ROMA TRE
UNIVERSITÀ DEGLI STUDI

Dipartimento di Ingegneria
Via della Vasca Navale, 79
00146 Roma, Italy

Metaheuristics for efficient aircraft scheduling and re-routing at busy terminal control areas

Marcella Samà [1], Andrea D'Ariano [1], Francesco Corman[2], Dario Pacciarelli [1]

RT-DIA-213-2015 **Gennaio 2015**

(1) Università degli Studi Roma Tre,
Dipartimento di Ingegneria,
Sezione Informatica e Automazione,
Via della Vasca Navale, 79
00146 Roma, Italy

(2) Delft University of Technology,
Department of Maritime and Transport Technology,
Section of Transport Engineering and Logistics,
Mekelweg, 2
2628 CD Delft, The Netherlands.

We acknowledge support from Ing. Giacomo Zaninotto and Ing. Alessandro Toli.

ABSTRACT

Intelligent decision support systems for the real-time management of landing and take-off operations can be very effective in helping traffic controllers to limit airport congestion at busy terminal control areas. The key optimization problem to be solved to this aim can be formulated as a mixed integer linear program. However, since this problem is strongly NP-hard, heuristic algorithms are typically adopted in practice to compute good quality solutions in a short computation time. This paper presents a number of algorithmic improvements implemented in the AGLIBRARY solver in order to improve the possibility of finding good quality solutions quickly. The proposed framework starts from a good initial solution for the scheduling problem with fixed routes, obtained via a truncated branch-and-bound algorithm. A metaheuristic is then applied to improve the solution by re-routing some aircraft. The new metaheuristics are based on variable neighbourhood search, tabu search and hybrid schemes. The neighbourhoods differ from each other for the number of aircraft that are re-routed in each move. Computational experiments are performed on an Italian terminal control area under various types of disturbances, including multiple aircraft delays and a temporarily disrupted runway. The metaheuristics achieve solutions of remarkable quality, within a small computation time, compared with a commercial solver and with the previous versions of AGLIBRARY.

Keywords: Optimal Air Traffic Control; Landing and Take-Off Operations; Disruption Management; Disjunctive Programming; Variable Neighbourhood Search; Tabu Search; Hybrid Algorithms.

1 Introduction

1.1 The investigated problem

In the last years, air traffic controllers are experiencing increasing difficulties to manage the ever increasing transport flows while ensuring safety and efficiency of operational schedules. This is partially due to the limited space and funds to build new infrastructure in bottleneck areas, but also due to the limited support offered by the actual traffic control systems. One typical bottleneck of the entire air traffic system is the terminal control area (TCA). During operations, aircraft delays are considered to cause a substantial cost from both airlines and passengers' points of view. The computation of optimal aircraft landing and take-off schedules is thus one of the most relevant operational problems. These facts stimulated the interest for effective intelligent transport system solutions that can show how to better use the existing resources [35].

The aim of this paper is to develop good quality solutions for the Air Traffic Control in a Terminal Control Area (ATC-TCA) problem. This problem consists of simultaneously determining the routing (i.e. the resources to be traversed), sequencing (i.e. the orders between aircraft) and timing of landing and take-off aircraft on the TCA resources, which may include several runways and air segments. The objective is to minimize the maximum positive deviation from the target landing and take-off times. The resolution of this problem requires to consider several factors related to safety, efficiency and equity [5]. Safety requires the careful modelling of practical TCA constraints, efficiency consists of reducing aircraft delays with global conflict detection and resolution approaches, equity can be achieved by the minimization of the largest delay due to conflicting aircraft routes, that requires the consideration of all aircraft travelling in the network during the studied traffic horizon.

1.2 The related literature

In the aircraft scheduling literature, it is often mentioned a big gap existing between the level of sophistication of published results and algorithms and the simple methods that are employed in practice. One motivation reported for this gap is that the theory typically addresses very simplified problems for which optimal or near-optimal performance can be achieved, while the practice must face all the complexity of real-time operations, often with little regard to the performance level. However, poorly performing aircraft scheduling and routing methods that are used in practice directly impact on the quality of service offered to the passengers, the effect being more evident as traffic density gets close to saturation. In fact, in this case, since any small disturbance may propagate to other aircraft, altering the regularity of air traffic even some hours after the end of the original disturbance.

The recent trend of research is to incorporate more practical constraints in the detailed (microscopic) models, since too simplified (macroscopic) models may have a limited impact on the practice of air traffic control. In view of the extensive reviews reported in [4, 6, 9, 22, 23, 28], we limit our review of the related literature to two streams of research: (i) the development of microscopic models for the management of aircraft flows in terminal control areas, (ii) the development of algorithmic methods in air traffic control.

Regarding stream (i), there are a few microscopic models that are able to incorporate

all detailed information that is compliant with the safety regulations of the TCA, including the characteristics of the airport infrastructure resources and the individual flight paths. Such a level of detail is required to safely detect and solve potential conflicting routes at the level of runways, ground and air segments of the TCA. The most detailed model used in this context is the job shop scheduling model in which each operation denotes the traversal of a resource (air/ground segment, runway) by a job (aircraft). The variables are the start time of each operation to be performed by an aircraft on a specific resource. A *no-wait* version of this model has been firstly proposed in [10] and successively extended in [14, 13, 15, 24] as a *blocking and no-wait* version. In the latter approach, air segment resources are treated as no-wait resources with time windows for modelling minimum/maximum aircraft travel times, while runway resources are treated as blocking resources which can host at most one aircraft at a time.

Regarding stream (ii), exact and heuristic algorithms have been proposed for the ATC-TCA problem. However, the former algorithms can quickly compute near-optimal solutions only for quite small instances. Consequently, numerous metaheuristics have been recently proposed to search for good quality solutions in a short computation time, the most used being the following: genetic algorithms [7, 18, 20], scatter search [29], tabu search [3, 15], ant colony [8, 21, 36], simulated annealing [17, 30, 34], iterated local search [27], variable neighbourhood search [1, 2, 27, 30, 31]. Several of the proposed algorithms have also been hybridized in order to combine interesting properties and to take the best from each of them. All these approaches have proposed significant improvements compared to the commonly used air traffic control rules, such as the first-in-first-out rule. In fact, the usual control rules take a few sequencing and routing decisions at a time in a myopic fashion, ignoring the propagation of delays to other aircraft in the network [20].

In view of the above discussion of the recent literature regarding the management of landing and take-off operations, there is a clear need to incorporate an increasing level of detail and realism in the models while keeping the computation time of the algorithms at an acceptable level. Furthermore, the ATC-TCA problem is well-known to be NP-hard, requiring the use of advanced heuristics, especially when solving complex instances with multiple delayed aircraft and severe resource capacity deficiencies. This paper deals with the real-world instances of Samà et al. [33], with up to more than 200000 scheduling variables and more than 400 routing variables. Since it is not possible to solve these instances with an exact method in a reasonable amount of time, we focus our work to the development of hybrid metaheuristics (based on tabu search and variable neighbourhood search schemes) to derive good quality solutions in a short time.

1.3 The paper contribution

A recent stream of research on detailed ATC-TCA problem formulations focuses on the Alternative Graph (AG) of Mascis and Pacciarelli [24]. This graph has been first successfully applied to manage other transportation and production problems [11, 12, 26]. In this paper, the ATC-TCA problem is modelled as a generalized job shop scheduling problem via alternative graphs, enriching the model of [10] by the addition of real-world constraints. This graph allows a more accurate modelling of relevant TCA aspects and safety constraints, such as holding circles, waiting in flight before landing, travelling in feasible time windows, hosting multiple aircraft simultaneously in air segments and individual aircraft simultaneously in runways. In order to include the routing flexibility in the

AG model, we make use of the Mixed Integer Linear Programming (MILP) formulation of [33]. The MILP formulation can been efficiently solved by the rolling horizon framework of [32]. However, the latter approach requires a large computation time when dealing with complex ATC-TCA instances.

This paper presents a number of algorithmic improvements implemented in the solver AGLIBRARY, a set of optimization algorithms for complex job shop scheduling problems developed at Roma Tre University. The solver is based on the following framework: a good initial solution for the scheduling problem with fixed routes is computed by the (truncated) branch-and-bound algorithm in [13, 14]. Metaheuristics are then applied to improve the solution by re-routing some aircraft. This action corresponds to the concept of a move, from a metaheuristics perspective. In [15], a tabu search algorithm has been applied to solve practical-size instances for small disturbances. Previous research left open the following two relevant issues. The first issue concerns the extent at which different solution methods might outperform the tabu search algorithm and the rolling horizon framework. A second issue is to study algorithmic improvements, in order to reduce the time to compute good quality solutions. Both these issues motivate the development of the new metaheuristics proposed in this paper. The paper contributions are next outlined:

- We present new routing neighbourhoods that differ from each other for the number of aircraft that are re-routed in each move and for the set of candidate aircraft to be re-routed;
- We alternate the search for promising moves in neighbourhoods of different size, similarly to [25], and present strategies for searching within these neighbourhoods based on variable neighbourhood search, tabu search and hybrid schemes;
- We implement fast rescheduling heuristics for the evaluation of each neighbour;
- We apply the proposed algorithms to the management of complex disturbed situations, including multiple delayed landing and/or take-off aircraft and a temporarily disrupted runway. The situations tested are the most complex instances in [33]. The new metaheuristics are compared with the other existing methods based on the AG model, and with solutions computed with a commercial MILP solver. Significantly better results are obtained in terms of an improved solution quality and/or a reduced computation time with respect to both the MILP solver and the previous versions of AGLIBRARY.

Section 2 formally describes the ATC-TCA problem and the MILP formulation. Section 3 presents the metaheuristic algorithms proposed in this paper. Section 4 reports the performance of the algorithms on the MXP instances of Samà et al. [33]. Section 5 summarizes the paper results and outlines future research directions. An appendix illustrates the neighbourhoods used in this paper with a numerical example.

2 Problem definition and formulation

2.1 The ATC-TCA problem

Landing aircraft move in the landing air segments of the TCA, following a standard descent profile, from an air entry point to a common glide path, that is the final landing

air segment before the runway. Take-off aircraft move in the ground resources till they get access to the runway and finally fly toward their assigned exit point via take-off air segments.

A minimum longitudinal and diagonal safety separation distance between every pair of consecutive aircraft must be always respected, depending on their type, altitude and relative positions. This minimum distance can be translated into a *minimum separation time* that is sequence-dependent, since it depends on the relative processing order of the common resources by the different aircraft categories (e.g. heavy, medium and light aircraft).

Each aircraft has a *processing time* on each TCA resource, according to its landing/take-off profile. On the air segments, the processing time varies between minimum and maximum feasible values.

Each landing/take-off aircraft has a minimum entrance time into the TCA, *release time*, according to its current position and speed. Landing aircraft can also be constrained to have a maximum entrance time, *deadline time*, into the TCA, e.g. due to limited fuel availability.

All aircraft have scheduled times, *due date times*, to start processing some TCA resources. A departing aircraft is supposed to take-off within its assigned time window and is late whenever it is not able to accomplish the departing procedure within its assigned time window. Following the procedure commonly adopted by air traffic controllers, we consider a time window for take-off between 5 minutes before and 10 minutes after the *Scheduled Take-off Time* (STT). A departing aircraft is considered delayed in exiting the TCA if leaving the runway after 10 minutes from its STT. Arriving aircraft are late if landing after their *Scheduled Landing Time* (SLT).

Before entering the TCA, landing aircraft can fly in *holding circles* that are air segments dedicated to accumulating aircraft delays during the flight. In each holding circle, landing aircraft must fly at a fixed speed for a number of half circles, as prescribed by the air traffic controller. Departing aircraft instead can be delayed in entering the TCA at ground level, i.e. before entering the runway.

We use the following notation for aircraft delays. *Entrance delay* (*exit delay*) is the delay of an aircraft on the entrance to (the exit from) the TCA. The exit value is partly a consequence of a possible late entrance, which causes an *unavoidable delay*, and partly due to additional delays caused by the resolution of potential aircraft conflicts in the TCA, which is the *consecutive delay*. In this paper, we minimize the maximum consecutive delay that is an equitable approach for the minimization of aircraft delay propagation.

The next section will show a model of ATC-TCA problem in which a route is assigned to each aircraft. This assumption will be then relaxed in order to deliver a general optimization model.

2.2 The AG model

This subsection presents the alternative graph for the ATC-TCA problem with pre-defined routes. This graph is a triple $G = (N, F, A)$: $N = \{s, 1, ..., n-2, t\}$ is the set of *nodes*, where nodes s and t represent the start and the end operations of the schedule, while the other $n-2$ nodes are related to the start of the other operations; F is the set of *fixed directed arcs* that model the pre-defined aircraft routes; A is the set of *alternative pairs* that model the sequencing decisions. Each pair is composed of two directed arcs.

Each node, except s and t, is associated with the start of an operation krj, where k indicates the aircraft, r the route chosen and j the resource it traverses. The start time h_{krj} of operation krj is the entrance time of aircraft k with route r in resource j.

Each fixed arc $(krp, krj) \in F$ is identified by the two nodes krp and krj that are connected, and it has associated the weight $w^F_{krp\_krj}$, representing a minimum time constraint between h_{krp} and h_{krj} (i.e. $h_{krj} \geq h_{krp} + w^F_{krp\_krj}$). The set F is the union of the disjoint sets: F_{rt} is the set of release time constraints, F_{dt} is the set of due date time constraints, F_{Dt} is the set of deadline time constraints, F_{HC} is the set of holding circle constraints, F_{AS} is the set of air segment constraints, F_{RW} is the set of runway constraints.

Each alternative pair $((krp, dij), (uml, vnw)) \in A$ models an aircraft sequencing or holding decision. The two arcs of the pair have associated the weights $w^A_{krp\_dij}$ and $w^A_{uml\_vnw}$. In any solution, only one arc of each pair can be selected. If alternative arc (krp, dij) $[(uml, vnw)]$ is selected in a solution, the constraint $h_{dij} \geq h_{krp} + w^A_{krp\_dij}$ $[h_{vnw} \geq h_{uml} + w^A_{uml\_vnw}]$ has to be satisfied. The set A is composed by the subsets: A_{HC} for holding decisions, A_{AS} and A_{RW} for sequencing decisions at air segments and runways.

A selection S is a set of alternative arcs obtained by selecting exactly one arc from each alternative pair in A and such that the resulting graph $\mathcal{G}(F, S) = (N, F \cup S)$ does not contain positive weight cycles. A solution to the ATC-TCA problem with pre-defined routes is a selection S. This allows to associate orders and times to all operations. The minimization of the maximum consecutive delay is measured as a makespan minimization. Given a selection S and any two nodes krp and uml, we let $l^S(krp, uml)$ be the weight of the longest path from krp to uml in $\mathcal{G}(F, S)$. By definition, the start time h_{krp} of $krp \in N$ is the quantity $l^S(s, krp)$, which implies $h_s = 0$ and $h_t = l^S(s, t)$.

