$

$$\sum_{m\in V(r,p,m)} y_{r,m} = 1 - s_r \qquad ,\forall r,p\in R_{\text{flex}} \quad (8)$$

$$\sum_{f\in F_p\cap X(p,f)}\delta_{F_{p,f}} + \sum_{d\in D\cap X(p,f)}\delta_{D_{p,f,d}} = 0 \quad \forall p\in P \quad (9)$$

Bounds:

$$\delta_{F_{p,f}}\in\{0,1\} \quad \forall p\in P,\forall f\in F \quad (10)$$

$$\delta_{D_{p,f,y}}\in\{0,1\} \quad \forall p\in P,\forall f\in F,\forall d\in D \quad (11)$$

$$\delta_{G_{p,g}}\in\{0,1\} \quad \forall p\in P,\forall g\in G \quad (12)$$

$$\delta_{C_f}\in\{0,1\} \quad \forall f\in F \quad (13)$$

$$y_{r,m}\in\{0,1\} \quad \forall r,m\in R_{flex} \quad (14)$$

$$s_r\in\{0,1\} \quad \forall r\in R \quad (15)$$

$$s_p\in\{0,1\} \quad \forall p\in P \quad (16)$$

$$s_{a,e}\in\mathbb{Z}_{\geq 0} \quad \forall a\in A,\forall e\in E \quad (17)$$

We propose an integer linear programming (ILP) model that minimizes cost, while ensuring that all scheduled flights are flown if possible. The objective function (1) includes both operating and disruption costs. The first term defines the operating cost of flights departing at their scheduled time, while the second term covers the cost of delayed flights. If a flight is canceled, a cancellation cost is assigned in the third term. The fourth term covers the cost of operating ground arcs (if applicable). We define the sum of the first four terms as the *operational* disruption cost. The last three terms define the *artificial* disruption costs and penalties, discussed in Section 3.5.

The problem is subjected to several constraints. The flight coverage constraint (2) ensures each flight is either performed by one of the (on time or delayed) aircraft in the fleet or canceled. Flow balance is ensured by two constraints. For the intermediate and start nodes (3), balance is maintained for all nodes in each aircraft time-space network: if an aircraft is supplied to a certain node, it should also depart from that node. For nodes at the end of the time window (4), the demanded number of aircraft of a certain type should be ensured. These constraints must be included to guarantee that aircraft are in the right position to resume scheduled operations.

Using Constraint (5), a penalty is assigned in the objective function whenever the scheduled routing of an aircraft is changed. The set $F_p$ contains all the flights scheduled for an aircraft. If all these flights are flown by the aircraft, no penalty is assigned. But in the case that one of the flights is performed by another aircraft (or canceled), the scheduled string of flights is broken and the variable $s_p$ is used to satisfy the constraint.

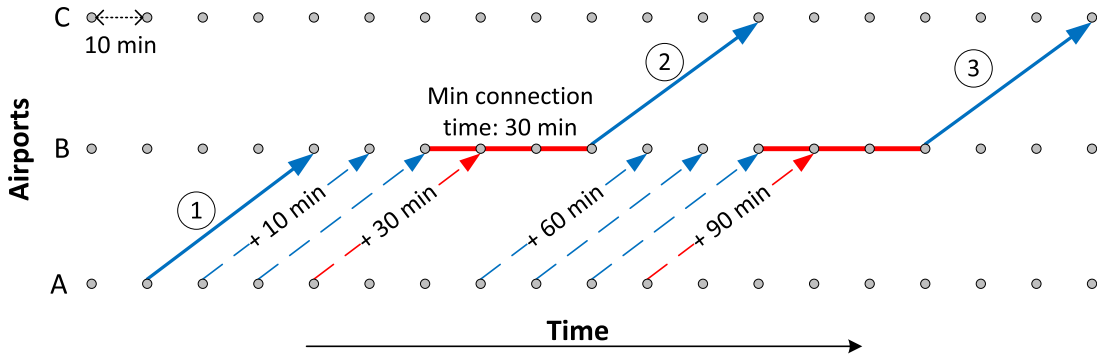


**Fig. 3.** Connecting flight example. Passengers on Flight 1 need to connect to Airport C at Airport B. Flight 2 is the scheduled connection, Flight 3 is a recovery connection.

Constraints (6), (7) and (8) guarantee aircraft maintenance requirements. They ensure that ground arcs that represent scheduled maintenance are covered by the respective aircraft. Maintenance activities are divided into fixed and flexible maintenance requirements, as will be further explained in Section 3.4. Fixed maintenance activities (6) have to be respected, unless the binary slack variable $s_r$ is activated, with an associated high cost in the objective function. For flexible maintenance tasks, the set of constraints (8) ensure that one of the possible maintenance slots is chosen, while Constraint (7) defines the ground arcs that correspond to each of the maintenance slots.

Constraint (9) ensures that in a situation with multiple aircraft types, flights are only performed by aircraft that satisfy conditions such as range and number of required seats. Constraints (10)–(17) define the decision variables as being binary or integer.

### 3.3. Connecting passengers

Especially for network carriers, connecting passengers amount to a significant portion of the airline's customers. Passengers who miss their connecting flight can be severely disrupted, resulting in large compensation costs for the airline. This research aims to include the effects of passengers who miss their connection in the aircraft recovery problem. However, as discussed in Bratu and Barnhart (2006) and Marla et al. (2017), this increases the complexity of the Aircraft Recovery Problem (ARP) to a significant extent. In light of this consideration, we propose an expedited approach to consider connecting passengers' recovery itinerary options without compromising the computation time of our model.

We precompute the connecting passenger delay costs for multiple delay values. Delay is measured at the final destination of the passengers. To do this, we estimate the delay impact of having to divert passengers if their connection is lost due to a delay of the inbound flight at the hub airport. The determination of the cost of missed connections is illustrated in Fig. 3. In this example, Flight 1, from Airport A to Airport B, carries passengers that want to transfer at Airport B to their final destination Airport C. If Flight 1 is delayed up to 20 min, the connection at Airport B can be guaranteed and no delay is experienced by passengers. If the flight is delayed by 30 min or more, the delay for the connecting passengers would be 60 min (at their end-destination). Should Flight 1 be delayed by 90 min or more,then also Flight 3 is missed. This would result in a much higher delay cost equivalent to canceling the passenger's itinerary, if we assume that Flight 3 is the last flight of the day flying to Airport C.

Following this approach, we precompute two matrices of size *flights* * *(time steps + 1)* (the additional time step dimension corresponds to the cancellation of the flight). The first matrix stores the delay costs at the final destination for connecting passengers per flight and delay time step. The second matrix stores a tuple with the number of the downstream flight to which the connecting passengers will be allocated to.

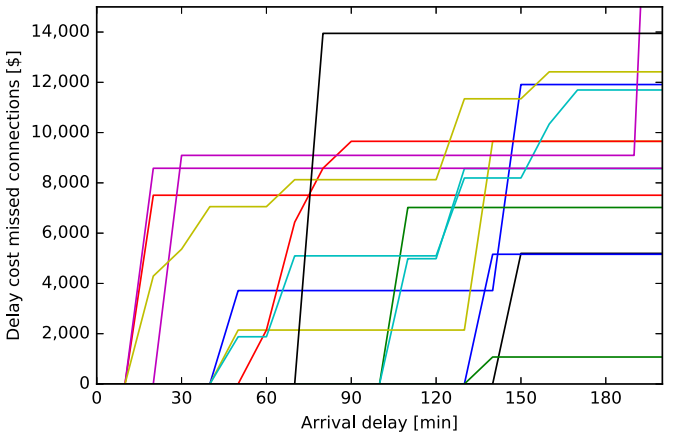


**Fig. 4.** Delay cost profiles for different flights when downstream flights are missed as a result of arrival delay. Each line represents a different arrival flight.