2.3 The MILP formulation

The ATC-TCA problem with flexible routes is formulated as a particular *disjunctive program* [33]. This is achieved via a MILP formulation in which the scheduling and routing decisions are considered simultaneously. The starting point is the alternative graph model for the ATC-TCA problem with pre-defined routes. The graph is formulated via a $big - M$ formulation enlarging the sets F and A in order to include the fixed and alternative arcs related to all possible aircraft routes. Each fixed arc translates into a constraint, while each alternative pair into a pair of alternative constraints. For each alternative pair $((krp, dij), (uml, vnw)) \in A$ there is a binary variable $x^{uml,vnw}_{krp,dij}$ modelling the sequencing/holding decision. For each route r and aircraft k there is a binary variable y_{kr} modelling the possible route selection. For each operation krp there is a non-negative real variable h_{krp} modelling its start time. The MILP formulation is the following.

$$\min \; h_t - h_s \tag{1}$$

$$\sum_{r=1}^{R_k} y_{kr} = 1 \quad k = 1, ..., Z \tag{2}$$

$$h_{krj} - h_s + M(1 - y_{kr}) \geq w^{F_{rt}}_{s\_krj} \quad \forall (s, krj) \in F_{rt} \tag{3}$$

$$h_t - h_{krj} + M(1 - y_{kr}) \geq w^{F_{dt}}_{krj\_t} \quad \forall (krj, t) \in F_{dt} \tag{4}$$

$$h_s - h_{krj} + M(1 - y_{kr}) \geq w^{F_{Dt}}_{krj\_s} \quad \forall (krj, s) \in F_{Dt} \tag{5}$$

$$\begin{aligned} h_{krj} - h_{krp} + M(1 - y_{kr}) &\geq w^{F_{HC}}_{krp\_krj} \quad \forall (krp, krj) \in F_{HC} \\ h_{krp} - h_{krj} + M(1 - y_{kr}) &\geq w^{F_{HC}}_{krj\_krp} \quad \forall (krj, krp) \in F_{HC} \end{aligned} \tag{6}$$

$$\begin{aligned} &h_{krj} - h_{krp} + M x^{krj\_krp}_{krp\_krj} + M(1 - y_{kr}) \geq w^{A_{HC}}_{krp\_krj} \\ &h_{krp} - h_{krj} + M(1 - x^{krj\_krp}_{krp\_krj}) + M(1 - y_{kr}) \geq w^{A_{HC}}_{krj\_krp} \quad \forall ((krp, krj), (krj, krp)) \in A_{HC} \end{aligned} \tag{7}$$

$$\begin{aligned} h_{krl} - h_{krm} + M(1 - y_{kr}) &\geq w^{F_{AS}}_{krm\_krl} \quad \forall (krm, krl) \in F_{AS} \\ h_{krm} - h_{krl} + M(1 - y_{kr}) &\geq w^{F_{AS}}_{krl\_krm} \quad \forall (krl, krm) \in F_{AS} \end{aligned} \tag{8}$$

$$\begin{aligned} &h_{uim} - h_{krm} + M x^{uin\_krl}_{krm\_uim} + M(2 - y_{kr} - y_{ui}) \geq w^{A_{AS}}_{krm\_uim} \\ &h_{krl} - h_{uin} + M(1 - x^{uin\_krl}_{krm\_uim}) + M(2 - y_{kr} - y_{ui}) \geq w^{A_{AS}}_{uin\_krl} \quad \forall ((krm, uim), (uin, krl)) \in A_{AS} \\ &h_{krm} - h_{uim} + M x^{krl\_uin}_{uim\_krm} + M(2 - y_{kr} - y_{ui}) \geq w^{A_{AS}}_{uim\_krm} \\ &h_{uin} - h_{krl} + M(1 - x^{krl\_uin}_{uim\_krm}) + M(2 - y_{kr} - y_{ui}) \geq w^{A_{AS}}_{krl\_uin} \quad \forall ((uim, krm), (krl, uin)) \in A_{AS} \end{aligned} \tag{9}$$

$$h_{kro} - h_{krj} + M(1 - y_{kr}) \geq w^{F_{RW}}_{krj\_kro} \quad \forall (krj, kro) \in F_{RW} \tag{10}$$

$$\begin{aligned} &h_{uij} - h_{kro} + M x^{uig\_krj}_{kro\_uij} + M(2 - y_{kr} - y_{ui}) \geq w^{A_{RW}}_{kro\_uij} \\ &h_{krj} - h_{uig} + M(1 - x^{uig\_krj}_{kro\_uij}) + M(2 - y_{kr} - y_{ui}) \geq w^{A_{RW}}_{uig\_krj} \quad \forall ((kro, uij), (uig, krj)) \in A_{RW} \end{aligned} \tag{11}$$

$$x^{krj\_krp}_{krp\_krj} \in \{0, 1\} \quad \forall ((krp, krj), (krj, krp)) \in A_{HC} \tag{12}$$

$$\begin{aligned} x^{uin\_krl}_{krm\_uim} &\in \{0, 1\} \quad \forall ((krm, uim), (uin, krl)) \in A_{AS} \\ x^{krl\_uin}_{uim\_krm} &\in \{0, 1\} \quad \forall ((uim, krm), (krl, uin)) \in A_{AS} \end{aligned} \tag{13}$$

$$x^{uig\_krj}_{kro\_uij} \in \{0, 1\} \quad \forall ((kro, uij), (uig, krj)) \in A_{RW} \tag{14}$$

$$y_{kr} \in \{0, 1\} \quad k = 1, ..., Z \ ; \ r = 1, ..., R_k \tag{15}$$

The objective function is reported in Equation 1. We next describe the ATC-TCA problem constraints.

Constraints 2 model the routing decision for each aircraft k among its set of R_k routes. The route r is chosen for aircraft k if and only if $y_{kr} = 1$. In total, there are Z aircraft.

Constraints 3 model the *release arcs* $(s, krj) \in F_{rt} \subset F$. The weight $w^{F_{rt}}_{s\_krj}$ is the release time to start processing operation krj, i.e. the earliest entrance time of aircraft k in resource j when using route r.

Constraints 4 model the *due date arcs* $(krj, t) \in F_{dt} \subset F$. The weight $w^{F_{dt}}_{krj\_t}$ is the due date time to start processing operation krj, i.e. the scheduled arrival time of aircraft k in resource j when using route r.

Constraints 5 model the *deadline arcs* $(krj, s) \in F_{Dt} \subset F$. The weight $w^{F_{Dt}}_{krj\_s}$ is the deadline time to start processing operation krj, i.e. the latest possible arrival time of aircraft k in j when using route r.

Constraints 6 and 7 model the holding decisions regarding the holding circle resources. Let krp/krj be the operations regarding the entrance p/the exit j resource of aircraft k with route r in/from the holding circle. Constraints 6 model the *holding arcs* (krp, krj) and $(krj, krp) \in F_{HC} \subset F$, with weights $w^{F_{HC}}_{krp\_krj} = 0$ and $w^{F_{HC}}_{krj\_krp} = -\phi$, where ϕ is the time required to perform a half circle. Constraints 7 model the *holding pairs of alternative arcs* $((krp, krj), (krj, krp)) \in A_{HC} \subset A$, with weights $w^{A_{HC}}_{krp\_krj} = \phi$ and $w^{A_{HC}}_{krj\_krp} = 0$. When the alternative arc $((krp, krj) \in A_{HC}$ is selected, aircraft k with route r performs

a half circle in the corresponding holding circle resource. This decision corresponds to fix the binary variable $x^{krj\_krp}_{krp\_krj} = 0$. The formulation of multiple half circles can be viewed as a generalization of the single holding decision.

Constraints 8 model the minimum/maximum travel time constraints in the air segments. Let krm/krl be the entrance m/the exit l of aircraft k with route r in/from the air segment. The two *air segment arcs* (krm, krl) and $(krl, krm) \in F_{AS} \subset F$ model the minimum w_{min} and the maximum w_{max} travel time, with weights $w^{F_{AS}}_{krm\_krl} = w_{min}$ and $w^{F_{AS}}_{krl\_krm} = -w_{max}$.

Constraints 9 model the sequencing decisions in the air segment resources. Since an overtake between an aircraft k with route r and an aircraft u with route i in the same air segment m is not allowed for safety reasons, the entrance and exit orders between the two aircraft must be the same over such resource. The sequencing decision at air segment m with no-intra-segment-overtake-constraint is modelled by the two *air segment pairs of alternative arcs* $((krm, uim), (uin, krl))$ and $((uim, krm), (krl, uin)) \in A_{AS} \subset A$. We note that l and n are the successive resources for aircraft k with route r and aircraft u with route i, respectively. The weights $w^{A_{AS}}_{krm,uim}$ and $w^{A_{AS}}_{uim,krm}$ [$w^{A_{AS}}_{uin,krl}$ and $w^{A_{AS}}_{krl,uin}$] model the time separation at the entrance [exit] of the air segment m. This separation time is sequence-dependent since it depends on the characteristics of aircraft k and aircraft u. The selection of one arc for each alternative pair models the order in which the two aircraft enter/exit the air segment. Even if there are four possible selections of the arcs of the two pairs, there are only two feasible sequencing solutions: Either aircraft k or u enters first and exits first air segment m. When aircraft k is first, the two alternative arcs (krm, uim) and (krl, uin) are selected. This decision corresponds to fix the binary variables $x^{uin\_krl}_{krm\_uim} = 0$ and $x^{krl\_uin}_{uim\_krm} = 1$. Alternatively, aircraft u is first and the other two alternative arcs (uim, krm) and (uin, krl) are selected, i.e. $x^{uin\_krl}_{krm\_uim} = 1$ and $x^{krl\_uin}_{uim\_krm} = 0$.

Constraints 10 model the *runway arcs* $(krj, kro) \in F_{RW} \subset F$, of weight $w^{F_{RW}}_{krj\_kro}$ equal to the processing time of runway j by aircraft k with route r.

Constraints 11 represent the *runway pairs of alternative arcs* $((kro, uij), (uig, krj)) \in A_{RW} \subset A$, of weights $w^{A_{RW}}_{kro\_uij}$ and $w^{A_{RW}}_{uig\_krj}$ equal to the sequence-dependent separation time in runway j between the two aircraft. Since the runway is a blocking resource, o and g are the successive resources with respect to runway j for aircraft k with route r and aircraft u with route i, respectively. The selection of one arc for the alternative pair models the order in which the aircraft use the runway. When aircraft k is first, the alternative arc (kro, uij) is selected. This decision corresponds to fix the binary variable $x^{uig\_krj}_{kro\_uij} = 0$.

Constraints $12 - 15$ are for the x and y binary variables.

3 Scheduling and re-routing algorithms

This section describes the algorithmic approaches proposed in this paper to solve the ATC-TCA problem. Section 3.1 presents the general framework of the solver that is based on a combination of aircraft scheduling and re-routing algorithms. Section 3.3 describes the neighbourhoods for the search of new aircraft routes starting from a routing and scheduling solution, Section 3.4 the scheduling heuristic procedure used to evaluate the neighbours (the new routing combinations), Section 3.2 the algorithm used to compute a new aircraft schedule for given routes. The routing neighbourhoods and the scheduling

algorithms are used in Sections 3.5, 3.6, 3.7 that describe the metaheuristics developed and tested in this paper.

3.1 Solution framework

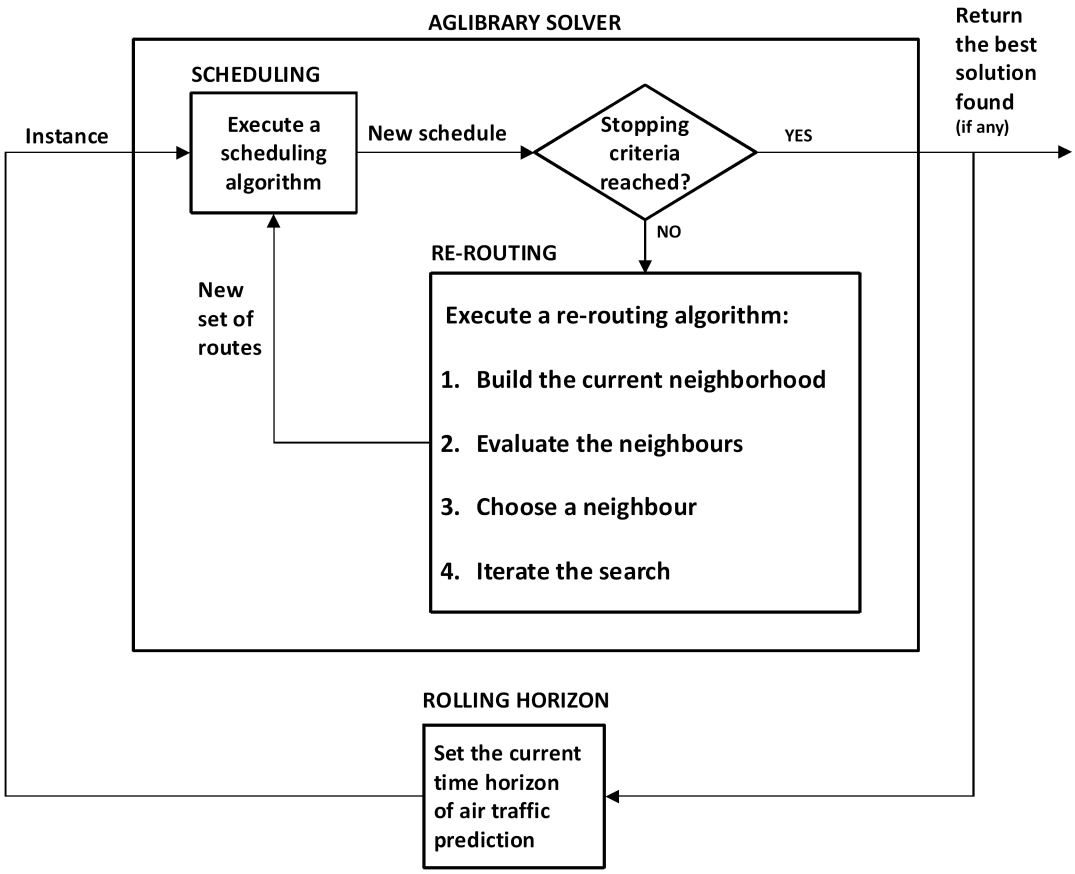


Figure 1: A general scheme of the solver

Figure 1 illustrates the general scheme of the solver. Since the ATC-TCA problem is an NP-hard problem, we adopt a temporal decomposition and a decomposition in routing and scheduling variables. The former is solved via the rolling horizon procedure in [32], while the latter is solved via the scheduling and routing algorithms of the AGLIBRARY solver. Specifically, we use the scheduling algorithms in [14, 13], the re-routing algorithms in [15], the new scheduling and re-routing algorithms developed in this paper. The two decomposition frameworks can be further combined together.

The rolling horizon decomposition framework divides the ATC-TCA problem into time horizons of traffic predictions. Each time horizon is a sub-problem instance to be solved by the AGLIBRARY solver. We assume that all aircraft information is known at the start time t_0 of the traffic prediction. This rolling horizon framework corresponds to a centralized framework when the overall problem is solved with a single time horizon (i.e. when no temporal decomposition is performed).

The decomposition framework into routing and scheduling works instead as follows. The AGLIBRARY solver iterates between the computation of a new aircraft schedule for

a set of routes, and the selection of a new set of routes. The basic idea is to first compute an aircraft scheduling solution given fixed (default) routes, and then search for better aircraft routes. The latter procedure is based on a local search for routing alternatives starting from the scheduling solution, and an iterative scheduling and routing technique to continue the search. The iterative procedure returns the best aircraft schedule and the best set of routes after a stopping criteria is reached. In this paper, the maximum computation time is a stopping criteria.

The overall framework returns a feasible aircraft schedule in which a route is fixed for each aircraft and all potential routing conflicts are solved. In case no feasible schedule is computed, the solver reports the conflicting routes via a detailed time-space diagram. Based on the information provided by the solver, the en-route/ground human traffic controllers could take suitable rescheduling actions on the potential conflicts that are not allowed by the automated decision support system, including re-routing some aircraft to other resources in the same or other airports.

3.2 Branch-and-bound scheduling algorithm

The ATC-TCA problem with fixed routes is solved by the branch-and-bound (BB) algorithm of D'Ariano et al. [14, 13], truncated at a given maximum computation time. A near-optimal solution is computed in a short time by this algorithm for practical-size instances. In particular, the algorithm is based on a binary branching scheme in which the branching decision is either a sequencing order between two aircraft in a TCA resource or a holding decision for an aircraft in a holding circle. In the alternative graph model, this sequencing decision corresponds to the selection of an alternative arc from each pair $((krp, dij), (uml, vnw)) \in A$. The branching decision is thus on the arcs (krp, dij) and (uml, vnw). Since the runways are the bottleneck resources of the TCA, the branching decisions are prioritized by giving precedence to the sequencing decisions on the runway resources (i.e. to the alternative pairs $\in A_{RW}$).

3.3 Routing neighbourhoods

This subsection describes the neighbourhood structures used in this paper. To this aim, we need to introduce the following notation. Let $S(F)$ be a ATC-TCA solution with the routes defined in F and the sequencing decisions defined in S, and let $\mathcal{G}(F, S)$ be the graph of this solution. The search for a better solution is based on the computation of a new graph $\mathcal{G}'(F', S')$ This graph differs from the former $\mathcal{G}(F, S)$ by a different route for some aircraft, and different orders and times of operations. This corresponds to a neighbour, in metaheuristics terms. The longest path in $\mathcal{G}'(F', S')$ is denoted as $l^{S'(F')}(s, t)$. We observe that F' improves over F in terms of the objective function value if $l^{S'(F')}(s, t) < l^{S(F)}(s, t)$.

The neighbourhoods studied in this paper are based on observations on the graph $\mathcal{G}(F, S)$ regarding the nodes that represent operations delayed due to the resolution of potential aircraft conflicts. These nodes are *critical* when they are on the longest path from the start node s to the end node t in $\mathcal{G}(F, S)$, that is called the *critical path set* $\mathcal{C}(F, S)$. Given a solution $S(F)$, $krp \in N(F) \setminus \{s, t\}$ is a *critical node* of aircraft k with route r if $l^{S(F)}(s, krp) + l^{S(F)}(krp, t) = l^{S(F)}(s, t)$. A critical node krp is a *waiting node* if $l^{S(F)}(s, krp) > l^{S(F)}(s, \nu(krp)) + w^F_{\nu(krp),krp}$, where the node $\nu(krp)$ precedes the node

krp on route r. For each waiting node krp, there is at least one *hindering node* $\eta(krp)$ in $\mathcal{G}(F,S)$, different from node $\nu(krp)$, such that $l^{S(F)}(s,krp) = l^{S(F)}(s,\eta(krp)) + w^{F}_{\eta(krp),krp}$.

Given a node $krp \in N(F) \setminus \{s,t\}$, we recursively define the *backward ramification* $R_B(krp)$ as follows. If node krp is waiting, then $R_B(krp) = R_B(\nu(krp)) \cup R_B(\eta(krp) \cup \{krp\}$, otherwise $R_B(krp) = R_B(\nu(krp)) \cup \{krp\}$. Similarly, we recursively define the *forward ramification* $R_F(krp)$ as follows. If node krp is the hindering of a waiting node dij, then $R_F(krp) = R_F(\sigma(krp)) \cup R_F(dij) \cup \{krp\}$, where the node $\sigma(krp)$ follows the node krp on route r. Otherwise, $R_F(krp) = R_F(\sigma(krp)) \cup \{krp\}$. By definition, $R_B(s) = R_F(s) = \{s\}$ and $R_B(t) = R_F(t) = \{t\}$. Given $\mathcal{C}(F,S)$, we define a *ramified critical path* set as $\mathcal{F}(F,S) = \bigcup_{krp \in \mathcal{C}(F,S)}[R_B(krp) \cup R_F(krp)]$. We study the five neighbourhood structures listed below.

- *Complete K-Route neighbourhood* $\mathcal{N}_{CKR}$: contains all the feasible solutions to the ATC-TCA problem in which K aircraft follows a different route compared to the incumbent solution. To limit the number of neighbours to be evaluated, $\mathcal{N}_{CKR}$ is only partially explored as follows. A move is obtained by choosing K routes different from the ones of the current solution at random (i.e., all alternative routes having the same probability), until a number ψ (parameter) of alternative routing solutions is obtained:

- *Ramified Critical Path Operations neighbourhood* $\mathcal{N}_{RCPO}$ considers only the routing alternatives for the aircraft associated to the nodes in $\mathcal{B}(F,S)$ plus $\mathcal{F}(F,S)$. The idea is that the maximum consecutive delay of an optimal solution to the ATC-TCA problem can be reduced by removing aircraft conflicts causing it. This requires either removing, anticipating or postponing some operations from the critical path set (i.e., re-routing the associated aircraft through different resources). The latter result can be obtained by re-routing some aircraft represented by jobs with nodes in $\mathcal{B}(F,S)$ or $\mathcal{F}(F,S)$ and then rescheduling aircraft movements;

- *Waiting Operations Critical Path neighbourhood* $\mathcal{N}_{WOCP}$ is a restriction of $\mathcal{N}_{RCPO}$ that considers the routing alternatives for the aircraft associated to the waiting nodes in $\mathcal{C}(F,S)$;

- *Delayed Jobs neighbourhood* $\mathcal{N}_{DJ}$ considers only the aircraft (jobs) that have a consecutive delay on some due date arcs on the incumbent solution;

- *Free-Net Waiting Operations Jobs neighbourhood* $\mathcal{N}_{FNWJ}$ considers only the aircraft (jobs) that have some waiting nodes in the graph of the incumbent solution in which all alternative arcs are unselected (i.e. free-net traffic situation).

The appendix will illustrate some neighbourhood structures for an illustrative example.

3.4 Heuristic evaluation of routing neighbours

The choice of a best neighbour in the neighbourhood required the computation a new ATC-TCA solution $S'(F')$ starting for an incumbent solution $S(F)$, that is characterized by the routing decisions in F' and the sequencing decisions in S'. To this aim, we use fast heuristics based on a two-step graph building procedure in which the graph $\mathcal{G}(F,S)$ is translated into the graph $\mathcal{G}'(F',S')$. In the first step, a sub-graph of $\mathcal{G}'(F',S')$ is

generated by considering all the nodes $\in N(F^I)$ associated to the routes modelled by the arcs $\in F^I = F \bigcap F'$, all the fixed arcs $\in F^I$ and all the alternative arcs in $S(F)$ incident in a node $\in N(F^I)$. This corresponds to keeping a subset of decisions from the incumbent solution into the neighbour solution. In the second step, the fixed arcs $\in F^R = F' \setminus F^I$ and the nodes $\in N(F^R)$ are added to the sub-graph. Finally, $\mathcal{G}'(F', S')$ is obtained by adding a selection of alternative arcs $S'(F^R)$ to the sub-graph.

The selection $S'(F^R)$ is computed by selecting the best solution among two greedy algorithms based on the idea of repeatedly enlarging a selection by choosing an unselected pair at a time from set A and by selecting one of the two arcs until a feasible schedule is found or an infeasibility (i.e., positive weight cycle in the graph) is detected [13]. The first greedy algorithm AMSP (*Avoid Most Similar Pair*) chooses an unselected alternative pair $((krp, dij), (uml, vnw)) \in A$ maximizing the quantity $l^{S'(F^R)}(s, krp) + w^A_{krp,dij} + l^{S'(F^R)}(dij, t) + l^{S'(F^R)}(s, uml) + w^A_{uml,vnw} + l^{S'(F^R)}(vnw, t)$. The other greedy algorithm AMCC (*Avoid Most Critical Completion Time*) chooses the alternative pair $((krp, dij), (uml, vnw)) \in A$ such that the quantity $l^{S'(F^R)}(s, krp) + w^A_{krp,dij} + l^{S'(F^R)}(dij, t)$ is maximum among all the unselected alternative arcs, Both algorithms select the arc of the pair causing the minimum consecutive delay.

3.5 Tabu search re-routing algorithm

The *Tabu Search* (TS) is a deterministic metaheuristic based on local search, which makes extensive use of memory for guiding the search [16]. A basic ingredient is the *tabu list*, that is used to avoid being trapped in local optima and revisiting the same solution. From the incumbent solution, non-tabu moves define a set of solutions, named the *incumbent solution neighbourhood*. At each step, the best solution in this set is chosen as the new incumbent solution. Some attributes of the former incumbent are then stored in the tabu list. The moves in the tabu list are forbidden as long as these are in the list, unless an aspiration criterion is satisfied. The tabu list length can remain constant or be dynamically modified during the search.

The algorithm used in this paper for the iterative scheduling and re-routing framework is the Tabu Search (TS) of D'Ariano et al. [15]. The neighbourhood strategy used by TS explores candidate solutions in $\mathcal{N}_{RCPO}$ unless this neighbourhood is empty. In the latter case, ψ consecutive moves are performed in $\mathcal{N}_{CKR}$ with $K = 1$ before searching again in $\mathcal{N}_{RCPO}$. All neighbours are evaluated via the scheduling heuristics of Section 3.4. The best neighbour is set as the move to be made, and evaluated via the branch-and-bound of Section 3.2; the resulting best solution is set as the new incumbent solution. The inverse of the chosen move is stored in a tabu list of length λ (parameter). The moves in the tabu list are forbidden for λ iterations and no aspiration criteria is used. When no potentially better solution is found on the incumbent solution neighbourhood, the search alternates the above neighbourhood strategy with a diversification strategy, which consists of changing at random the route of μ (parameter) aircraft at the same time. From the tuning performed in [15], the best overall exploration strategy has the following parameters $\psi = 10$, $\lambda = 32$ and $\mu = 5$.

3.6 Variable neighbourhood search re-routing algorithm

A Variable Neighbourhood Search (VNS) metaheuristic is proposed in this subsection in order to efficiently solve the ATC-TCA problem. This metaheuristic is based on the combination of different neighbourhoods. Neighbourhood changes are proposed both in a local search phase in order to compute a local minimum, and in a perturbation phase in order to escape from a local minimum [19]. The choice of this search method is motivated by the following facts: (i) the scheduling solution with fixed routes can be improved in terms of multiple routing modifications, (ii) there is a need of improving upon the local minima found by local search. For both reasons, there is a need to develop intensification and diversification phases that make use of neighbourhoods of different size and/or type, differing in the set of candidate aircraft that are re-routed in each move and in the set of routing alternatives.

The VNS algorithm of Figure 2 is an adaptation of the basic VNS described in Hansen et al. [19]. This algorithm combines the classic ingredients of the VNS algorithm with the routing neighbourhood structures of Section 3.3 and new sophisticated neighbourhood search strategies to search for better aircraft routes.

The general structure of the VNS is the following. The algorithm starts from an incumbent solution of the ATC-TCA problem, named $IncSol$ $\mathcal{G}(F, S)$, computed via the branch-and-bound scheduling algorithm of Section 3.2 given a default (off-line) route to each aircraft. A counter K is adopted to fix the number of aircraft that are re-routed in each move. The initial value of K is set to 1, i.e. a single aircraft is re-routed in $IncSol$ $\mathcal{G}(F, S)$. The metaheuristic iterates the search for better solutions starting from $IncSol$ until a maximum computation time T_{max} is reached or until the maximum consecutive delay is larger than 0. At each iteration, a neighbourhood of $IncSol$ is generated and a new solution $IncSol'$ is selected via a shaking procedure. The search continues with the generation of a neighbourhood of $IncSol'$ and a local search based on best-improvement is performed in a restricted neighbourhood. Then, a *Move Or Not* function is performed as follows. In case an improving move $IncSol''$ is obtained via the local search (i.e. $f(IncSol'') < f(IncSol)$), a new iteration is performed by setting $IncSol''$ as the new incumbent solution and K is set to 1. Otherwise, the parameter K is set to $K + 1$ and a new iteration is performed until $K \leq K_{max}$. When $K = K_{max}$ the algorithm diversifies the search with a change of neighbourhood structure (if the algorithm works with a single neighbourhood structure this step is not performed). The metaheuristic returns the best ATC-TCA solution ($IncSol$) and the objective function value ($f(IncSol)$). The pseudo-code of the VNS is reported in Figure 2. We next describe the specific features of the VNS.

Build Neighbourhood. Starting from an incumbent solution, the $\mathcal{N}_{CKR}$ neighbourhood is generated, in which exactly K aircraft are re-routed in the graph $\mathcal{G}(F, S)$ of the incumbent solution.

Shake. This is a typical diversification procedure in metaheuristics that consists in changing the route of K aircraft randomly in the $\mathcal{N}_{CKR}$ neighbourhood of the incumbent solution ($IncSol$), and in computing a new incumbent solution ($IncSol'$) via the scheduling heuristics of Section 3.4 and the new set of routes.

Neighbourhood Search Strategy. This procedure is proposed in order to reduce the local search to the evaluation of up to L neighbours in the current neighbourhood.

Algorithm VNS
Input: $IncSol\ \mathcal{G}(F,S)$, K_{max}, T_{max}, L, $\mathcal{N}_1$, $\mathcal{N}_2$
$\mathcal{N}_i \leftarrow \mathcal{N}_1$,
While $(T < T_{max})$ & $(f(IncSol) > 0)$ **do**
  Begin
  $K \leftarrow 1$,
  While $(K \leq K_{max})$ **do**
    Begin
    $BuildNeighbourhood(IncSol, K)$,
    $IncSol' \leftarrow Shake(IncSol, K)$,
    $BuildNeighbourhood(IncSol', K)$,
    $NeighbourhoodSearchStrategy(IncSol', K, L, \mathcal{N}_i)$,
    $IncSol'' \leftarrow FirstImprovement(IncSol')$,
    $(IncSol, K) \leftarrow MoveOrNot(IncSol, IncSol'', K)$,
    If $(K = K_{max})$ **do**
      Begin
      $\mathcal{N}_i \leftarrow NeighbourhoodChange(IncSol, \mathcal{N}_i, \mathcal{N}_1, \mathcal{N}_2)$,
      End
    $T \leftarrow$ CPU time()
    End
  End

Figure 2: Sketch of the VNS algorithm

Starting from an incumbent solution and the $\mathcal{N}_{CKR}$ neighbourhood of this solution, a restricted neighbourhood is generated by using a given neighbourhood structure $\mathcal{N}_i$. The selection of L neighbours is achieved in the following steps:

1. *aircraft ranking* : Each aircraft gets a score based on the criterion specified in a neighbourhood structure $\mathcal{N}$. The ranking is based on one of the neighbourhood structures of Section 3.3. In $\mathcal{N}_{RCPO}$, each aircraft on the ramified critical path gets a score based on the maximum value $l^{S'(F^R)}(s, krp) + l^{S'(F^R)}(krp, t)\ \forall\ (krp)$ in the ramified critical path of the graph of the incumbent solution. In $\mathcal{N}_{WOCP}$, each aircraft gets a score based on the sum of the consecutive delays collected at each critical node in the graph of the incumbent solution. In $\mathcal{N}_{DJ}$, each aircraft gets a score based on the maximum consecutive delay collected on the due date arcs for each job. In $\mathcal{N}_{FNWJ}$, each aircraft gets a score based on the sum of the consecutive delays collected at each waiting node in the graph of the incumbent routing solution in which all alternative arcs are unselected (i.e. free-net traffic situation) but the one generating the waiting node. The scores are used to decide how many times each aircraft has to be re-routed in the L neighbours;

2. *route ranking* : The routes of each aircraft get a score based on the distance from the route of the incumbent solution. The larger is the difference between the routes, the higher is the score. The route ranking thus suggests for each aircraft to select the most different routes. Among the selected routes, the ranking gives precedence to the routing alternatives in which there is a change of runway, since this is often the conflicting resource of the TCA aircraft routes;

3. *neighbour generation* : This is the assignment of the routes to the aircraft in each neighbour. This is done by selecting the aircraft based on the aircraft ranking and by selecting the routes based on the route ranking. A combinatorial combination of the routes is used in order to generate L different neighbours. In each neighbour, exactly K aircraft are re-routed compared to the incumbent solution.