The delay costs matrix provides a *delay profile* per flight, as represented in Fig. 4. The delay costs increases represent a group of passengers that will miss their connection. When a connection is missed, we consider the capacity available on the alternative flights. If an alternative flight can not accommodate some or all of the disrupted passengers, the delay costs of these passengers are determined based on the arrival time of a later flight. If no capacity is available in later flights, or if there are no later flights, the cancellation and accommodation costs are considered. The delay costs matrix for connecting passengers is processed together with the calculation of the delay costs for direct passengers in each flight to compute the matrix of coefficients in our optimization model ($DC_{f,d}$ and $CC_f$).

This approach allow us to estimate the delay experienced by passengers, the costs of these delays and the itineraries followed by passengers, without adding additional constraints or decision variables in our optimization model. However, the following simplifications have been incorporated:

- For a given disruption event, it is assumed that the outbound flights are not delayed in the recovery solution. This assumption was made deliberately to maintain tractability of the problem. However, before solving a future disruption, the seat availability per flight, the departure times of outbound flights and the previous estimations of passenger delays can be updated.
- Reallocation of passengers will follow the sequence of airports in their original itinerary. That is, we do not consider the possibility to reroute passengers via another hub of the airline.

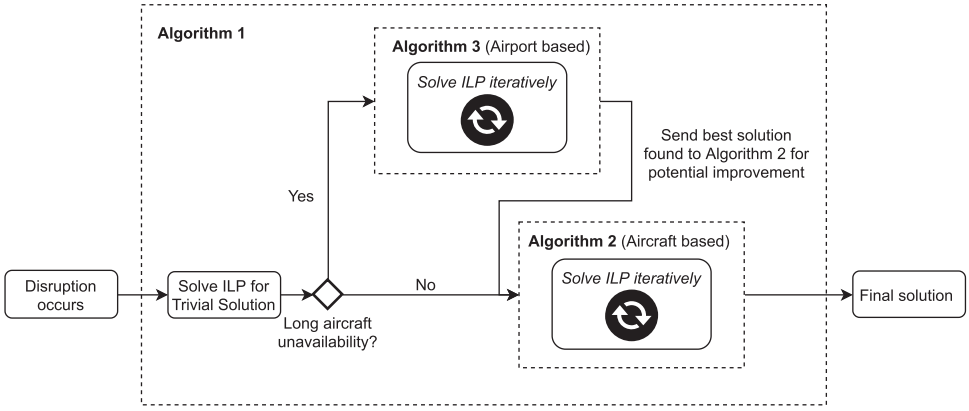


**Fig. 5.** Flowchart of selection algorithm

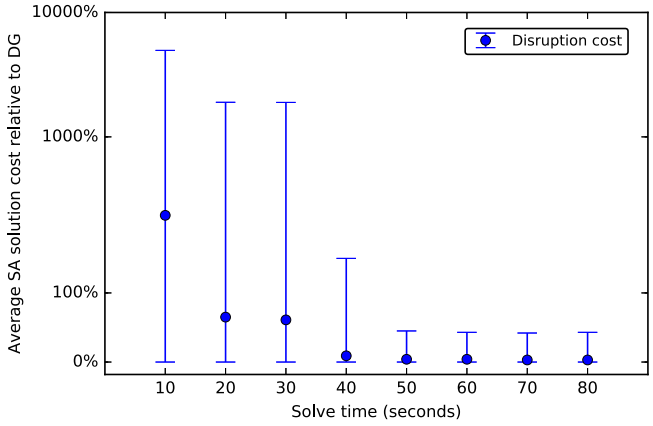


**Fig. 6.** Reduction of best solution (lowest disruption cost) found by SA as time progresses.

- We only consider itineraries with two flight legs. However, the approach could be extended to also assess the effect on itineraries with more than two flights.

The advantages of this approach are three-fold. First, it provides better estimations of the exact passenger delay costs than the fixed costs per connecting passengers approach proposed by previous authors (e.g., Bratu and Barnhart 2006; Marla et al. 2017). Second, it provides additional value by considering passengers' itineraries and providing reallocation actions without compromising computational times. Third, reallocation actions are defined based on booking information and the delay costs are calculated irrespective of bank times at a hub, as proposed by Marla et al. (2017).

### 3.4. Maintenance requirements

Scheduled maintenance activities are included in the recovery model (constraints (6), (7) and (8)), such that in the recovery solution all aircraft are routed to their maintenance station, if scheduled as such. A distinction is made between *fixed* and *flexible* maintenance activities. A fixed maintenance activity is a maintenance check that can not easily be rescheduled, either because it involves many resources (e.g. hangar space) or because it is close to the aircraft maintenance interval limits. A flexible maintenance activity are, most commonly, inspections scheduled to be performed on an in-service aircraft during its turnaround time (TAT), (also known as *line* or *platform* maintenance). These are activities that, in some cases, can be rescheduled to other times and eventually to a different maintenance station. The flexible maintenance activities can be rescheduled only if available maintenance slots exist and if they do not exceed the maximum number of aircraft that can be serviced simultaneously at an airport (given resource limitations). Determining which maintenance activities are fixed or flexible is up to the airline, in coordination with the maintenance provider.

Maintenance activities ($R$) are modeled using ground arcs ($U(r, p)$ and $V(r, p, m)$, respectively for fixed and flexible maintenance activities). Whenever a fixed maintenance task is scheduled for an aircraft, it is ensured in the model that the aircraft uses the ground arcs at the corresponding airport. For a flexible maintenance task, sets of sequential ground arcs are defined, representing the elapsed time of the maintenance task. One of these sets needs to be used in the obtained solution in order to guarantee the performance of the maintenance. Because we use the existing ground arcs, additional decision variables to model the maintenance tasks are not required.

### 3.5. Initial schedule and infeasible solutions

This section addresses two additional issues regarding the practicability of the proposed model. The first concerns the goal to respect the initial schedule as much as possible, which is assumed to be the optimal schedule for the day before disruptions take place. The second issue is the impossibility to respect some of the airline's preferences in some disruptions scenarios, making the optimization model infeasible. Both issues are handled in the optimization model using penalty costs, either associated with slack variables or incorporated in the operating costs in the objective function.

#### 3.5.1. Initial schedule

The goal of airline disruption management is to minimize disruption costs. Ideally costs are minimized while swapping the least number of aircraft possible. The reason for this being that each aircraft swap represents an operational burden that is usually associated with hidden costs (e.g. the cost of re-positioning an aircraft or adapting crew schedules).

We propose two measures that can be used to respect the airline's initial schedule. Firstly, a penalty is applied whenever a flight is operated by a different aircraft than was scheduled. This concept is similar to the penalty costs used by Bisaillon et al. (2011). This penalty is included in the operating cost of the flights ($OC_{p,f}$). Using this penalty, the model has a preference for utilizing the scheduled aircraft or, if not possible, an aircraft of the same type. This penalty should represent the costs that are incurred by an airline in the case of a tail swap, such that a swap is only performed when the mitigated delay costs offset the swap costs.

The second measure concerns the aircraft routing. The penalty on a flight-level does not protect the consecutive operation of flights by the scheduled aircraft. As such, this could have a great impact on crew schedules and passengers that are on a through-flight. Therefore a second penalty ($M_p$) is applied whenever the initially scheduled routing of an aircraft is changed. This concept is similar to the protection arcs presented by Thengvall et al. (2003). The slack variables used to indicate route changes ($s_p$) are integer, meaning that the solution will be penalized proportionally to the number of changes in the schedule.