Figure 3 presents a numerical example of the neighbourhood search strategy, in which four aircraft ($J = \{J1, J2, J3, J4\}$) can be re-routed in a TCA. More details are reported in the paper appendix. J1 and J4 have four alternative routes (e.g. the routes of $J1$ are: $J1$-1, $J1$-2, $J1$-3, $J1$-4), while J2 and J3 have two alternative routes each. In the incumbent solution, all aircraft use the first route ($J1$-1, $J2$-1, $J3$-1, $J4$-1). The parameters of the procedure are set to the following values: $L = 4$ and $K = 2$ (i.e. the neighbourhood is restricted to 4 neighbours and 2 aircraft are re-routed in each neighbour).

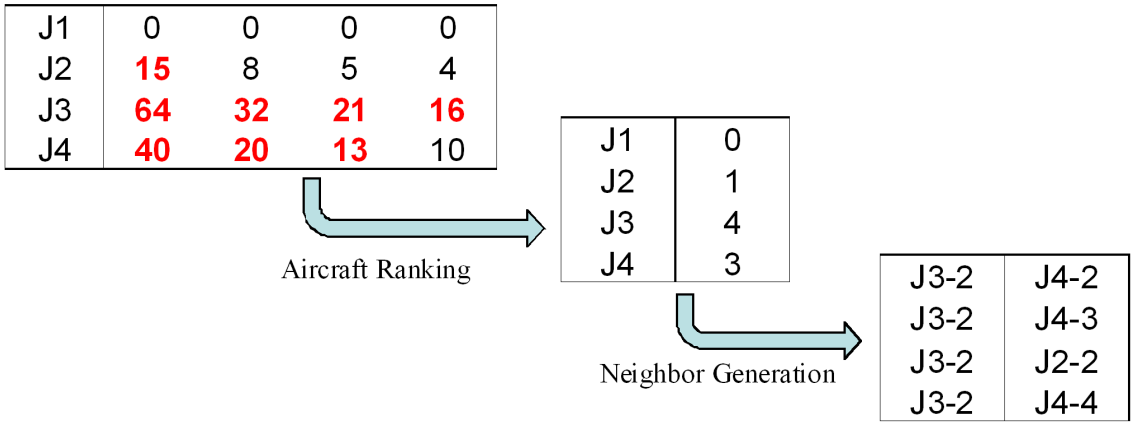


Figure 3: Example of neighbourhood search strategy with $|J| = 4$, $L = 4$ and $K = 2$

The aircraft ranking determines a score matrix in which each row represents an aircraft and each column the number of times each aircraft can be re-routed. This score matrix is depicted in the left-hand side of Figure 3. Specifically, the first column reports the score for each aircraft based on the neighbourhood structure $\mathcal{N}_{DJ}$ (e.g. the value 40 in this example is the maximum consecutive delay collected by aircraft $J4$, see Appendix). The other columns report the score of the first column divided by the number of column, e.g. $40/2 = 20$, $40/3 = 13$, $40/4 = 10$. By taking the highest KL scores in the score matrix (these values are reported in bold in Figure 3), we get the number of times each aircraft has to be re-routed in the four neighbours (e.g. $J4$ is re-routed in three neighbours). This is reported in the center table of Figure 3.

The route ranking orders the list of re-routing alternative of each aircraft based on maximizing the difference with the incumbent route and giving precedence to the routing alternatives in which there is a change of runway. In this case, the route ranking of $J4$ is $J4$-2, $J4$-3, $J4$-4 and the most different route with a change of runway is $J4$-2.

The neighbour generation procedure assigns the routes to the aircraft in each neighbour. In this example, $J4$ appears in the neighbours with its three different routes; $J2$ and $J3$ have only a route to be chosen. The neighbours are ordered based on the aircraft ranking, and in case of tie on the route ranking. Every row of the table in right-hand side of Figure 3 corresponds to a candidate move.

First Improvement. This is a local search procedure in the restricted neighbourhood in which the candidate moves are ordered based on the neighbour generation procedure. At each step of the procedure, the neighbour with the highest ranking in the restricted neighbourhood is considered, a new graph is build with the new routes of the neighbour, and a new solution is computed via the scheduling heuristics of Section 3.4 for the new set of routes. This procedure lasts until a better solution is obtained compared to the incumbent solution or until all L neighbours in the restricted neighbourhood have been evaluated.

Move Or Not. This procedure is responsible for possibly making a move. In case the best solution found in the neighbourhood is better than the incumbent, the resulting graph is solved by the branch-and-bound algorithm of Section 3.2, and the best solution is set as the new incumbent solution. Otherwise, the best solution in the neighbourhood is chosen as incumbent, or some diversification strategy is employed.

Neighbourhood Change. This procedure is used to diversify the search by alternating K iterations of the neighbourhood search strategy with $\mathcal{N}_1$ and K iterations of the neighbourhood search strategy with $\mathcal{N}_2$.

3.7 Hybrid search re-routing algorithm

We introduce a hybrid metaheuristic, named Variable Neighbourhood Tabu Search (VNTS), consisting of a combination of VNS and TS, as proposed earlier in other contexts by Moreno Pérez et al. [25]. The hybridization is proposed in order to take promising ideas from both the metaheuristics. The VNS is used in order to explore different neighbourhoods of the incumbent solution, while the TS is used to avoid cycling and is combined with strategies to escape from local minimum. Specifically, the ingredients of the proposed hybrid metaheuristic are an intensification strategy based on the exact exploration of a restricted neighbourhood, and various diversification strategies based on the evaluation of different neighbourhood structures and the implementation of a restart technique based on a problem-specific memory structure.

Figure 4 introduces the pseudo-code of the hybrid metaheuristic. Starting from an incumbent solution $IncSol$ $\mathcal{G}(F, S)$, a given neighbourhood structure $\mathcal{N}_i$ and a counter Q (initialized as 0), the metaheuristic iterates the search for better solutions in various neighbourhoods until a maximum computation time T_{max} is reached, as far as the maximum consecutive delay is larger than 0. Each iteration evaluates all the neighbours in a restricted neighbourhood of $IncSol$ (i.e. L non-tabu neighbours, determined via a neighbourhood search strategy, analogously to the VNS procedure) and the best neighbour is implemented via the branch-and-bound scheduling algorithm of Section 3.2. The *Move Or Not* function is performed as for the VNS algorithm of Figure 2. However, when $K = K_{max}$ the algorithm diversifies the search as follows. If $Q < Q_{max}$ a change of neighbourhood structure is implemented, and Q is set to $Q + 1$; otherwise a restart strategy is applied and Q is set to 0. Before a new iteration is performed, a tabu search memory and a time counter are updated. We next describe key VNTS features in more detail.

Best Improvement. Given a restricted neighbourhood of an incumbent solution, this procedure evaluates all neighbours via the scheduling heuristics of Section 3.4. The best neighbour is set as the move to be made, and evaluated via the branch-and-bound of Section 3.2; the resulting best solution is set as the new incumbent solution.

Neighbourhood Change. This procedure is used to diversify the search by alter-

Algorithm VNTS
Input: $IncSol\ \mathcal{G}(F,S)$, K_{max}, T_{max}, L, TL_{max}, Q_{max}, $\mathcal{N}_1$, $\mathcal{N}_2$
$Q \leftarrow 0$,
$\mathcal{N}_i \leftarrow \mathcal{N}_1$,
While $(T < T_{max})$ & $(f(IncSol) > 0)$ **do**
Begin
$K \leftarrow 1$,
While $(K \leq K_{max})$ **do**
Begin
$BuildNeighbourhood(IncSol, K)$,
$TabuMemoryFilter(TL)$,
$NeighbourhoodSearchStrategy(IncSol, K, L, \mathcal{N}_i)$,
$IncSol' \leftarrow BestImprovement(IncSol)$,
$(IncSol, K) \leftarrow MoveOrNot(IncSol, IncSol', K)$,
If $(K = K_{max})$ **do**
Begin
If $(Q < Q_{max})$ **do**
Begin
$\mathcal{N}_i \leftarrow NeighbourhoodChange(IncSol, \mathcal{N}_i, \mathcal{N}_1, \mathcal{N}_2)$,
$Q \leftarrow Q + 1$,
End
Else
Begin
$RestartStrategy(Q)$,
$Q \leftarrow 0$,
End
End
$TL \leftarrow TabuMemoryUpdate(TL, TL_{max}, Q, Q_{max})$,
$T \leftarrow$ CPU time()
End
End

Figure 4: Sketch of the VNTS algorithm

nating K iterations of the neighbourhood search strategy with $\mathcal{N}_1$ with K iterations of the neighbourhood search strategy with $\mathcal{N}_2$. In particular, when $Q < Q_{max}$, $K = K_{max}$ and the current neighbourhood structure $\mathcal{N}_i = \mathcal{N}_1$, the procedure sets $\mathcal{N}_i = \mathcal{N}_2$ and the search continues with K neighbourhood search strategy iterations with $\mathcal{N}_2$.

Restart Strategy. This is a typical diversification strategy that we implemented as follows. A restart memory structure is used to store the number of times each aircraft routing has been evaluated in any neighbour so far. From this structure, we compute an ordered list of routing alternatives for each aircraft from the less frequently used to the most frequently used. The restart strategy consists on the implementation of a move (disregarding the tabu memory) in which the less frequently used route is set for each aircraft. In case of tie, a random decision is taken for each aircraft among its routes being used the least. The restart memory structure is reset each time the restart strategy is used.

Tabu Memory. This technique is used to avoid cycling and revisiting the same solution. This is achieved by two functions named *tabu memory filter* and *tabu memory update*. The former function removes from the current neighbourhood all the tabu moves (no aspiration criteria is used), while the latter function updates the list of tabu moves during the search. The tabu list (TL) used in the VNTS is made by up to TL_{max} moves. For each move that has been implemented in a local search step, the following key information is stored: the re-routed aircraft of the move, and the old route chosen for each re-routed aircraft. For example, we have a problem with three aircraft and three routes for each aircraft. An incumbent solution is $J1-1, J2-1, J3-1$ and the move performed by a local search is $J1-1, J2-2, J2-3$. The tabu memory update will store in the tabu list the following information: $J2$ and $J3$ have been re-routed and the tabu-routes are $J2-1$ and $J3-1$. A move is tabu when the two routes $J2-1$ and $J3-1$ are chosen as the new routes of $J2$ and $J3$. The tabu list is reset when a restart move is performed (i.e. when $Q = Q_{max}$).

4 Computational experiments

This section presents the experimental assessment of the various metaheuristics of Section 3. The test bed is the Milan Malpensa terminal control area (MXP) and the instances are taken from Samà et al. [33]. In particular, we evaluate the metaheuristics on the most complex instances with strong disturbances and a large number of aircraft. The objective of this evaluation is to report the marginal improvement achieved by the new metaheuristics compared with the approaches in [33]. To this end, the experiments are executed on the same processor used in [33], that is Intel Core 2 Duo E6550 (2.33 GHz), 2 GB of RAM, Windows XP. For all the metaheuristics tested in this paper, we adopt the deadline implications and the pre-processing procedure developed in [33]. For all the approaches based on the rolling horizon framework, we use the same parameter setting in [33] (i.e. we fix the roll period to 10 min and the look-ahead period to 15 min).

4.1 Description of the TCA

Figure 5 shows the TCA scheme of Malpensa (MXP) TCA. There are two runways (RWY 35L, RWY 35R), used both for departing and arriving procedures. The MXP resources

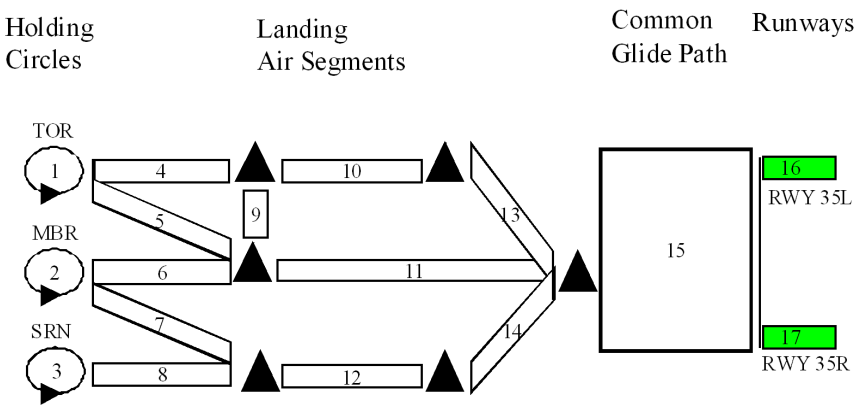


Figure 5: Malpensa (MXP) Terminal Control Area

are: three airborne holding circles (resources 1-3 in Figure 5, named TOR [Torino], MBR [Mebur], SRN [Saronno]), eleven air segments for arriving procedures (resources 4-14), a common glide path (resource 15) and two runways (resources 16-17). The common glide path resource includes two parallel air segments before the runways for which traffic regulations impose a minimum diagonal/longitudinal distance between landing aircraft.

4.2 Tested instances

The ATC-TCA instances are based on real data collected for the Milan Malpensa TCA. We consider a subset of the instances generated in [33], that are obtained by varying the following parameters: (i) the model variant, (ii) the time horizon of traffic prediction, (iii) the aircraft delays, (iv) the disruption.

Model variant. Three variants are investigated to model objective functions and user requirements. Model 1 (M1) measures the delay of landing and take-off aircraft at the runways, that are the most used TCA resources. Model 2 (M2) modifies the objective function for landing aircraft by measuring their delay both at the runways and at the entrance of the TCA, penalizing a late entrance in the TCA. The latter model takes into consideration the extra work required to coordinate the solutions of TCA and en-route traffic controllers. Model 3 (M3) extends M2 with additional deadline constraints that limit the maximum possible entrance time of each landing aircraft in the TCA. These constraints are inserted in order to consider the limited possibility of airborne holding, being more expensive and constrained than ground holding.

Time horizon of traffic prediction. Two time horizons of different length are considered: 60 and 180 minutes. The shorter time horizon can be considered of practical interest for the management of light disturbances, while the longer time horizon can better assess the traffic control measures in terms of aircraft delay propagation. The latter time horizon is particularly relevant to study in case of disruptions.

Aircraft delays. Disturbed traffic conditions are generated by delaying the entrance time of some aircraft in the TCA. The entrance delays are randomly generated according to a uniform distribution, and are applied at some aircraft entering the TCA during the first half of the time horizon under examination. For each time horizon of traffic prediction, we consider 10 delay instances in which random delays are up to 5 minutes

and other 10 delay instances in which random delays are up to 15 minutes.

Disruption. We consider the case in which one of the two runways of the TCA is unavailable in a time window, and all landing and take-off aircraft have to be scheduled in the only available runway during the disruption. The disrupted case is only studied for the 180-minute traffic predictions with a disrupted runway between the first and the second hour of traffic prediction.

In total, there are 3 groups of instances: two groups regard undisrupted operations with different time horizons (60 and 180 minutes), for a total of 120 delayed instances (i.e. for the 3 model variants, the 2 time horizons, and the 20 aircraft delays); one group of 60 disrupted instances (i.e. the 3 model variants, the 180-minute time horizon, and the 20 aircraft delays).

Table 1 gives average information on the aircraft delay instances regarding the MILP formulation; every row is an average over the 20 delay instances. Moreover, Column 1 reports the time horizon of traffic prediction, Column 2 the model variant, Column 3 the fixed constraints, Column 4 the alternative constraints, Column 5–7 the MILP variables, Column 8 the number of aircraft. The model variants differ in terms of the fixed constraints, since different date and deadline arcs are used. Instead, disrupted instances have the same characteristics (concerning the variables in Table 1) than the corresponding undisrupted instances.

Table 1: Size of the MILP formulation for the various ATC-TCA instances

Time Horizon	Model Variant	Num of Fixed Constraints	Num of Alternative Constraints	MILP Variables			Num of Aircraft
				h	x	y	
60 min	M1	728	11526	264	5763	62	40
	M2	751	11526	264	5763	62	40
	M3	774	11526	264	5763	62	40
180 min	M1	4331	456476	871	228238	414	117
	M2	4649	456476	871	228238	414	117
	M3	4967	456476	871	228238	414	117

4.3 Assessment of VNS and VNTS parameters

This section discusses the choice of alternative configurations for the VNS and VNTS algorithms, while the TS algorithm configurations were evaluated in previous works [15, 33]. The computational assessment is based on 20 pilot ATC-TCA instances with aircraft delays. Disrupted instances have not been considered since these present a reduced number of alternative routes. The assessment is based on the evaluation of the metaheuristics in the centralized framework with $T_{max} = 180$ seconds.

Table 2 shows the parameters considered in the algorithmic configurations assessment. The parameters used in both algorithms are the time given to the branch-and-bound scheduling algorithm (BB Time, in seconds), the number of aircraft that are rerouted in the current neighbourhood (K_{max}), the size of the restricted neighbourhood (L), and the choice of the neighbourhood structure ($\mathcal{N}_1 + \mathcal{N}_2$ or $\mathcal{N}$ when the two coincide). The VNTS algorithm has the following additional parameters: the tabu list length (TL_{max}), the counter used for the diversification strategies (Q_{max}). The best value of each parameter is reported in bold and used for the computational experiments.

Table 2: Experimental setting of algorithmic parameters

T_{max} (sec)	BB Time (sec)	K_{max}	L	TL_{max}	Q_{max}
180	4/**10**/20/30	2/**3**/4/5	5/**10**/20	0/8/15/**32**	2/**3**/4

$\mathcal{N}$	$\mathcal{N}_1 + \mathcal{N}_2$
DJ / RCPO / WOCP / FNWJ	**WOCP+FNWJ** / FNWJ+WOCP / DJ+WOCP / WOCP+DJ

4.4 Assessment of solution quality

This section presents the results obtained for the best metaheuristics developed in this paper and compares them with the results obtained in Samà et al. [33], that is used as a benchmark comparison for the newly developed algorithms. Table 3 gives an overview of the best ALGOrithm in [33] (ALGO, Column 4), the best CEntralized MetaHeuristic (CE MH, Column 5) and the best Rolling Horizon MetaHeuristic (RH MH, Column 6) for the instances grouped by the time horizon of traffic prediction (Column 1), the type of disturbance (Column 2), the model variant (Column 3). Each row reports the best algorithm on the set of 20 ATC-TCA instances described in Section 4.2.

We now recall the different acronyms used: MILP refers to the Mixed-Integer Linear Programming formulation of the ATC-TCA problem solved by using the IBM ILOG CPLEX MIP 12.0 solver; VNS, VNTS and TS refer to the metaheuristics Variable Neighbourhood Search, Variable Neighbourhood Tabu Search and Tabu Search; DJ and WOCP+FNWJ refer to the restricted neighbourhood Delayed Job and the combined restricted neighbourhoods Waiting Operation Critical Path and Free-Net Waiting Operations. We next give detailed information on the performance of the various best algorithms reported in Table 3.

Table 3: Best algorithms for various groups of instances

Time Horizon	Disturbance Type	Model Variant	Best Algo [33]	Best CE MH	Best RH MH
60 min	Normal	M1	RH MILP	VNTS DJ	VNS DJ
		M2	CE MILP	VNTS DJ	VNS DJ
		M3	CE MILP	VNTS DJ	VNS DJ
180 min	Normal	M1	RH MILP	VNTS DJ	VNS DJ
		M2	RH MILP	VNTS DJ	VNS DJ
		M3	RH MILP	VNTS DJ	VNS DJ
180 min	Disrupted	M1	RH TS	VNS WOCP+FNWJ	VNS DJ
		M2	RH TS	VNS WOCP+FNWJ	VNS DJ
		M3	RH TS	VNS WOCP+FNWJ	VNS DJ

Regarding CE MILP, RH MILP and RH TS, we use the same setting of the parameters and thus the same results presented in Samà et al. [33]. We recall that their time limit of computation is 240 (720) seconds for the 60-minute (180-minute) instances.

Regarding the metaheuristics developed in this paper, the parameters are set as described in Section 4.3. The best metaheuristics have been taken from among VNS WOCP+FNWJ, VNS DJ, VNTS WOCP+FNWJ, VNTS DJ. We recall that the time limit of computation is 180 seconds for the CE MH. The RH MH have a time limit of 30 seconds (10 seconds) for each roll period of the 60-minute (180-minute) instances, since 6 (18) periods are required to solve the overall time horizon and the maximum computation

time is also fixed equal to 180 seconds. The BB time limit for the CE MH (RH MH) is 10 seconds (4 seconds).

Tables 4, 5 and 6 provide the results obtained for the three groups of instances: the 60-minute delayed instances, the 180-minute delayed instances and the 180-minute disrupted instances. For each table, Column 1 presents a four-field code in order to identify each instance, where each code is structured as follows: [model variant]-[time horizon]-[average entrance delay]-[NOrmal or DISrupted traffic]. Columns 2-3 report the objective function value (in seconds) of the best known solution obtained in [33] for each instance, and the computation time (in seconds) at which this solution has been found. Columns 4-5 report the objective function value of the best solution found for each instance by the best metaheuristics developed in this paper combined with the centralized and rolling horizon frameworks, and the computation time at which this solution has been found. Columns 6-7 (8-9) (10-11) report the objective function value of the best solution and the time to compute it for the best algorithm in [33] (for the best CEntralized MetaHeuristic) (for the best Rolling Horizon MetaHeuristic).

For each group of instances in Tables 4, 5 and 6, the average values are also given in terms of the objective function and the time to compute the best solution. The best average values are reported in bold regarding the best solution columns and the best algorithm columns. We note that the best algorithms are reported in Table 3 and are identified as the algorithms with the best average performance for each group of instances. The objective function values of the solutions obtained for the best algorithms (Columns 6, 8, 10) are not necessarily equal to the objective function values of best known solutions (Columns 2, 4), since different algorithms may have found the best solution for different instances.

In Table 4, the best solution found by the new metaheuristics (Column 4) is always equal to the best solution found the algorithms in [33] (Column 2), and all these solutions are proven optimal. When comparing the algorithms, the best CE MH computes the optimal solution for all instances but M1-60-165.8-NO. The latter instance is solved to the proven optimum by the best MH RH. The best CE MH is the best algorithm in terms of the objective function value, even if it often requires a longer computation time that the best algorithm in [33]. The best RH MH is outperformed by the other best algorithms in terms of computation time, since the computation time of the rolling horizon framework is fixed to 180 seconds by construction.

In Table 5, the best CE MH and RH MH are, on average, significantly faster to compute their best solution compared to the best algorithm in [33] (i.e. RH MILP). Regarding the solution quality, the new metaheuristics compute equal or better quality solutions for several instances. In particular, the best RH MH is often better than the best CE MH. However, the best algorithm in [33] gives the best average results, in a longer computation time, except for M2.

In Table 6, the best CE MH, on average, outperforms the previously best known algorithm and the best RH MH in terms of both solution quality and time to compute the best solution. This strong improvement is due to the multiple simultaneous routing and scheduling changes performed by the new metaheuristics, that are often the actions required in case of disrupted traffic. The new best known solution is found partly by the best RH MH and partly by the best CE MH, but the latter algorithm is quicker and has the best average performance in terms of the objective function value.

Table 4: Results for the 60-minute delayed instances

Instance	Best Sol [33]		Best Sol MH		Best Algo [33]		Best CE MH		Best RH MH	
	Value	Time	Value	Time	Value	Time	Value	Time	Value	Time
	(sec)	(sec)	(sec)	(sec)	(sec)	(sec)	(sec)	(sec)	(sec)	(sec)
M1-60-34.3-NO	0	5.0	0	1.7	0	5.0	0	1.7	0	180
M1-60-6.1-NO	0	4.9	0	1.7	0	4.9	0	1.7	0	180
M1-60-41.8-NO	0	5.6	0	2.3	0	5.6	0	2.3	0	180
M1-60-17.6-NO	0	5.3	0	1.6	0	5.3	0	1.6	0	180
M1-60-46.8-NO	0	5.5	0	1.6	0	5.5	0	1.6	0	180
M1-60-10.5-NO	0	4.9	0	1.6	0	4.9	0	1.6	0	180
M1-60-50.2-NO	0	5.5	0	1.6	0	5.5	0	1.6	0	180
M1-60-11.8-NO	0	4.7	0	1.6	0	4.7	0	1.6	0	180
M1-60-28.7-NO	0	5.4	0	2.0	0	5.4	0	2.0	0	180
M1-60-24.3-NO	27	7.5	27	1.6	27	7.5	27	1.6	27	180
M1-60-165.8-NO	6	8.3	6	180	6	8.3	11	21.4	6	180
M1-60-75.4-NO	0	28.1	0	85.3	0	28.1	4	85.3	34	180
M1-60-151.7-NO	12	9.6	12	129.6	12	9.6	12	129.6	12	180
M1-60-61.9-NO	0	5.0	0	2.3	0	5.0	0	2.3	0	180
M1-60-115.4-NO	0	5.8	0	55.0	0	5.8	0	55.0	0	180
M1-60-86.7-NO	0	5.8	0	16.5	0	5.8	0	16.5	0	180
M1-60-80.0-NO	35	8.3	35	12.7	35	8.3	35	12.7	35	180
M1-60-61.6-NO	0	5.1	0	16.3	0	5.1	0	16.3	0	180
M1-60-134.7-NO	0	5.6	0	11.1	0	5.6	0	11.1	0	180
M1-60-56.1-NO	0	5.6	0	14.5	0	5.6	0	14.5	0	180
Avg Value M1	**4.0**	7.1	**4.0**	27.0	**4.0**	7.1	4.4	19.1	5.7	180
M2-60-34.3-NO	0	10.8	0	11.5	0	10.8	0	11.5	0	180
M2-60-6.1-NO	0	8.7	0	11.5	0	8.7	0	11.5	0	180
M2-60-41.8-NO	21	26.6	21	12.1	21	26.6	21	12.1	21	180
M2-60-17.6-NO	0	36.3	0	11.5	0	36.3	0	11.5	8	180
M2-60-46.8-NO	0	7.0	0	11.1	0	7.0	0	11.1	0	180
M2-60-10.5-NO	0	15.3	0	23.6	0	15.3	0	23.6	0	180
M2-60-50.2-NO	0	9.8	0	69.7	0	9.8	0	69.7	0	180
M2-60-11.8-NO	0	28.1	0	11.5	0	28.1	0	11.5	0	180
M2-60-28.7-NO	0	11.1	0	1.7	0	11.1	0	1.7	0	180
M2-60-24.3-NO	27	37.6	27	1.6	27	37.6	27	1.6	27	180
M2-60-165.8-NO	20	240.0	20	45.6	20	240	20	45.6	20	180
M2-60-75.4-NO	34	240	34	114.9	35	240	34	114.9	40	180
M2-60-151.7-NO	12	97.5	12	164.0	12	97.5	12	164.0	12	180
M2-60-61.9-NO	0	6.1	0	2.8	0	6.1	0	2.8	0	180
M2-60-115.4-NO	0	19.8	0	99.8	0	19.8	0	99.8	7	180
M2-60-86.7-NO	0	14.3	0	2.7	0	14.3	0	2.7	0	180
M2-60-80.0-NO	35	25.3	35	33.5	35	25.3	35	33.5	35	180
M2-60-61.6-NO	0	6.7	0	26.9	0	6.7	0	26.9	0	180
M2-60-134.7-NO	0	42.3	0	15.5	0	42.3	0	15.5	0	180
M2-60-56.1-NO	0	13.8	0	13.8	0	13.8	0	13.8	0	180
Avg Value M2	**7.45**	44.9	**7.45**	34.3	7.5	44.9	**7.45**	34.3	8.5	180
M3-60-34.3-NO	0	6.6	0	11.6	0	6.6	0	11.6	0	180
M3-60-6.1-NO	0	11.6	0	11.6	0	11.6	0	11.6	0	180
M3-60-41.8-NO	21	6.5	21	12.2	21	6.5	21	12.2	21	180
M3-60-17.6-NO	0	7.0	0	11.6	0	7.0	0	11.6	0	180
M3-60-46.8-NO	0	5.2	0	11.1	0	5.2	0	11.1	0	180
M3-60-10.5-NO	0	6.0	0	23.7	0	6.0	0	23.7	0	180
M3-60-50.2-NO	0	5.8	0	71.0	0	5.8	0	71.0	0	180
M3-60-11.8-NO	0	7.1	0	11.5	0	7.1	0	11.5	0	180
M3-60-28.7-NO	0	7.2	0	1.7	0	7.2	0	1.7	0	180
M3-60-24.3-NO	27	12.0	27	1.7	27	12.0	27	1.7	27	180
M3-60-165.8-NO	20	240	20	45.9	46	1.0	20	45.9	20	180
M3-60-75.4-NO	34	59.8	34	115.7	34	59.8	34	115.7	40	180
M3-60-151.7-NO	12	11.1	12	166.5	12	11.1	12	166.5	12	180
M3-60-61.9-NO	0	6.1	0	2.8	0	6.1	0	2.8	0	180
M3-60-115.4-NO	0	7.8	0	162.7	0	7.8	0	162.7	7	180
M3-60-86.7-NO	0	11.0	0	2.7	0	11.0	0	2.7	0	180
M3-60-80.0-NO	35	8.8	35	33.7	35	8.8	35	33.7	35	180
M3-60-61.6-NO	0	6.4	0	27.1	0	6.4	0	27.1	0	180
M3-60-134.7-NO	0	6.4	0	15.7	0	6.4	0	15.7	0	180
M3-60-56.1-NO	0	10.7	0	13.9	0	10.7	0	13.9	0	180
Avg Value M3	**7.45**	22.1	**7.45**	37.7	8.7	10.1	**7.45**	37.7	8.1	180