Two observations need to be made regarding these penalties. The first is that the penalties are optional and adjustable to the airline specific operations and policies. There is always the option to not include these penalties. The second observation is the fact that these penalties can help us to return to the initial schedule during a day of operations, after deviating from the initial schedule when solving earlier disruptions. Even after a tail swap is performed, the initially scheduled tail number is taken into account by the model. If the situation presents itself that later during the day the model can return a flight to the initially scheduled aircraft, the cost incentive still exists. This is a consequence of the followed dynamic approach.

#### 3.5.2. Infeasible solutions

Certain disruptions may have such an effect on the flight schedule that some of the airline's preferences for the recovery solution can not be satisfied. For example, when an aircraft is grounded for the rest of the day for technical reasons, it may not be possible to satisfy the aircraft demand at the nodes at the end of the time window (end nodes). The same applies to constraints related to maintenance tasks. If an aircraft cannot be routed to its maintenance location due to disruptions, this can result in an infeasible integer linear programming (ILP) problem. Therefore, slack variables are included in each constraint for maintenance tasks and end nodes. These slack variables are associated with *artificial* disruption costs in the objective function. These costs are many times greater than the largest cost of normal variables, and therefore the recovery model will only select these slack variables if there is no feasible solution. The slack variables are used to identify constraints that could not be satisfied and the operations controller is made aware of the specific airline's preferences not satisfied by the final solution.

The slack variables ($s_r$), associated with the maintenance task constraints, are binary; the task is either completed or not. The slack variables associated with the end node constraints ($s_{a,e}$) are integer, corresponding to the number of aircraft of a certain type that are missing at the end node. As a result, every missing aircraft is penalized.

## 4. Aircraft selection algorithm

The recovery model previously presented in this paper finds the set of recovery decisions resulting in the lowest disruption cost. However, constructing the parallel time-space networks for each aircraft in the fleet and solving the resulting integer linear programming (ILP) problem can require too much time to be suitable for operational use.

By working with several airlines, we observed that only a subset of aircraft are used in practice to solve disruptions. Even in the case of large disruptions. This happens because, on one hand, for minor disruptions it is not necessary to involve many aircraft and, on the other hand, because airline controllers avoid to involve too many aircraft when solving disruptions. Therefore, we propose a *Selection Algorithm*, in which different selections of aircraft are iteratively considered. This is a rule-based procedure used to ensure that solutions are always provided to the airline within seconds. Intermediate solutions are not necessarily optimal. Yet this enables the airline to, when needed, implement quick decisions before the algorithm is completely finished. The rules were defined following several experiments with different rules and the observation of some of the decisions made at the Operations Control Center (OCC) from multiple airlines when solving disruptions. The costs of the solutions found using the proposed Selection Algorithm, and the solution that is found when all aircraft are considered by the integer linear programming (ILP) solver, are compared in Section 5.

The work from Vos et al. (2015) already suggested the use of aircraft selections to solve the Aircraft Recovery Problem (ARP). The authors suggest that aircraft selections can be made based on the assumption that the ground-time of aircraft is the best indicator for a good candidate. Aircraft with the longest ground-time are considered to be the best candidates to help solve disruptions and are solved by the integer linear programming (ILP) solver first. In this paper we demonstrate that better candidate aircraft can be found based on the location of the aircraft rather than ground-time. Also, long-time impact disruptions (e.g. a flight cancellation or the long unavailability of an individual aircraft) require different sorting techniques. Another difference to the solution technique proposed by Vos et al. (2015) is the fact that our method always provides an initial solution in seconds, based on which better results will be provided as time progresses, making it more suitable for operational use.

The Selection Algorithm proposed in this paper consists of three individual algorithms, for which a high-level overview is shown in Fig. 5. As can be seen, when a disruption occurs the trivial solution is solved first. Next, Algorithm 2 is always executed, and Algorithm 3 will be executed only if the disruption to be solved is a long aircraft unavailability. The notation used for the three algorithms is introduced next, after which the procedure to determine the aircraft selections is explained in more detail using the three algorithms.

*Notation Selection Algorithm*

| | |
|---|---|
| **Aircraft sets** | |
| $S_{disrupt}$ | Disrupted aircraft at $t_0$ |
| $S_{unavailable}$ | Aircraft with unavailability disruption at $t_0$ |
| $S_{used}$ | Aircraft used in best found integer linear programming (ILP) solution |
| $P$ | Aircraft to be considered in the integer linear programming (ILP) |
| $C$ | Candidate aircraft |
| $C_{delay}$ | Candidate aircraft for delayed flights |
| $C_{cancel}$ | Candidate aircraft for cancelled flights |
| $C_{other}$ | Candidate aircraft not in $C_{delay}$ or $C_{cancel}$ |
| $C_{select}$ | Candidate aircraft selected for integer linear programming (ILP) solve |
| $L_1$ | Aircraft in Level 1 sort of Algorithm 3 |
| $L_2$ | Aircraft in Level 2 sort of Algorithm 3 |
| **Aircraft information** | |
| $p$ | Aircraft index |
| $D_p$ | Time duration of unavailability disruption |
| **Flight information** | |
| $F_{delay}$ | Delayed flights in integer linear programming (ILP) solution |
| $F_{cancel}$ | Cancelled flights in integer linear programming (ILP) solution |
| $f$ | Flight index |
| $o_f$ | Origin airport of flight $f$ |
| $depart_f$ | Scheduled departure time of flight $f$ |

(*continued on next page*)

| | |
|---|---|
| $delay_f$ | Delay of flight $f$ |
| **Airport information** | |
| $a$ | Airport index |
| $A_{sche}$ | Airports in flight schedule |
| $A_{cand}$ | Candidate airports for Algorithm 3 |
| **Selection Algorithm information** | |
| $t_0$ | Time at which disruption(s) is(are) known |
| $Sol_{trivial}$ | Trivial solution |
| $Sol_{best}$ | Best solution found so far |
| $Sol_{final}$ | Final solution |
| $Sol_{airport}$ | Best found airport solution |
| $T_{unavailable}$ | Threshold of unavailability time |
| $T_{cancel}$ | Additional time to determine candidate aircraft for cancelled flights |
| $K$ | Multiplication factor used to determine the size of $C_{select}$ |

Algorithm 1 explains the high-level structure of the Selection Algorithm. First, the *Main Selection Algorithm* is run with only the disrupted aircraft. This is generally a small selection of aircraft, and thus results in a simple problem which can be solved in a few seconds. The solution provides the airline with an overview of the potential effects of the disruptions, and a disruption cost indication. We call this solution the *trivial solution*, where we only allow disruption effects to propagate within the flights of the disrupted aircraft.