Table 5: Results for the 180-minute delayed instances

Instance	Best Sol [33]		Best Sol MH		Best Algo [33]		Best CE MH		Best RH MH	
	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)
M1-180-97.9-NO	11	319.5	11	115.7	11	319.5	15	70.3	11	180
M1-180-78.7-NO	71	720	71	44.6	78	214.9	74	116.8	78	180
M1-180-100.1-NO	11	276.8	11	59.4	11	276.8	11	110.2	11	180
M1-180-85.7-NO	11	192.2	11	180	11	192.2	60	16.1	11	180
M1-180-95.1-NO	14	356.6	11	53.8	14	356.6	11	80.1	11	180
M1-180-87.3-NO	11	212.8	11	180	11	212.8	60	5.0	18	180
M1-180-96.8-NO	62	227.7	62	155.2	62	227.7	71	86.3	71	180
M1-180-93.6-NO	15	224.7	17	44.6	15	224.7	17	160.7	52	180
M1-180-94.0-NO	11	267.3	40	180	11	267.3	61	17.1	61	180
M1-180-95.3-NO	27	720	27	25.2	149	295.6	27	38.5	27	180
M1-180-145.8-NO	36	463.6	38	180	36	463.6	71	180	80	180
M1-180-96.7-NO	11	293.4	11	180	11	293.4	40	90.6	35	180
M1-180-135.2-NO	35	77.4	35	180	35	423.6	66	107.7	71	180
M1-180-104.5-NO	77	720	69	83.1	81	219.0	69	180	69	180
M1-180-117.1-NO	11	180.9	11	180	11	180.9	15	85.5	15	180
M1-180-111.8-NO	15	286.3	11	180	15	226.3	26	82.0	11	180
M1-180-102.7-NO	19	346.5	11	172.0	19	346.5	11	172.0	11	180
M1-180-102.1-NO	18	268.2	11	180	18	268.2	60	33.9	66	180
M1-180-116.7-NO	11	207.9	11	114.8	11	207.9	11	114.8	11	180
M1-180-99.4-NO	14	255.9	11	180	14	255.9	53	33.2	11	180
Avg Value M1	**24.5**	330.9	**24.5**	133.4	**31.2**	273.7	41.4	89.0	36.5	180
M2-180-97.9-NO	31	276.3	11	78.5	31	276.3	15	79.8	11	180
M2-180-78.7-NO	71	342.0	71	79.1	71	342.0	71	79.1	71	180
M2-180-100.1-NO	47	720	21	180	54	384.3	21	112.5	21	180
M2-180-85.7-NO	31	287.6	47	180	31	287.6	47	78.2	61	180
M2-180-95.1-NO	11	276.8	12	180	11	276.8	42	53.0	12	180
M2-180-87.3-NO	18	378.2	47	180	18	378.2	47	34.8	47	180
M2-180-96.8-NO	62	306.2	62	180	62	306.2	64	139.4	62	180
M2-180-93.6-NO	47	377.1	47	180	47	377.1	49	177.5	47	180
M2-180-94.0-NO	43	339.3	43	34.8	43	339.3	43	34.8	59	180
M2-180-95.3-NO	27	399.5	27	89.0	27	399.5	27	89.0	65	180
M2-180-145.8-NO	47	472.1	47	180	47	472.1	73	130.7	71	180
M2-180-96.7-NO	45	720	30	180	84	367.8	40	138.2	30	180
M2-180-135.2-NO	60	34.5	36	180	77	456.0	60	149.7	36	180
M2-180-104.5-NO	69	358.8	69	48.8	69	358.8	69	48.8	71	180
M2-180-117.1-NO	15	333.5	15	110.7	15	333.5	15	110.7	15	180
M2-180-111.8-NO	29	10.0	29	180	42	279.0	29	104.2	29	180
M2-180-102.7-NO	18	326.1	18	180	18	326.1	26	117.6	18	180
M2-180-102.1-NO	42	335.2	47	180	42	335.2	47	180	47	180
M2-180-116.7-NO	28	315.1	28	180	28	315.1	47	55.4	28	180
M2-180-99.4-NO	42	356.0	46	169.1	42	356.0	46	77.7	53	180
Avg Value M2	39.2	348.2	**37.6**	147.5	42.9	348.4	43.9	99.6	**42.7**	180
M3-180-97.9-NO	17	347.2	11	116.8	17	347.2	15	78.9	11	180
M3-180-78.7-NO	42	402.6	71	79.1	42	402.6	71	79.1	71	180
M3-180-100.1-NO	21	387.9	21	180	21	387.9	21	112.4	21	180
M3-180-85.7-NO	17	391.5	47	180	17	391.5	47	78.2	61	180
M3-180-95.1-NO	42	412.1	12	180	42	412.1	42	53.2	12	180
M3-180-87.3-NO	17	342.9	47	180	17	342.9	47	34.8	47	180
M3-180-96.8-NO	17	374.5	62	180	17	374.5	64	139.7	62	180
M3-180-93.6-NO	12	372.8	47	180	12	372.8	49	177.7	47	180
M3-180-94.0-NO	11	371.2	43	34.8	11	371.2	43	34.8	59	180
M3-180-95.3-NO	42	466.7	27	89.0	42	466.7	27	89.0	65	180
M3-180-145.8-NO	73	499.1	47	180	73	499.1	73	131.1	71	180
M3-180-96.7-NO	34	720	30	180	47	445.4	40	138.0	30	180
M3-180-135.2-NO	17	484.3	40	180	17	484.3	60	150.0	66	180
M3-180-104.5-NO	42	396.1	69	37.8	42	396.1	69	48.8	71	180
M3-180-117.1-NO	11	304.0	15	112.0	11	304.0	15	112.0	15	180
M3-180-111.8-NO	11	372.6	29	180	11	372.6	29	105.9	29	180
M3-180-102.7-NO	42	390.2	18	180	42	390.2	26	118.0	18	180
M3-180-102.1-NO	11	351.0	47	180	11	351.0	47	180	47	180
M3-180-116.7-NO	11	332.7	28	55.8	11	332.7	47	55.8	28	180
M3-180-99.4-NO	11	320.6	46	77.7	11	320.6	46	77.7	53	180
Avg Value M3	**25.1**	403.8	37.8	138.2	**25.7**	388.3	43.9	99.8	44.2	180

Table 6: Results for the 180-minute disrupted instances

Instance	Best Sol [33]		Best Sol MH		Best Algo [33]		Best CE MH		Best RH MH	
	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)	Value (sec)	Time (sec)
M1-180-97.9-DIS	670	7.8	643	53.8	860	720	643	53.8	878	180
M1-180-78.7-DIS	697	415.2	788	145.8	994	720	827	156.5	994	180
M1-180-100.1-DIS	828	440.5	643	128.5	946	720	707	2.9	949	180
M1-180-85.7-DIS	916	720	625	89.4	916	720	692	12.2	965	180
M1-180-95.1-DIS	816	425.2	734	2.9	1026	720	734	2.9	908	180
M1-180-87.3-DIS	874	425.7	630	2.9	1043	720	630	2.9	946	180
M1-180-96.8-DIS	703	472.7	721	3.0	953	720	721	12.3	953	180
M1-180-93.6-DIS	640	464.8	326	180	640	720	703	2.9	670	180
M1-180-94.0-DIS	738	720	736	180	738	720	797	3.0	738	180
M1-180-95.3-DIS	830	506.4	671	180	830	720	694	17.2	717	180
M1-180-145.8-DIS	709	720	709	180	709	720	778	3.0	709	180
M1-180-96.7-DIS	716	720	798	180	716	720	842	3.0	891	180
M1-180-135.2-DIS	911	546.2	911	180	911	720	1078	3.6	911	180
M1-180-104.5-DIS	579	720	770	14.7	579	720	770	14.7	975	180
M1-180-117.1-DIS	691	422.1	740	180	923	720	772	131.6	740	180
M1-180-111.8-DIS	714	720	783	179.8	714	720	839	2.9	998	180
M1-180-102.7-DIS	582	575.5	694	3.6	698	720	694	3.7	709	180
M1-180-102.1-DIS	911	220.1	706	2.9	943	720	706	2.9	943	180
M1-180-116.7-DIS	659	466.7	659	180	828	720	772	57.5	659	180
M1-180-99.4-DIS	802	427.5	707	80.2	925	720	725	2.9	925	180
Avg Value M1	749.3	506.8	**699.7**	107.4	844.6	720	**756.2**	24.6	858.9	180
M2-180-97.9-DIS	463	720	643	68.3	463	720	643	68.3	878	180
M2-180-78.7-DIS	703	720	704	180	703	720	827	63.5	919	180
M2-180-100.1-DIS	916	552.8	643	91.8	949	720	643	91.8	1105	180
M2-180-85.7-DIS	709	720	586	99.3	709	720	586	99.3	916	180
M2-180-95.1-DIS	739	720	734	3.0	739	720	734	3.0	739	180
M2-180-87.3-DIS	709	720	611	31.2	709	720	630	2.9	828	180
M2-180-96.8-DIS	707	434.3	714	180	1134	720	802	2.9	834	180
M2-180-93.6-DIS	640	720	703	2.9	640	720	703	2.9	828	180
M2-180-94.0-DIS	703	432.0	797	2.9	1153	720	797	2.9	862	180
M2-180-95.3-DIS	830	720	613	88.8	904	720	694	20.1	904	180
M2-180-145.8-DIS	513	567.2	709	180	709	720	778	3.0	709	180
M2-180-96.7-DIS	800	720	696	180	1089	720	842	3.0	806	180
M2-180-135.2-DIS	911	586.5	787	180	911	720	1078	3.6	911	180
M2-180-104.5-DIS	580	720	758	180	580	720	770	17.7	962	180
M2-180-117.1-DIS	702	720	671	180	702	720	772	72.1	786	180
M2-180-111.8-DIS	708	625.1	708	180	708	720	770	180	708	180
M2-180-102.7-DIS	585	720	694	3.7	585	720	694	3.7	709	180
M2-180-102.1-DIS	607	486.3	706	2.9	641	720	706	2.9	943	180
M2-180-116.7-DIS	659	720	647	180	659	720	772	57.7	659	180
M2-180-99.4-DIS	592	514.1	701	2.9	730	720	701	106.3	849	180
Avg Value M2	**688.8**	641.9	691.2	100.9	770.8	720	**747.1**	40.4	842.7	180
M3-180-97.9-DIS	705	389.8	643	180	867	720	734	10.5	908	180
M3-180-78.7-DIS	703	720	683	10.0	703	720	843	12.1	958	180
M3-180-100.1-DIS	882	720	733	17.3	882	720	733	17.3	1105	180
M3-180-85.7-DIS	916	720	677	180	919	720	692	14.6	1002	180
M3-180-95.1-DIS	783	720	734	4.9	783	720	734	4.9	1026	180
M3-180-87.3-DIS	702	720	630	2.8	702	720	630	2.8	828	180
M3-180-96.8-DIS	939	462.4	714	180	1134	720	721	86.7	714	180
M3-180-93.6-DIS	640	720	625	36.4	640	720	625	36.4	828	180
M3-180-94.0-DIS	862	493.9	797	3.0	1153	720	797	3.0	1153	180
M3-180-95.3-DIS	830	720	609	2.9	830	720	898	4.6	946	180
M3-180-145.8-DIS	826	720	783	180	826	720	859	126.3	916	180
M3-180-96.7-DIS	806	606.3	701	180	1089	720	842	3.0	796	180
M3-180-135.2-DIS	911	595.6	787	180	911	720	1152	7.1	911	180
M3-180-104.5-DIS	751	720	750	180	751	720	865	2.3	954	180
M3-180-117.1-DIS	698	483.5	671	180	848	720	772	122.0	671	180
M3-180-111.8-DIS	862	513.0	671	10.0	1093	720	1072	5.8	884	180
M3-180-102.7-DIS	709	574.2	694	6.5	928	720	694	6.5	911	180
M3-180-102.1-DIS	828	511.8	706	2.9	828	720	706	2.9	828	180
M3-180-116.7-DIS	828	720	647	180	828	720	958	9.1	828	180
M3-180-99.4-DIS	730	720	725	128.7	730	720	801	123.2	973	180
Avg Value M3	795.5	627.5	**699.0**	92.3	872.2	720	**806.4**	30.1	907.0	180

Tables 7, 8, 9 present an aggregate comparison of the best solutions and algorithms obtained in [33] against the best solutions and algorithms proposed in this paper, respectively for the 60-minute delayed instances, the 180-minute delayed instances, the 180-minute disrupted instances. Specifically, we compare: the best solutions computed via the new metaheuristics versus the best known solutions in [33], the best centralized metaheuristic (CE MH) versus the best performing algorithm in [33], the best rolling horizon metaheuristic (RH MH) versus the best performing algorithm in [33].

Table 7: Performance comparisons for the 60-minute instances

Model	Comparison Between Solutions	Num Better Solut	Avg Value Reduction (sec)	Avg Time Variation (sec)	Num Equal Solut	Avg Time Variation (sec)	Num Worse Solut	Avg Value Increase (sec)	Avg Time Variation (sec)
M1	Best Sol MH vs Best Sol [33]	0	-	-	20	+20.0	0	-	-
	Best CE MH vs Best Algo [33]	0	-	-	18	+9.5	2	4.5	+35.2
	Best RH MH vs Best Algo [33]	0	-	-	19	+174.0	1	34.0	+151.9
M2	Best Sol MH vs Best Sol [33]	0	-	-	20	-10.6	0	-	-
	Best CE MH vs Best Algo [33]	1	1.0	-125.1	19	-4.6	0	-	-
	Best RH MH vs Best Algo [33]	0	-	-	17	+144.6	3	6.7	+81.3
M3	Best Sol MH vs Best Sol [33]	0	-	-	20	+15.6	0	-	-
	Best CE MH vs Best Algo [33]	1	26.0	+45.9	19	+26.6	0	-	-
	Best RH MH vs Best Algo [33]	1	26.0	+180.0	17	+172.0	2	6.5	+146.2

Table 8: Performance comparisons for the 180-minute instances

Model	Comparison Between Solutions	Num Better Solut	Avg Value Reduction (sec)	Avg Time Variation (sec)	Num Equal Solut	Avg Time Variation (sec)	Num Worse Solut	Avg Value Increase (sec)	Avg Time Variation (sec)
M1	Best Sol MH vs Best Sol [33]	6	5.5	-230.8	11	-183.1	3	11.0	-183.7
	Best CE MH vs Best Algo [33]	5	29.8	-169.1	2	-129.8	13	27.2	-199.1
	Best RH MH vs Best Algo [33]	6	25.3	-103.3	5	-62.3	9	28.8	-104.7
M2	Best Sol MH vs Best Sol [33]	4	21.3	-283.1	11	-194.3	5	11.0	-149.0
	Best CE MH vs Best Algo [33]	5	24.6	-235.8	5	-282.1	10	14.2	-238.6
	Best RH MH vs Best Algo [33]	5	32.2	-172.7	6	-153.3	9	17.3	-176.0
M3	Best Sol MH vs Best Sol [33]	6	17.5	-324.2	1	-207.9	13	27.8	-243.0
	Best CE MH vs Best Algo [33]	4	10.0	-306.4	3	-334.1	13	31.1	-272.5
	Best RH MH vs Best Algo [33]	5	15.8	-238.8	1	-207.9	14	32.1	-197.4

Table 9: Performance comparisons for the 180-minute disrupted instances

Model	Comparison Between Solutions	Num Better Solut	Avg Value Reduction (sec)	Avg Time Variation (sec)	Num Equal Solut	Avg Time Variation (sec)	Num Worse Solut	Avg Value Increase (sec)	Avg Time Variation (sec)
M1	Best Sol MH vs Best Sol [33]	10	160.4	-345.7	3	-397.6	7	87.4	-476.9
	Best CE MH vs Best Algo [33]	13	197.5	-684.7	0	-	7	114.3	-715.3
	Best RH MH vs Best Algo [33]	5	136.0	-540.0	7	-540.0	8	120.8	-540.0
M2	Best Sol MH vs Best Sol [33]	9	109.7	-571.7	1	-445.1	10	103.6	-523.0
	Best CE MH vs Best Algo [33]	9	187.4	-683.1	0	-	11	110.2	-676.8
	Best RH MH vs Best Algo [33]	3	291.3	-540.0	6	-540.0	11	210.2	-540.0
M3	Best Sol MH vs Best Sol [33]	20	96.6	-535.0	0	-	0	-	-
	Best CE MH vs Best Algo [33]	13	162.6	-695.7	0	-	7	113.9	-679.3
	Best RH MH vs Best Algo [33]	5	223.2	-540.0	4	-540.0	11	164.6	-540.0

In Tables 7, 8, 9, Column 1 gives the model variant compared; Column 2 the type of comparison; Columns 3–5 the number of better solutions obtained by the new metaheuristics, the average reduction of the objective function value (in seconds), the average variation of the time (in seconds) to compute the best solution (we note that a negative variation means that the new metaheuristics are faster, while a positive variation means that the new metaheuristics are slower); Columns 6–7 the number of equal solutions obtained by the new metaheuristics, the average variation of the best solution computation time (in seconds); Columns 8–10 the number of worse solutions obtained by the new metaheuristics, the average increase of the objective function value (in seconds), the average variation of the best solution computation time (in seconds). Each row presents the average results over the 20 instances of a specific model variant and type of instance.

In Table 7, the number of equal solutions is very large for all models and there is a small time variation in computing the best known solutions by the different approaches. This is related to the relative low complexity of the 60-minute delayed instances, and to the fact that the best known solutions in [33] were already optimal. Furthermore, the best algorithms in [33] are often very fast to compute the optimal solution, leaving a relatively small margin for further improving the computational performance. On average, the new metaheuristics take a larger computation time. However, the best CE MH presents a small increase of the computation time. Regarding the objective function value, the new metaheuristics compute three better solutions than the best algorithm in [33], for which the optimal solution is found by another algorithm.

In Table 8, for 16 out of 60 instances the new metaheuristics compute a new best known solution in a significantly smaller computation time compared to the one required to compute the previously best known solution. For other 23 instances, the best known solution is computed by new metaheuristics again in a smaller computation time compared to [33], while for the remaining instances 180 seconds of computation time are not sufficient in order to find the best known solution. In general, the main advantage of the new metaheuristics is related to the quickness to compute their best solution.

Table 9 shows that a new best known solution is found by the new metaheuristics in 39 out of 60 cases, the same best known solution is found in 4 cases but with a strongly reduced computation time, and a worsening solutions is found with 180 seconds of computation in the remaining cases. For each row of the table, the average objective function value increase is always smaller than the average objective function value reduction. For all cases, a main advantage of the new metaheuristics (specially in the centralized framework) is again a strongly reduced time to compute the best solution, that is an important requirement to fit the real-time application of the proposed methodology even for the most difficult instances.

4.5 Assessment of computational performance

This section presents a detailed performance assessment in the first 180 seconds of computation for a limited set of algorithms in the centralized framework: the best metaheuristics developed in this paper and the tabu search algorithm. The metaheuristics combined with the rolling horizon framework are not reported, since they deliver a feasible solution at the end of the given computation time (i.e. at 180 seconds). We also do not report the results obtained for the other approaches in [33], since we limit our assessment to the only algorithms that are able to find a feasible solution for all ATC-TCA instances in the given maximum computation time. We note that the commonly used First-In-First-Out rule (FIFO) is very fast but it computes a feasible solution for the undisrupted instances only. The other algorithms in [33] have a better feasibility rate than FIFO but they take a too long computation time for the 180-minute instances.

Figures 6, 7 and 8 report the plots of average best objective function values found at time $T = 10; 20; ...; 170; 180$ seconds, for the 60-minute delayed, the 180-minute delayed and the 180-minute disrupted instances. These figures show the results for five metaheuristics: TS, VNS DJ, VNS WOCP+FNWJ, VNTS DJ, VNTS WOCP+FNWJ. Each point of the plots gives the average values obtained at a given computation time. In case some infeasibility is reported at an intermediate time point T_i for a metaheuristic, its average value at T_i considers the worst solutions obtained at T_i by the other metaheuristics.

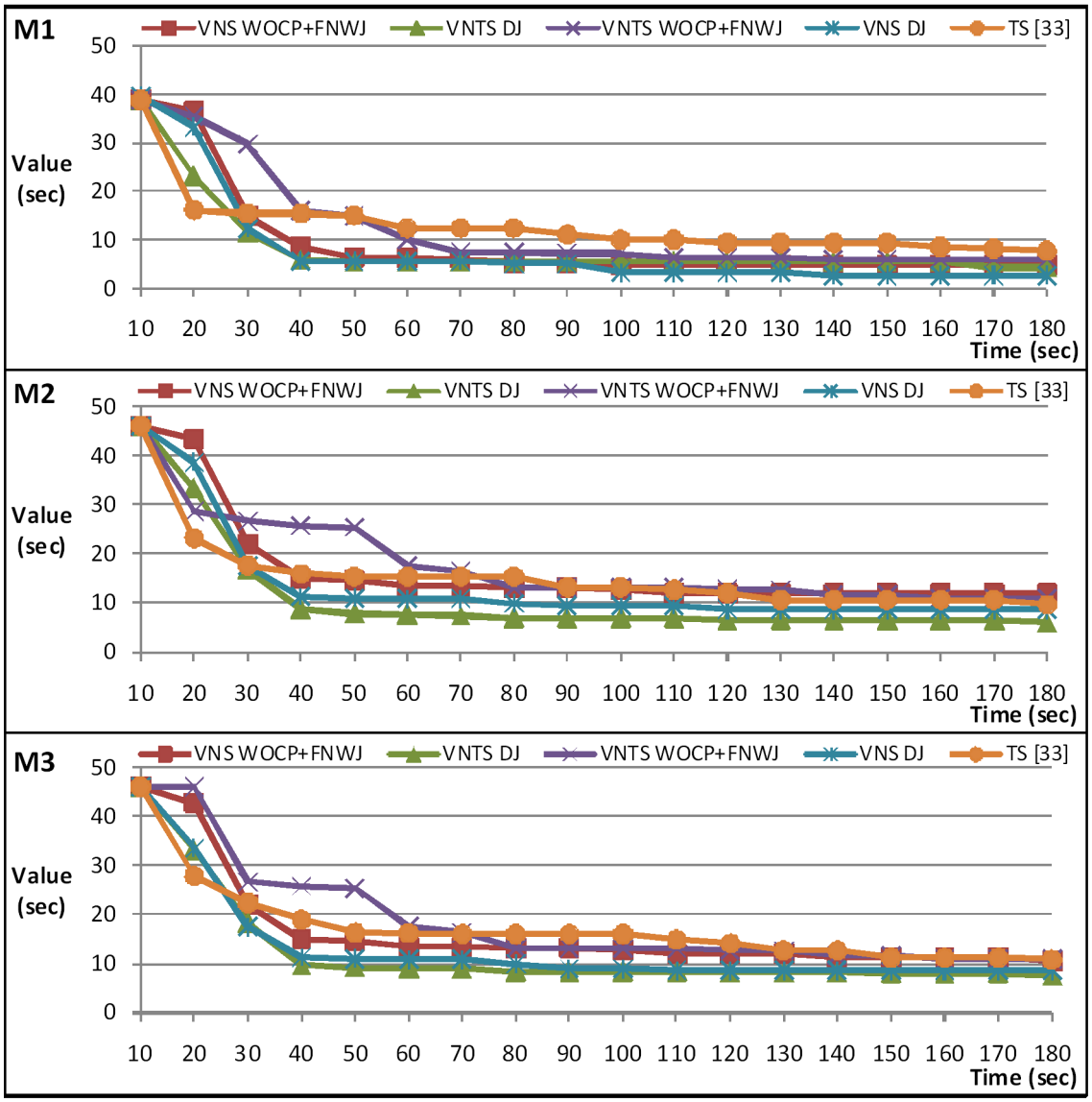


Figure 6: 60-minute delayed instances: results obtained for the metaheuristics

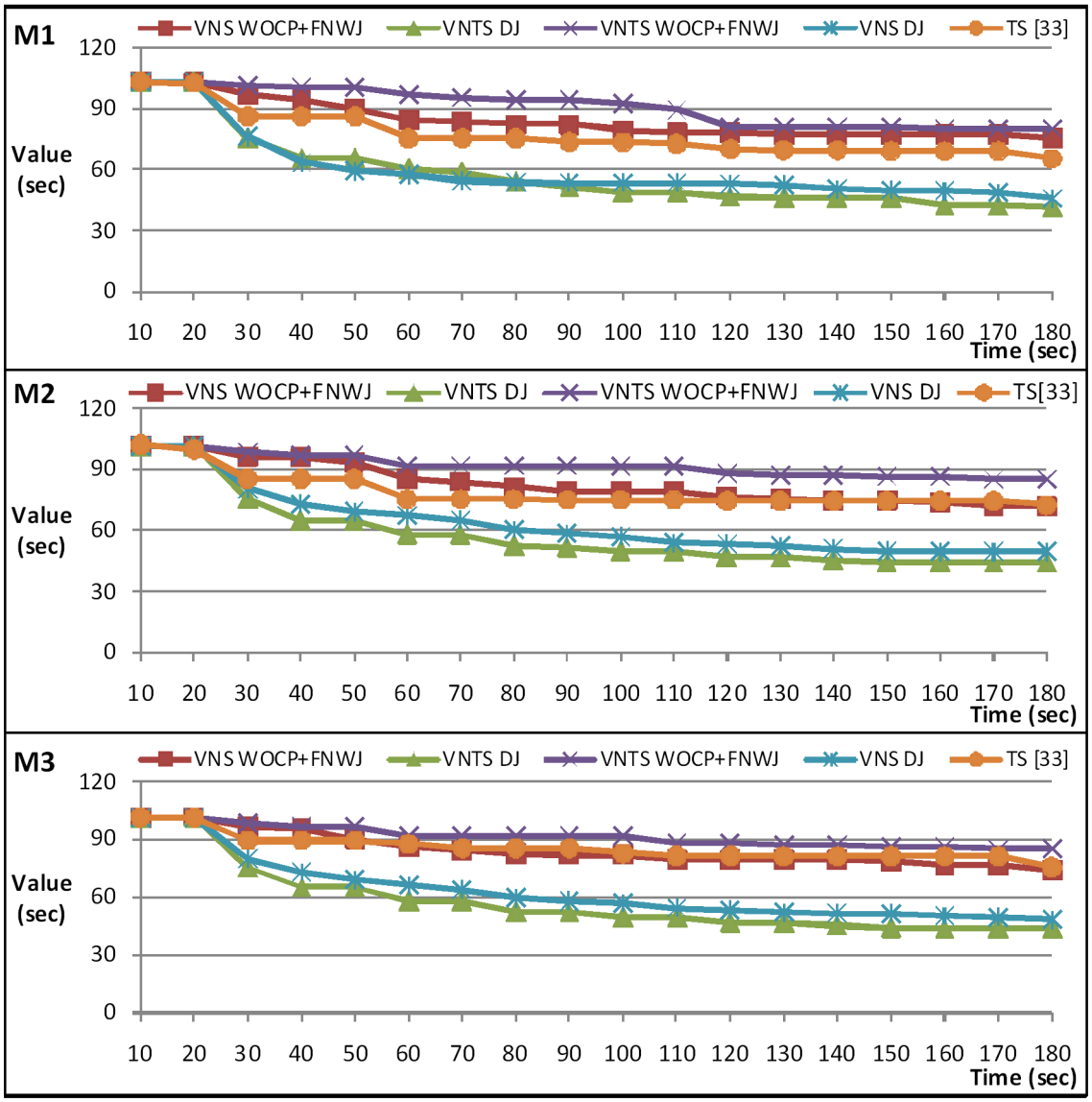


Figure 7: 180-minute delayed instances: results obtained for the metaheuristics

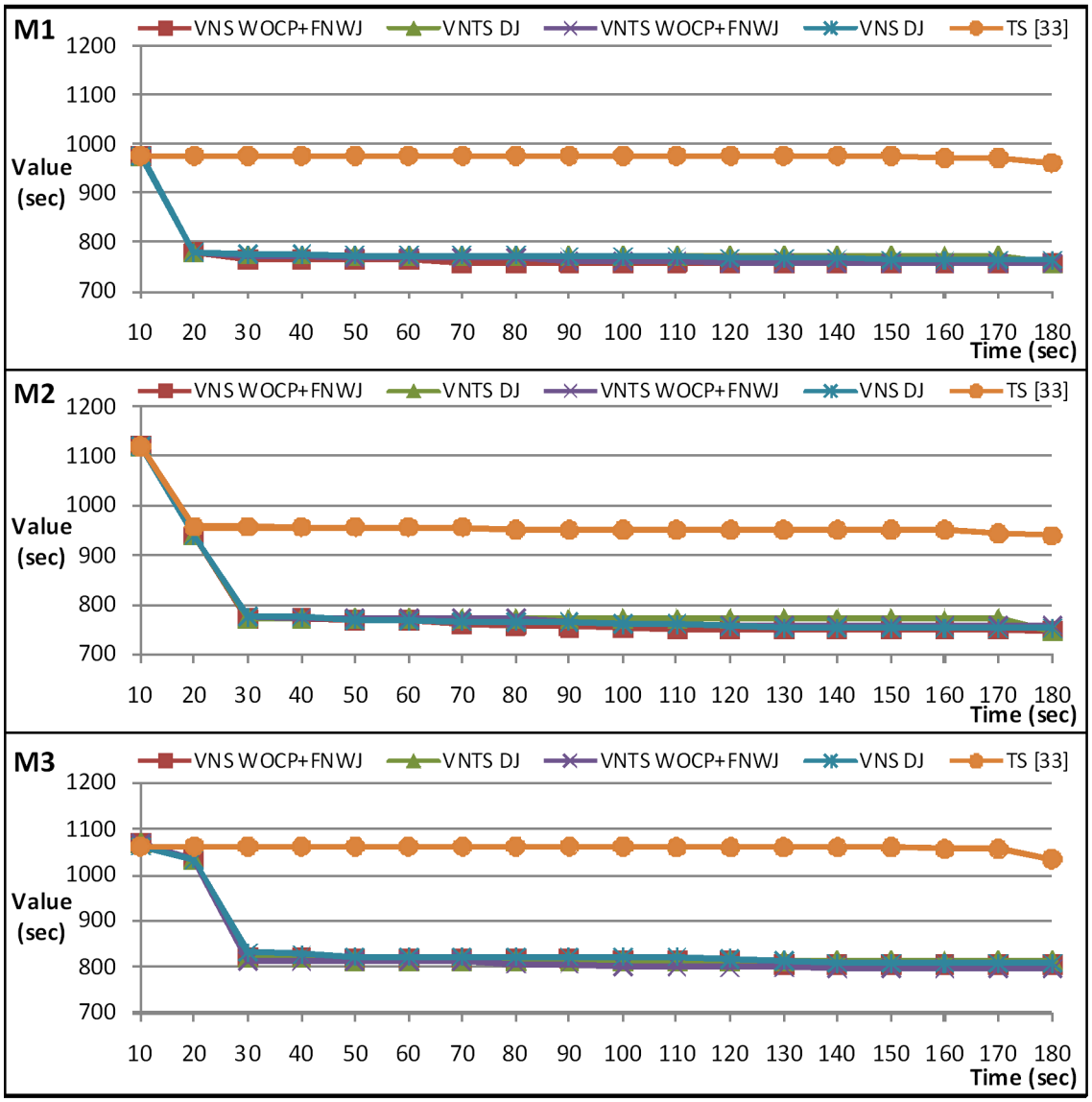


Figure 8: 180-minute disrupted instances: results obtained for the metaheuristics

From the plots in Figure 6, a number of observations follow for various time points. At 10 seconds of computation, the initial solution is provided, that is often the one computed by the branch-and-bound scheduling algorithm with default routes. At 20 seconds of computation, TS outperforms, on average, all the other metaheuristics. For large computation times, most of the new metaheuristics present better quality results. This is probably due to the different neighbourhoods and search strategies used: TS mostly performs single routing changes on the ramified critical path neighbourhood, while the new metaheuristics are based on multiple simultaneous routing changes in different neighbourhoods. Among the new metaheuristics, VNS DJ and VNTS DJ are the best performing algorithms. It is remarkable that the solution improvement compared to TS is mainly in the first 40 seconds of computation, even if further marginal improvements are obtained till the time limit of computation. At 180 seconds of computation, the combination of VNS DJ and VNTS DJ delivers the optimal solution for all instances. Looking at the gap between the initial and final solutions, the reduction in objective function is quite impressive for all metaheuristics. The latter trend shows the importance of aircraft re-routing during disturbed operations.

Figure 7 shows a different performance pattern compared to Figure 6. The best solution is often improved during the 180 seconds of computation for the 180-minute delayed instances, since the solution space is very large and the computation of good quality solutions takes more time. From the results of Figure 7, VNTS DJ is very competitive during the entire search process and offers the best solutions at the end of the computation time. The second best metaheuristic is VNS DJ, remarking the very positive performance of the DJ neighbourhood. All metaheuristics present a good improvement of the initial solution after one minute of computation, and further marginal improvements are obtained in the other two minutes of computation.

Figure 8 presents a quite flat performance pattern compared to the other cases. Taking the average solution at $T = 10$ seconds as reference solution, a significant improvement is obtained at 20 seconds for TS in case of M2, while the other metaheuristics have their largest improvements during the first 30 seconds of computation. All the new metaheuristics have a similar performance, but VNS WOCP+FNWJ gives slightly better average results. Some marginal reduction of the objective function value is also obtained during the remaining computation time for all metaheuristics. In general, the flat performance pattern after 30 seconds is due to the limited number of feasible solutions available for re-routing and scheduling aircraft traffic in case of disruption. However, for this set of instances the new metaheuristics provide their largest improvement of the objective function value compared to the previously best known solutions.