**Algorithm 1** Main Selection Algorithm

**procedure** TRIVIAL SOLUTION
  Solve integer linear programming (ILP) (1) - (17) with $P = S_{disrupt}$ ▷ First find Trivial Solution
  $F_{delay} \leftarrow$ delayed flights in trivial solution
  $F_{cancel} \leftarrow$ canceled flights in trivial solution
  update $Sol_{trivial}$
**for** $p \in S_{unavailable}$ **do**
  **if** $D_p > T_{unavailable}$) **then** ▷ Strategy in case of long unavailability
    **procedure** SOLVE AIRPORTS ITERATIONS
      **goto:**Algorithm 3**return** $S_{used}, F_{delay}, F_{cancel}$
**procedure** SOLVE AIRCRAFT ITERATIONS
  **goto:**Algorithm 2**return** $Sol_{final}$ ▷ Default strategy
**end**

From this trivial solution we attempt to reduce disruption costs by adding aircraft that are likely to help reduce cost. Based on airline defined criteria, candidate aircraft are drawn from the fleet of aircraft indicated to be a potential substitute of the disrupted aircraft (e.g. same fleet or same (sub-) type). The Selection Algorithm has two different solution strategies for the selection of aircraft, depending on the set of disruptions. The first strategy focus on the airports in which the disruption(s) take place to select the aircraft that may contribute to solve the disruption(s). This strategy is is activated in all disruption events and is presented in Algorithm 2. The second strategy considers other airports in the scheduled routes of aircraft present at airport in which the disruption(s) take place at the time of the disruption(s). This strategy is only activated in the presence of a long aircraft unavailability (larger than a given threshold defined by $T_{unavailable}$) and is discussed in Algorithm 2.

In Algorithm 2 the aircraft are sorted based on their location during the day. Any aircraft that is at the same airport as a disrupted aircraft between the time of occurrence of the disruption

**Algorithm 2** Solve Aircraft Iterations

**INPUT:** $S_{used}, S_{disrupt}, F_{delay}, F_{cancel}$ from Algorithm 1
**for** $f \in F_{delay}$ **do**
  $C_{delay} \leftarrow C_{delay}+$ aircraft at $o_f$ between $t_0$ and $(depart_f + delay_f)$ ▷ No duplicate aircraft
**for** $f \in F_{cancel}$ **do**
  $C_{cancel} \leftarrow C_{cancel}+$ aircraft at $o_f$ between $t_0$ and $(depart_f + T_{cancel})$ ▷ No duplicate aircraft
$C \leftarrow$ sorted list $C_{cancel} + C_{delay}$ ▷ Sorted based on ground time
$C_{other} \leftarrow$ any aircraft $\notin (C_{cancel} \cup C_{delay})$ ▷ Other aircraft
$C = C + C_{other}$
**while** $|C| > 0$ **do** ▷ Iterative integer linear programming (ILP) solver
  $C_{select} \leftarrow$ first $K \cdot |S_{disrupt}|$ aircraft of $C$
  Solve integer linear programming (ILP) (1) - (17) with $P = S_{disrupt} + S_{used} + C_{select}$
  update $S_{used}$
  update $Sol_{best}$
  Remove $C_{select}$ from $C$
**return** $Sol_{best}$

and the delayed departure time is considered a *candidate aircraft*. We assume that these candidates are in a good position to potentially swap with the disrupted aircraft. Within these candidates, we give higher priority to those that have a larger ground time during that airport visit. For canceled flights the procedure is similar; candidates are those that are at the same airport between the time of occurrence of the disruption and scheduled departure time of the canceled flights, plus an additional time window. This window is introduced to consider aircraft that arrive after the arrival time of the canceled flight. Given the high costs of canceling a flight, these aircraft may still be valuable for the recovery. The candidate aircraft from canceled and delayed flights are sorted at the top of the list according to their total time on the ground. The remaining aircraft are then added to the bottom of the list of candidates. In this way all aircraft will be taken into consideration once, but those considered earlier in the process are more likely to contribute to a good recovery solution.

After sorting the list of aircraft, the integer linear programming (ILP) solver is iteratively called, with a different set of aircraft of the sorted candidate list ($C$), together with the disrupted aircraft ($S_{disrupt}$). The number of aircraft of the sorted list is defined as a multiple of the size of $S_{disrupt}$. When a better solution is found, the best solution is updated and the aircraft involved in the solution are stored as $S_{disrupt}$. These aircraft are considered in subsequent iterations, guaranteeing that at the end of each iteration the solution is as good or better than the previously found solution. The algorithm stops when all aircraft in the candidate list have been considered to solve the set of disruptions.

Algorithm 3 was developed to address demanding cases in which aircraft are unavailable for a long duration of time. These are difficult cases to be solved because these disruptions may potentially involve the cancellation of many flights. This can be hard to solve by only using the tail swaps between aircraft that are both in the same selection, as defined in Algorithm 2. Therefore, to understand better how to solve these large disruptions, we tried to solve these cases with the entire fleet of aircraft. This gave us the optimal solutions for these cases. It was observed that, in some cases, the optimal solution would involve aircraft that did not visit the disrupted airport. However, these aircraft did visit airports that were also visited by aircraft that originated from the disrupted airport. At these undisrupted airports, they were used in tail swaps with aircraft that took some of the flights from the disrupted air-

craft. From this we concluded that for making a good selection of candidate aircraft, it is useful to determine which airports in the network are visited by aircraft that originate from the disrupted airport.

**Algorithm 3** Solve Airport Iterations

**INPUT:** $S_{disrupt}, F_{delay}, F_{cancel}, D_p$, $p$ from Algorithm 1
**procedure** LEVEL 1 AIRPORT SORT
  $f_1 \leftarrow$ first flight $f$ of unavailable aircraft $p$
  $L_1 \leftarrow$ aircraft departing from $o_{f_1}$ between $depart_{f_1}$ and $(depart_{f_1} + D_p)$
  **for** $a \in A_{sche}$ **do**
    Count number of flights in $A_1$ to $a$
  $A_{cand} \leftarrow$ any destination of flights in $L_1$ with $> 1$ flight ▷ These have swap opportunities
**procedure** LEVEL 2 AIRPORT SORT
  **for** a $\in A_{cand}$ **do** ▷ Iterative integer linear programming (ILP) solver
    $L_2 \leftarrow$ any aircraft at $a$ between $depart_{f_1}$ and $(depart_{f_1} + D_p)$
    Solve integer linear programming (ILP) (1) - (17) with $P = S_{disrupt} + L_2$
    update $Sol_{airport}$
  $S_{used} \leftarrow$ aircraft used in $Sol_{airport}$
  $F_{delay} \leftarrow$ delayed flights in $Sol_{airport}$
  $F_{cancel} \leftarrow$ canceled flights in $Sol_{airport}$
**return** $S_{used}, F_{delay}, F_{cancel}$

Therefore, the need of an additional selection strategy for the Selection Algorithm was considered. Algorithm 3 is initiated in Algorithm 1 in case of an aircraft unavailability that exceeds the user defined threshold ($T_{unavailable}$). This threshold is used to prevent additional integer linear programming (ILP) iterations in Algorithm 3 in case of shorter aircraft unavailability. These additional integer linear programming (ILP) iterations would require more computational time, while the default strategy (Algorithm 2) provides good selections of aircraft to solve the disruption.

Algorithm 3 consists of selecting aircraft that visit the same airport (other than the airport where the unavailable aircraft is grounded) later in the day. Let *Airport A* be the airport where the aircraft is grounded. We define $A_{canc}$, a list of airports that are visited later during the day by aircraft that are also at *Airport A* during the unavailability. From this list of airports we disregard the ones that are visited only by one of these aircraft (if only one aircraft visits the airport, there is little opportunity to perform swaps such that we mitigate delays at *Airport A*). We consecutively run the integer linear programming (ILP) solver for the different airports in $A_{canc}$ until all candidate airports have been run. For the candidate airport that provides the lowest cost recovery solution, we retrieve the aircraft that are used in the solution ($S_{used}$). After these runs, we will continue the Selection Algorithm with the default solution strategy defined in Algorithm 2.

It is important to note that the selections in the Selection Algorithm are created based on the location of aircraft, irrespective of the airline network structure. Therefore, the algorithm can be applied to both point-to-point and (multi) hub-and-spoke networks. In addition, the algorithm is solved several times, considering multiple selections of aircraft (see Algorithm 2). This provides intermediate solutions to the operations controller even before the problem is completely solved. Hence, if time is a limiting factor, the operations controller may accept the last solution found without waiting for the model to finish all selections.

## 5. Case study

To demonstrate the sequential approach for solving the aircraft recovery problem, the model described was tested based on information from an airline operating in the United States. The airline operates a fleet of over 100 aircraft, serving 70 destinations on 600 daily flights. All flights are medium-haul or short-haul flights. The fleet is split over two aircraft types of the same family with different seating capacity (70 and 80 seats). Flight schedule, fleet composition, maintenance schedule and booked passengers per flight were obtained from the airline. However, information regarding connecting passengers was not available and had to be estimated based on open-source data from the US Department of Transportation (https://www.transtats.bts.gov).

The recovery model was implemented in Python using CPLEX 12.7 to solve the integer linear programming (ILP) problem. Computational tests were conducted on a computer using a 64 bit Intel i5-3470 3.20 GHz processor and 8 GB RAM.

### 5.1. Parameter assumptions

The scenarios tested in the case study were performed using the following parameters.

- The time window of recovery is equal to 16 h, the duration of one day of operations for the flight schedule studied.
- Time is discretized in 10 min time steps.
- Flights can be delayed up to six hours. After six hours, flights are considered to be canceled.
- Passenger delay cost averages $1.28 per passenger per minute, but increases with the severity of the delay, using a piece-wise linear function, as proposed by Cook et al. (2012). This covers all cost experienced by the airline. Cost of reputation damage were obtained from the work from Cook et al. (2012), while cost of compensations to passengers were derived from airline data.
- Passengers that do not reach their final destination at the end of the day are treated as if they are delayed by ten hours (after this limit the delay costs are kept constant). In addition, a compensation cost of $250 is added to cover accommodation costs.
- Schedule penalties costs: $100 per tail swap, $1,000 per change of aircraft routing ($M_p$). These values are derived from the minimum delay required to offset the penalty costs. Given the capacity of the aircraft in the case study, and an average load factor of 75%, the $100 penalty corresponds to delays exceeding 10 min. To change the aircraft route is only justified if it can save a flight delay longer than 40 minutes.
- Slack variables cost: $1,000,000 per missing aircraft at end of time window ($M_{a,e}$) or when a maintenance task needs to be canceled ($M_r$). These costs should be sufficiently high such that the model will only violate these constraints if no other possibility exists. This value corresponds to the cost of about eight flight cancellations. A low slack variable cost can result in violation of constraints in favor of flight cancellations.
- Cost of ground arcs are neglected for the case study.
- The factor $K$ for the Selection Algorithm (Algorithm 2) is set to two.
- The time threshold $T_{unavailable}$ used to invoke Algorithm 3 is set to four hours. $T_{canc}$ is set to three hours.

In this case study we did not consider the possibility to upgrade (or downgrade) the seat class for a passenger subjected to an itinerary disruption. This means that when checking the possibility accommodation of passengers in different flights than the ones included in the original passenger itinerary we only consider seats of the same class.

### 5.2. Disruption scenarios

In the case study, the recovery model was tested using several illustrative disruption scenarios (Table 1). These scenarios are artificial as actual disruption information was not available for the network studied. However, they were proposed by the airline participating in this study as being representative for a disrupted day of operations. Each scenario covers one entire day of operations, during which several unexpected disruption events occur. The scenarios were constructed such that different types of disruptions are combined and that real-life situations, e.g. hub closures or reduced departure capacity as a result of extreme weather, are simulated.

A set of disruptions becoming known at the same time is called a *disruption event*. Each scenario is composed of two to five disruption events happening at different moments of the day. For the sake of simplicity, the disruption events of all scenarios are defined at five specific times of the day. However, these events could vary in number and occur at completely different moments.

### 5.3. Solutions being compared

For the disruption scenarios discussed, four different solutions are presented and compared.

- **Trivial (TR) solution** - the solution which is found taking only the disrupted aircraft in consideration. This solution resembles the immediate solution that a controller would find during operations and it is the first solution obtained by the Selection Algorithm. It is found in about two to three seconds, because the resulting problem involves few aircraft and is therefore small.
- **Selection Algorithm (SA) solution** - the best solution found by *Selection Algorithm* presented in this work, after considering all the selections of aircraft candidate to solve the disruption events.
- **Dynamic Global (DG) solution** - the solution when the entire fleet of candidate aircraft is used. Determining this solution takes too long to be suitable for real-time use. However, this solution corresponds to the *global* optimum solution to the disruption problem, as defined in this work. This is used as the reference solution in the analysis of the results.
- **Static Global (SG) solution** - the solution found if we assume that all disruptions for the day are know at the beginning of the day; all disruptions are solved in a single run. This solution also uses the entire fleet of candidate aircraft and it is used to assess the relevance of adopting a realistic dynamic approach when solving airline disruption problems.

To fairly compare the solutions obtained for each disruption event during the day, we opted to compute the TR solutions for one disruption event following the SA solution from the previous event (or the initial schedule, in the case of the first disruption event of the day). The DG solutions are always based on the previous DG solution. That is, the DG solution for the first disruption event is taken as the starting schedule for the computation of the DG solution for the second disruption event.

### 5.4. Overall results comparison

The results for all ten scenarios are summarized in Table 2. The values presented are the averages among the values obtained for the different disruption events in each scenario. In total, we modeled 48 disruption events over the 10 scenarios. The first three columns (after the scenarios column) compare the average computational time needed to find the DG solution and the best SA solution. The fourth column compares these same two solutions in terms of their objective function value. The same is done in the last column but for the DG and SG solutions. The compared solutions

**Table 1**
Scenarios as used in the case study. Different disruptions occur at the indicated time during the day.

| Scenario | Description | 06:00 | 09:00 | 12:00 | 15:00 | 18:00 | 20:00 |
|---|---|---|---|---|---|---|---|
| **1** | *Several disruptions in multiple aircraft* | 2 delays (50 & 50 min) | 4 delays (40, 60, 30 & 50 min) | 1 unavailable (2 h) | 2 delays (100 & 110 min) | 2 delays (100 & 120 min) | 1 delay (60 min) |
| **2** | *Disruptions at multiple out-stations* | 1 unavailable (3 h) | 1 delay (180 min) | 1 delay (120 min) | 1 delay (150 min) | - | - |
| **3** | *Small & medium disruptions 1* | 1 delay (90 min) | 2 delays (110 & 90 min) | 1 unavailable (3 h), 1 canx | 1 unavailable (2 h) | 1 delay (80 min) | 2 delays (20 & 80 min) |
| **4** | *Reduced departure capacity at hub* | - | 3 delays (90,80,90 min) | 2 canceled, 4 delays (100, 80, 70, 80 min) | 2 canceled, 2 delays (80, 90 min) | 2 delays (30, 30 min) | - |
| **5** | *Unexpected maintenance events* | - | 1 unavailable (8 h) | 1 unavailable (6 h) | 1 unavailable (8 h) | - | - |
| **6** | *Disrupted out-station (single)* | - | 2 delays (120 & 90 min) | 1 delay (50 min) | 1 delay (60 min) | 2 delays (60 & 70 min) | 2 delays (40 & 40 min) |
| **7** | *Multiple long delays at different airports* | 1 delay (90 min) | 1 delay (120 min) | 1 delay (80 min) | 1 delay (120 min) | 1 delay (90 min) | 1 delay (60 min) |
| **8** | *Small & medium disruptions 2* | 1 unavailable (2 h) | 1 delay (50 min) | 1 delay (70 min) | 2 delays (30 & 60 min) | 1 unavailable (1.5hrs) | 1 delay (40 min) |
| **9** | *Hub closed for 2 hrs + delays after* | - | hub closed (2 h) | 2 delays (60 & 40 min) | - | - | - |
| **10** | *Small & medium disruptions 3* | 1 unavailable (2 h) | 2 delays (40 & 50 min) | 1 delay (30 min) | 1 unavailable (3hrs) | 1 delay (60 min) | 1 delay (50 min) |

**Table 2**
Runtime and operational cost comparison for all scenarios. Runtimes are the average of all disruption events in a scenario. Model settings: time window = 16 hours, maximum delay = 6 hours, $K = 2$.

| Scenario | Average runtime SA [mm:ss] | Average runtime DG [mm:ss] | Selection runtime ratio [SA/DG] | Selection cost ratio [SA/DG] | Static cost ratio [SG/DG] |
|---|---|---|---|---|---|
| 1 | 00:21.7 | 06:59.0 | 0.05 | 1.00 | 0.77 |
| 2 | 00:22.5 | 07:41.9 | 0.05 | 1.03 | 0.99 |
| 3 | 00:06.6 | 08:22.3 | 0.01 | 1.00 | 1.00 |
| 4 | 00:41.4 | 18:24.2 | 0.04 | 1.00 | 0.92 |
| 5 | 00:44.0 | 12:26.8 | 0.06 | 1.39 | 0.88 |
| 6 | 00:05.8 | 07:47.2 | 0.01 | 1.00 | 1.00 |
| 7 | 00:17.7 | 07:08.4 | 0.04 | 1.00 | 0.97 |
| 8 | 00:20.2 | 05:41.8 | 0.06 | 1.00 | 1.00 |
| 9 | 00:36.6 | 17:58.4 | 0.03 | 1.19 | 0.99 |
| 10 | 00:04.8 | 05:49.5 | 0.01 | 1.00 | 0.37 |
| Average | 00:22.1 | 09:49.9 | 0.04 | 1.06 | 0.81 |

did not involve the use of slack variables and, for an easier comparison of the results, the schedule penalty costs are not included in the objective function values compared from the different solutions.

When analyzing the results, the first observation is that the best SA solution was always found in less than 50 s, and on average it only took 22.1 s to be found. On the other hand, we needed almost 10 min, on average, to compute the DG solution. In some cases, it took more than 20 min. This means that the SA is 20 times faster than when solving the integer linear programming (ILP) problem for the entire fleet. This, as might be expected, results from the reduction of the problem size. When using the entire fleet we deal with a problem of the order of 3300 thousand columns and 750 thousand rows (depending on the specific problem being solved). With the SA, per selection analyzed, we reduced the number of columns by a factor of about 100 and rows by a factor of 40. Still, for seven of the ten scenarios we found the global optimum solution with the SA – this happened for 90% of the 43 disruption events solved. Only for four disruption events this did not happen. The worst case was in Scenario 5 when three aircraft are unavailable for long periods. In this case, the SA obtained solutions that were 39% worse than the optimum solution. However, on average, we obtained solutions in a matter of seconds that were only 6% more expensive than the global optimum.

Comparing the SG solution with the DG solution, we can say that the average disruption costs are under-estimated by 19% when we assume that all disruptions are known at the start of the day. This is the *'price'* of not adopting a dynamic approach when solving disruption problems. The difference between both solutions can be of more than 60%, as it happened for Scenario 10.

The goal of the SA is to provide solutions in less than one or two minutes. The algorithm provides an initial solution, the TR solution, in less than three seconds. Every few seconds the algorithm solves the problem using a new selection of aircraft and provides a solution that is always at least as good as the solutions previously found. Fig. 6 depicts the trade-off between the time that the airline has available and the quality of the solution that the recovery model presents. It shows how disruption cost for all scenarios reduces as time progress. The dot figures represent the average value over all the scenarios while the error bars indicating the range of all observations. After 50 s the SA has converged and the solutions no longer improves. That means that within one minute the airline control has the best solution for the disruption event.

### 5.5. Scenarios analysis

The results of scenarios 1 and 5 are discussed in detail in this subsection. Scenario 1 is a standard disruption scenario, where different disruptions occur in several aircraft throughout the airline's network, at different times during the day. This was one of the scenarios in which the SA always found the optimum solution. On the other hand, Scenario 5 was the scenario with the worst performance from the SA. As it was previously mentioned, this is a scenario involving the long unavailability of several aircraft.

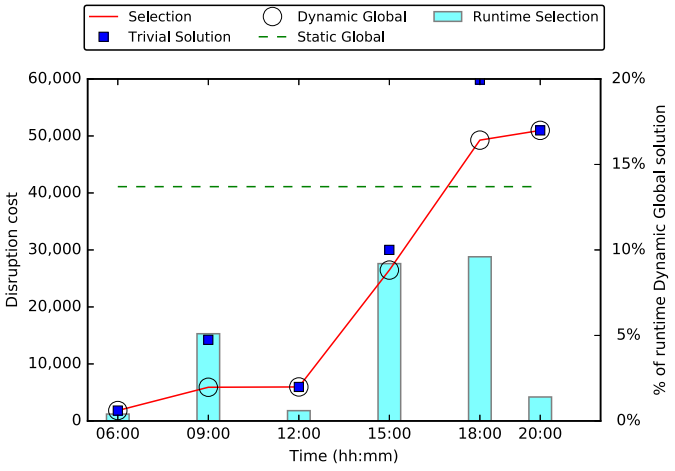


**Fig. 7.** Results for disruption Scenario 1. Comparison of different result types. Computational time required to find solution using SA is indicated by columns as percentage of time required for DG solution.

#### 5.5.1. Scenario 1

The results of Scenario 1 are summarized in Fig. 7. The figure illustrates the development of disruption cost during the day, for the different types of solutions.

In the morning, at 06:00, a small disruption (two delays) occurs. For this disruption, the TR, SA and DG solution were all of the same cost. The TR solution was equal to the DG solution, meaning that it was not possible to mitigate delays by performing tail swaps. In the second disruption set, at 09:00, the TR solution was over $8000 more expensive than the SA and DG solution. However, the latter two were the same, meaning that the SA corresponds to the best possible solution. The time required to find the solution using the SA was 27 s. This is 5.1% of the time required to find the DG solution (534 seconds). At 12:00, a disruption occurs that barely has an effect in terms of disruption cost. At 15:00 and 18:00 disruptions occur for which again costs could be reduced compared to the TR solution. For the 20:00 disruption, the TR, SA and DG solution were equal. This was observed in multiple other scenarios; at the end of the day the recovery options available to the airline are limited and as a result often the TR solution is also the best solution.

For these disruption events, the SA solution was equal to the DG solution. As it can be observed, the time required to find so-

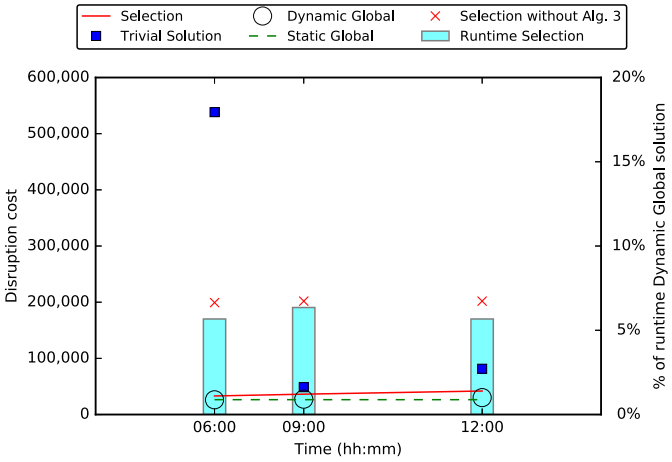


**Fig. 8.** Results for disruption Scenario 5. Also shown are the solutions of the SA if only Algorithm 2 is used.

lutions using the SA are longer when costs can be reduced with respect to the TR solution. In this case, more selections were required before the lowest cost solution was found. Still, required computational time is reduced by over 95% compared to the DG solutions. The SG solution cost for Scenario 1 was 23% lower than the solution obtained when considering the disruptions occurring dynamically throughout the day.

*5.5.2. Scenario 5*

The results of Scenario 5, shown in Fig. 8, illustrate a case where the SA did not find the same solution as the DG solution. In this case the DG approach benefits from the possibility to assess all candidate aircraft at the same time to solve the problem of having three aircraft unavailable for a long time. The SA is limited to re-assigning aircraft that are together in a single selection. As a result the SA solution has greatly reduced cost compared to the TR solution, but was still 39% more costly than the DG solution.

This scenario also allows us to illustrate the value of Algorithm 3, designed for cases involving long unavailability of aircraft. With a red cross we indicate the value that would be obtained by the SA in case this Algorithm 3 is not included. In this case the overall disruption cost for the day would be about 5 times higher.

Scenario 5 is not the only scenario involving the unavailability of aircraft (although this was the only scenario involving the use of Algorithm 3). In fact, for scenarios 1, 2, 3, 8 and 10 aircraft were unavailable and we were able to obtain the *global* optimum or a solution that was only 3% worse than the *global* optimum (for Scenario 2). This indicates that the Algorithm 2 is capable of finding optimal solutions if the disruptions are not extremely severe.

## 6. Robustness test

The results presented in Section 5 were based on ten artificial disruption scenarios. These scenarios were well-suited for demonstration of the progress of the solution during the day of operations, but limited in volume. In order to better evaluate the robustness of the Selection Algorithm, this approach was also tested on 370 generated disruption scenarios. The input data used in these scenarios can be downloaded from https://doi.org/10.4121/uuid:3f38f494-e9a5-447f-af2c-500544999294.

*6.1. Flight schedule*

The robustness test was performed on the flight schedule from a U.S. airline. This flight schedule was obtained from publicly available data of the 'Reporting Carrier On-Time Performance' database, published by the United States Department of Transportation (Bureau of Transportation Statistics, 2019). The schedule used consists of about 2100 flights a day, operated by over 800 aircraft in eight different families. Passenger data was estimated based on the methodology suggested by Barnhart et al. (2014).

Based on this historic data, probability distributions of different disruption events (e.g. flight delay, aircraft unavailability) were derived. Using these distributions, 370 disruption events were randomly generated and solved. Each disruption event contains one or more disruptions. In total these disruption events contain 565 individual flight disruptions. For the sake of simplicity, these events are modelled independently, and not as a sequence of events like we did in the case study presented in Section 5. In this robustness test we have followed the same parameter assumptions present in the case study.

**Table 3**
Normalized results of the Dynamic Global (DG) solution, the trivial solution (TR) and the solution found with the Selection Algorithm (SA). Averages for 370 random disruption events.

| Solution type | Solution time | Solution cost |
|---|---|---|
| DG | 1.00 | 1.00 |
| TR | 0.39 | 5.36 |
| SA | 0.42 | 1.19 |

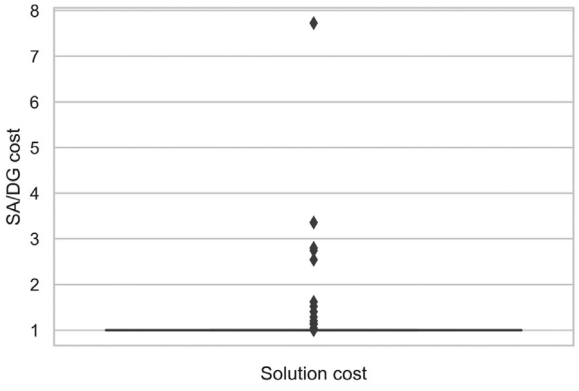


**Fig. 9.** Cost comparison

*6.2. Results*

A summary of the robustness test results are presented in Table 3. This table shows the normalized results in terms of solution time and cost. On average, the SA requires less than the half of the time used by the DG algorithm. It is almost as fast as the trivial solution. However, while the trivial solution is, on average, more than five times worse than the DG solution, the solution costs only deteriorates by 19% when using the SA.

Figs. 9 and 10 show the spread of the 370 different disruption events in a boxplot. As can be seen in Fig. 9, the majority of the runs have a cost which is similar to the solution found by the DG approach (i.e. a value close to 1). For 96% of the 370 scenarios, the SA found a solution with the same cost as the DG solution. In 88% of the 370 events, the SA found the trivial solution. There are a few outliers for which the solution found by SA is more expensive. In five of the 370 events the SA found a solution which was more than twice the cost found by the DG solution. Such cases are rare (1.35% of the cases) and they occur when the DG solution uses a large combination of aircraft that are not considered in the same iteration by the SA.

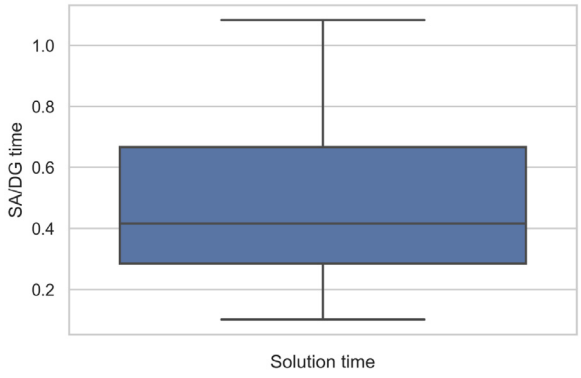


**Fig. 10.** Time comparison

Fig. 10 shows that for the SA the solution time is on average 42% of the time required by the DG solution. This time difference is not as large in this robustness test as it was in the case study (Table 2). The main reason for this is the fact that in this robustness test the disruption events are modelled independently. This makes the DG solution equivalent to the static global (SG). Furthermore, the probability distributions computed for this robustness test rarely generate long delays. These two fact contribute to computational times of the DG that are much lower than in the previous case study. In fact, there are three out of the 370 events for which the SA requires more time than the DG solution. In all three cases, the DG solution time was very small (within the fastest 20%). For these case the overhead required for creating the aircraft selections and solving them iteratively, did not prove to be beneficial for the overall solution time.

## 7. Conclusions

In this paper we presented an operational framework that solves the aircraft recovery problem. Maintenance requirements and passengers itineraries were taken into account when solving the disruptions. This included itinerary disruption for passengers that lost their connections at hub airports. An expedited method was proposed to model these passengers without compromising the computation time of the solution technique proposed. Crew concerns were indirectly modeled by penalizing solutions that potentially have effects on crew schedules.

By sequentially solving subproblems with selections of the airline's fleet, a recovery solution was be found in a fraction of the time that would be needed when considering the entire fleet. A rule-based algorithm was used to sort the aircraft according their potential contribution to solve the disruption. In following these simple rules, the algorithm proved to be very efficient, while generating solutions that are (close to) globally optimal. A case study performed with data from a U.S. airline indicated that a good solution can be achieved within 50 seconds. This is an important achievement for a tool that is intended for operational use. The results suggested that airlines can have a solution in less than 1 minute by only compromising the cost of the solution with 6%–19%, when compared with the globally optimal solution. These results where obtained from a case study of 10 manually designed scenarios, as well as a test for robustness on 370 randomly generated disruption. The operational tool was built in such a way that a solution is always provided to the user, highlighting elements in the solution that may be the cause of infeasibility. A third relevant feature of the operational tool was the fact that is prepared to solve the disruption problem in a dynamic way: i.e., re-solving the airline's schedule every time new disruption information is made available, starting from the solution for the previous disruption event and revoking previous decision(s) if it proved to improve the solution and if there was still time to change operations. With all these ingredients we are convinced that we presented a unique approach for operational use by airlines.

The framework presented in this paper has limitations to consider in future research. First, passenger itineraries are assumed to be static. That is, passengers are not rerouted or rebooked to different classes. Second, crew schedules are indirectly modelled. Third, the selection algorithm is a rule-based heuristic that may require adjustments for the network of individual airlines. The first two limitations are of particular interest for the development of a completely integrated airline disruption management operational tool. However, it will be a major challenge for future development to meet the computation time requirements for this integrated tool. This could be, in part, solved by addressing the last limitation. For instance, a promising area of development would be to consider a more efficient machine learning technique to select the best selection of aircraft to solve a set of disruptions.

## CRediT authorship contribution statement

**J. Vink:** Methodology, Software, Writing - original draft. **B.F. Santos:** Conceptualization, Supervision, Writing - review & editing. **W.J.C. Verhagen:** Conceptualization, Writing - review & editing. **I. Medeiros:** Resources, Validation. **R. Filho:** Resources, Validation.

## Acknowledgments

We thank the two anonymous reviewers whose comments helped improve and clarify this manuscript.

## References

Arikan, U., Gürel, S., Aktürk, M.S., 2016. Integrated aircraft and passenger recovery with cruise time controllability. Annals Oper. Res. 236 (2), 295–317. doi:10.1007/s10479-013-1424-2.

Bard, J.F., Yu, G., Argüello, M.F., 2001. Optimizing aircraft routings in response to groundings and delays. IIE Trans. 33 (10), 931–947. doi:10.1023/A:1010987008497.

Barnhart, C., Fearing, D., Vaze, V., 2014. Modeling passenger travel and delays in the national air transportation system. Oper. Res. 62 (3), 483–711. doi:10.1287/opre.2014.1268.

Bisaillon, S., Cordeau, J.-F., Laporte, G., Pasin, F., 2011. A large neighbourhood search heuristic for the aircraft and passenger recovery problem. Oper. Res. 9, 139–157. doi:10.1007/s10288-010-0145-5.

Bratu, S., Barnhart, C., 2006. Flight operations recovery: New approaches considering passenger recovery. J. Schedul. 9 (3), 279–298. doi:10.1007/s10951-006-6781-0.

Bureau of Transportation Statistics, 2019. Airline On-Time Performance Data. Technical Report. United States Department of Transportation.

Clausen, J., Larsen, A., Larsen, J., Rezanova, N.J., 2010. Disruption management in the airline industry - Concepts, models and methods. Comput. Oper. Res. 37 (5), 809–821. doi:10.1016/j.cor.2009.03.027.

Cook, A., Tanner, G., Lawes, A., 2012. The hidden cost of airline unpunctuality. J. Transport Econ. Policy (JTEP) 46 (2), 157–173.

Eggenberg, N., Salani, M., Bierlaire, M., 2010. Constraint-specific recovery network for solving airline recovery problems. Comput. Oper. Res. 37 (6), 1014–1026. doi:10.1016/j.cor.2009.08.006.

EUROCONTROL, 2019. All-Causes Delay and Cancellations to Air Transport in Europe - Q1 2019. Technical Report. EUROCONTROL.

Hu, Y., Xu, B., Bard, J.F., Chi, H., Gao, M., 2015. Optimization of multi-fleet aircraft routing considering passenger transiting under airline disruption. Comput. Ind. Eng. 80, 132–144. doi:10.1016/j.cie.2014.11.026.

Jarrah, A.I.Z., Yu, G., Krishnamurthy, N., Rakshit, A., 1993. A Decision Support Framework for Airline Flight Cancellations and Delays. Transp. Sci. 27 (3), 266–280. doi:10.1287/trsc.27.3.266.

Jozefowiez, N., Mancel, C., Mora-Camino, F., 2013. A heuristic approach based on shortest path problems for integrated flight, aircraft, and passenger rescheduling under disruptions. J. Oper. Res. Soc. 64 (3), 384–395. doi:10.1057/jors.2012.20.

Maher, S.J., 2015. A novel passenger recovery approach for the integrated airline recovery problem. Comput. Oper. Res. 57, 123–137. doi:10.1016/j.cor.2014.11.005.

Marla, L., Vaaben, B., Barnhart, C., 2017. Integrated disruption management and flight planning to trade off delays and fuel Burn. Transp. Sci. 51 (1), 88–111. doi:10.1287/trsc.2015.0609.

Petersen, J.D., Sölveling, G., Clarke, J.-P., Johnson, E.L., Shebalov, S., 2012. An optimization approach to airline integrated recovery an optimization approach to airline integrated recovery. Transp. Sci. 46 (4), 482–500. doi:10.1287/trsc.1120.0414.

Santos, B.F., Wormer, M.M., Achola, T.A., Curran, R., 2017. Airline delay management problem with airport capacity constraints and priority decisions. J. Air Transport Manage. 63, 34–44. https://doi.org/10.1016/j.jairtraman.2017.05.003.

Sinclair, K., Cordeau, J.F., Laporte, G., 2014. Improvements to a large neighborhood search heuristic for an integrated aircraft and passenger recovery problem. Eur. J. Oper. Res. 233 (1), 234–245. doi:10.1016/j.ejor.2013.08.034.

Sinclair, K., Cordeau, J.-F., Laporte, G., 2016. A column generation post-optimization heuristic for the integrated aircraft and passenger recovery problem. Comput. Oper. Res. 65 (1), 42–52. doi:10.1016/j.ejor.2013.08.034.

Teodorović, D., Guberinić, S., 1984. Optimal dispatching strategy on an airline network after a schedule perturbation. Eur. J. Oper. Res. 15 (2), 178–182. doi:10.1016/0377-2217(84)90207-8.

Teodorović, D., Stojković, G., 1990. Model for operational daily airline scheduling. Transp. Plann. Technol. 14 (4), 273–285. doi:10.1080/03081069008717431.

Teodorović, D., Stojković, G., 1995. Model to Reduce Airline Schedule Disturbances. J. Transp. Eng. 121 (4), 324–331. doi:10.1061/(ASCE)0733-947X(1995)121:4(324).

Thengvall, B.G., Bard, J.F., Yu, G., 2000. Balancing user preferences for aircraft schedule recovery during irregular operations. IIE Trans. (Inst. Ind.Eng.) 32 (3), 181–193. doi:10.1023/A:1007618928820.

Thengvall, B.G., Bard, J.F., Yu, G., 2003. A Bundle Algorithm Approach for the Aircraft Schedule Recovery Problem During Hub Closures. Transp. Sci. 37 (4), 392–407. doi:10.1287/trsc.37.4.392.23281.

Vos, H.W.M., Santos, B.F., Omondi, T., 2015. Aircraft Schedule Recovery Problem – A Dynamic Modeling Framework for Daily Operations. Transp. Res. Procedia 10, 931–940. doi:10.1016/j.trpro.2015.09.047.

Yan, S., Yang, D.-H., 1996. A decision support framework for handling schedule perturbation. Transp. Res. Part B 30 (6), 405–419. doi:10.1016/0191-2615(96)00013-6.