4.6 Discussion on the obtained results

Looking at the computational results, the main conclusions are next drawn. The new metaheuristics compute good quality solutions in much less time compared to state-of-the-art algorithms for the 180-minute traffic predictions. The results obtained by the new metaheuristics for the 60-minute traffic predictions are optimal, even if, for the latter instances, a slightly larger computation time is required compared with the best algorithms in [33]. A new best known upper bound value is provided for 55 out of the 120 instances for which no optimal solution is yet proven. For the disrupted instances, the new neighbourhoods and metaheuristic schemes, on average, outperform the best results obtained

with the TS algorithm (that is the only algorithm proposed in [33] that is able to find a feasible solution for all ATC-TCA instances during the first 180 second of computation) within one minute of computation on a standard processor.

When comparing the centralized versus the rolling horizon frameworks, the metaheuristics are, on average, faster in the centralized framework. However, the solutions provided by the combination of the metaheuristics with the rolling horizon framework help to find new best known solutions for a significant number of instances. The strength of the rolling horizon approach is the problem decomposition in small time horizons, that has the effect of simplifying the aircraft scheduling problem and thus enables a more effective evaluation of the potential routing moves during the neighbourhood search strategy.

The metaheuristics present a different performance for the different types of instances. Regarding the time horizon and type of disturbance, the 180-minute disrupted instances are more difficult to be solved compared to the other instances, since the 60-minute instances presents a reduced number of variables and the 180-minute delayed instances have a reduced number of deadline constraints. Regarding the model variants, no significant performance different is found when comparing M1 and M2, even if the two models have a different number of due date constraints. On the contrary, M3 is more difficult to solve than the other models, since there are additional deadline constraints compared to M1 and M2. We recall that the presence of deadline constraints may strongly reduce the set of feasible schedules.

For the instances with deadline constraints the new metaheuristics are performing very well compared to the approaches in [33], since these complex traffic situations require to perform multiple simultaneous re-routing actions frequently in order to find good quality solutions. Another important reason is the use of the deadline implications and the preprocessing procedure developed in [33], that help the metaheuristics to find good quality solutions and to compute better solutions than the commercial MILP solver.

5 Conclusions and future research

This paper proposes fast scheduling and routing metaheuristics for air traffic control at a busy TCA, with particular focus on the efficient control of strong traffic disturbances (such as multiple aircraft delays and a temporarily disrupted runway). To this end, several algorithmic innovations are considered, which relate to design of effective metaheuristics based on a decomposition of the problem into scheduling and routing decisions. A new set of neighbourhoods and new metaheuristic schemes are proposed on the basis of the hybridization between a variable neighbourhood search and a tabu search. Those algorithms are furthermore combined with a rolling-horizon based decomposition framework.

The algorithms and frameworks are evaluated on a Italian practical case study, i.e. the TCA of Milan Malpensa, and on a restricted group of instances that resulted the most challenging in [33]. Those instances can be considered some of the most complex instances in the literature of the ATC-TCA problem, including a large number of aircraft and various sources of disturbances. We also compared the performance of the various approaches for three model variants, with differences in the set of constraints.

The new metaheuristics are benchmarked against state-of-the-art ATC-TCA approaches which include optimization algorithms developed in the AGLIBRARY solver and a MILP formulation for simultaneous aircraft scheduling and routing solved by a commercial MILP

solver. The algorithms proposed in this paper, with new neighbourhood structures and search strategies, improve the effectiveness of the previously developed approaches. The main contributions the are next summarized: a general significant reduction of the time required to compute good quality solutions; a better performance in terms of solution quality and computation time, specially for the most difficult instances including disruptions; and the computation of new best known solutions for the instances for which no optimal solution is yet proven.

The average computational behaviour of the new metaheuristics is also analyzed in depth, resulting in a slightly slower descent compared to the TS (the best performing algorithm in [33] during the first three minutes of computation) in the first 20 seconds. However, at around 1 minute of computation the new metaheuristics offer the possibility to strongly reduce the objective function value computed by TS.

Future efforts should be dedicated to a path towards the implementation of advanced ATC-TCA approaches into a dynamic air traffic management system. The optimization models, algorithms and frameworks presented in this work should be a core component of the intelligent transport system. Other issues are worthwhile being investigated, including the development of good quality lower bounds for the ATC-TCA problem, the extension of our methodology to better deal with speed variations in the arriving and departing procedures, the integration with large-scale aircraft traffic control measures.

References

[1] Alonso-Ayuso, A., Escudero, L.F., Martín-Campo, F.J., Mladenović, N. (2012) VNS based algorithm for solving a 0–1 nonlinear nonconvex model for the Collision Avoidance in Air Traffic Management. *Electronic Notes in Discrete Mathematics* **39** 115–120.

[2] Alonso-Ayuso, A., Escudero, L.F., Martín-Campo, F.J., Mladenović, N. (2014) A VNS metaheuristic for solving the aircraft conflict detection and resolution problem by performing turn changes. *Journal of Global Optimization* DOI 10.1007/s10898-014-0144-8

[3] Atkin, J.A.D., Burke, E.K., Greenwood, J.S., Reeson, D. (2007) Hybrid Metaheuristics to Aid Runway Scheduling at London Heathrow Airport, *Transportation Science* **41 (1)** 90–106.

[4] Ball, M., Barnhart, C., Nemhauser, G., Odoni, A. (2007) Air Transportation: Irregular Operations and Control. *Handbooks in Operations Research and Management Science* **14(1)** 1–67.

[5] Balakrishnan, H., Chandran, B. (2010) Algorithms for scheduling runway operations under constrained position shifting. *Operations Research*, **58(6)** 1650–1665.

[6] Barnhart, C., Fearing, D., Odoni, A., Vaze, V. (2012) Demand and capacity management in air transportation. *EURO Journal of Transportation and Logistics* **1(1–2)** 135–155.

[7] Beasley, J.E., Sonander, J., Havelock, P. (2001) Scheduling aircraft landings at London Heathrow using a population heuristic. *Journal of the Operational Research Society* **52 (1)** 483–493.

[8] Bencheikh, G., Boukachour, J., Hilali Alaoui, A.E. (2011) Improved ant colony algorithm to solve the aircraft landing problem, *International Journal of Computer Theory and Engineering* **3 (2)** 224–233.

[9] Bennell, J.A., Mesgarpour, M., Potts, C.N. (2011) Airport runway scheduling. *4OR – Quarterly Journal of Operations Research* **4(2)** 115–138.

[10] Bianco, L., Dell'Olmo, P., Giordani, S. (2006) Scheduling models for air traffic control in terminal areas. *Journal of Scheduling* **9 (3)** 180–197.

[11] Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M. (2010) A tabu search algorithm for rerouting trains during rail operations. *Transportation Research – Part B* **44 (1)** 175–192.

[12] D'Ariano, A., Pacciarelli, D., Pranzo, M. (2007) A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research* **183 (2)** 643–657.

[13] D'Ariano, A., Pacciarelli, D., Pistelli, M., Pranzo, M. (2015). Real-time scheduling of aircraft arrivals and departures in a terminal maneuvering area. *Networks, Wiley.* TO APPEAR
http://www.dia.uniroma3.it/Plone/ricerca/technical-reports/2012

[14] D'Ariano, A., D'Urgolo, P., Pacciarelli, D., Pranzo, M. (2010) Optimal sequencing of aircrafts take-off and landing at a busy airport. *Proceedings of the 13th International IEEE Annual Conference on Intelligent Transportation Systems* 1569–1574. Madeira Island, Portugal.

[15] D'Ariano, A., Pistelli, M., Pacciarelli, D. (2012) Aircraft retiming and rerouting in vicinity of airports. *IET Intelligent Transport Systems Journal* **6(4)** 433–443.

[16] Glover, F. (1986) Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research* **13 (5)** 533–549.

[17] Hancerliogullari, G., Rabadi, G., Al-Salem, A.H., Kharbeche, M. (2013) Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem, *Journal of Air Transport Management* **32** 39–48.

[18] Hansen, V.J. (2004) Genetic search methods in air traffic control. *Computers and Operations Research* **31 (3)** 445–459.

[19] Hansen, P., Mladenović, N., Moreno Pérez, J.A. (2008) Variable neighbourhood search: methods and applications, *4OR – Quarterly Journal of Operations Research* **6 (4)** 319–360.

[20] Hu, X.-B., Di Paolo, E. (2008) Binary-Representation-Based Genetic Algorithm for Aircraft Arrival Sequencing and Scheduling, *IEEE Transactions on Intelligent Transportation Systems* **9 (2)** 301–310.

[21] Jiang, Y., Xu, Z., Xu, X., Liao, Z., Luo, Y. (2014) A Schedule Optimization Model on Multirunway Based on Ant Colony Algorithm, *Mathematical Problems in Engineering*, Volume 2014, 11 pages, http://dx.doi.org/10.1155/2014/368208

[22] Kohl, N., Larsen, A., Larsen, J., Ross, A., Tiourine, S. (2007) Airline disruption management – Perspectives, experiences and outlook. *Journal of Air Transport Management* **13 (3)** 149–162.

[23] Kuchar, J.K., Yang, L.C. (2000) A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems* **4 (1)** 179–189.

[24] Mascis, A. and Pacciarelli, D. (2002) Job shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research* **143 (3)** 498–517.

[25] Moreno Pérez, J.A., Moreno-Vega, J.M., Rodríguez Martín, I. (2003). Variable neighborhood tabu search and its application to the median cycle problem. *European Journal of Operational Research* **151 (2)** 365–378.

[26] Pacciarelli, D., Pranzo, M. (2003) Production scheduling in a steelmaking-continuous casting plant. *Computers & Chemical Engineering* **28 (12)** 2823–2835.

[27] Pellegrini, P., Castelli, L., Pesenti, R. (2012) Metaheuristic algorithms for the simultaneous slot allocation problem. *IET Intelligent Transport Systems* **6(4)** 453–462.

[28] Pellegrini, P., Rodriguez, J. (2013) Single European sky and single European railway area: A system level analysis of air and rail transportation. *Transportation Research – Part A* **57 (1)** 64–86.

[29] Pinol, H., Beasley, J.E. (2006) Scatter Search and Bionomic Algorithms for the aircraft landing problem. *European Journal of Operational Research* **171 (2)** 439–462.

[30] Salehipour, A., Modarres, M., Moslemi Naeni, L. (2013) An efficient hybrid metaheuristic for aircraft landing problem. *Computers and Operations Research* **40 (1)** 207–213.

[31] Salehipour, A., Moslemi Naeni, L., Kazemipoor, H. (2009) Scheduling aircraft landings by applying a variable neighborhood descent algorithm: Runway-dependent landing time case, *Journal of Applied Operational Research* **1 (1)** 39–49.

[32] Samà, M., D'Ariano, A., Pacciarelli, D. (2013) Rolling horizon approach for aircraft scheduling in the terminal control area of busy airports. *Transportation Research – Part E* **60 (1)** 140–155.

[33] Samà, M., D'Ariano, A., D'Ariano, P., Pacciarelli, D. (2014) Optimal aircraft scheduling and routing at a terminal control area during disturbances. *Transportation Research – Part C* **47 (1)** 61–85.

[34] Taylor, C. and Wanke, C. (2011) Dynamically Generating Operationally-Acceptable Route Alternative Using Simulated Annealing. *Ninth USA/Europe Air Traffic Management Research and Development Seminar (ATM 2011).* Berlin, Germany.

[35] Zhang, Y. (2008) *Real-time Inter-modal Strategies for Airline Schedule Perturbation Recovery and Airport Congestion Mitigation under Collaborative Decision Making (CDM).* University of California Transportation Center, UCTC Dissertation No. 154.

[36] Zhan, Z-H., Zhang, J., Li, Y., Liu O., Kwok, S.K., Ip, W.H., Kaynak, O. (2010) An efficient ant colony system based on receding horizon control for the aircraft arrival sequencing and scheduling problem. *IEEE Transactions on Intelligent Transportation Systems* **11 (2)** 399–412.

APPENDIX

A small illustrative example is proposed to explain the basic characteristics of the neighbourhoods, the neighbours and their evaluations. We consider 4 aircraft (J1,J2,J3,J4) on the network of Figure 9. The network is composed by 17 resources: 1–3 are holding resources, 4–14 are air segments, 15 is a common glide path, 16 and 17 are runways. Three aircraft (J1,J2,J4) are arriving, while aircraft J3 is departing.

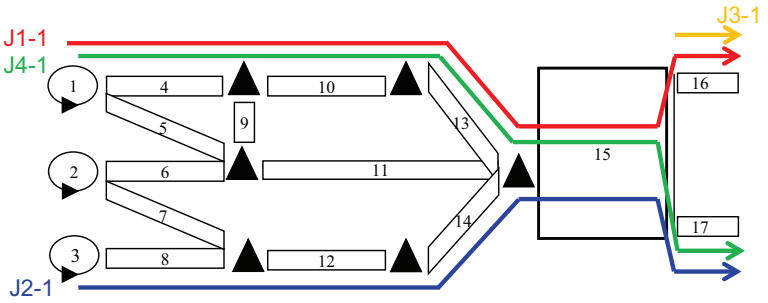


Figure 9: Illustrative network and aircraft default routes

Each aircraft has different possible available routings. J1 and J4 have 8 routing alternatives: the following ways to reach the resource 15: 4-10-13, 4-9-11, 5-9-10-13, 5-11; plus the choice of the runway (resource 16 or 17). J2 and J3 can exploit re-routing only on the resources 16 or 17, for two possible routings.

Figure 9 considers a default route for each aircraft, which is as follows: **J1-1:** 1, 4, 10, 13, 15, 16, out; **J2-1:** 3, 8, 12, 14, 15, 17, out; **J3-1:** 16, out; **J3-1:** 1, 4, 10, 13, 15, 17, out.

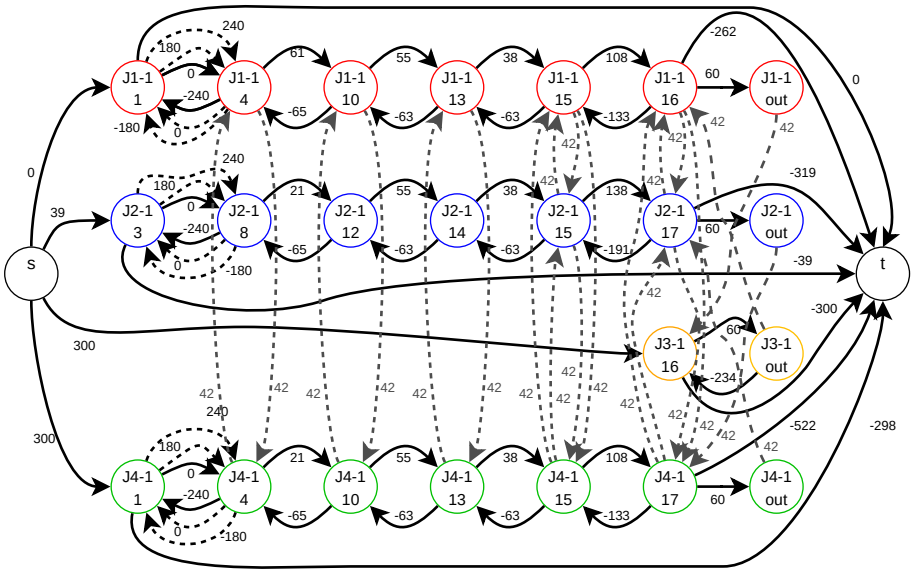


Figure 10: Alternative graph of the example with aircraft default routes

Figure 10 shows the alternative graph model of the ATC-TCA problem with the default routes. Each aircraft (identified by a different color) results in a sequence of

nodes (operations) reported horizontally and identified by an aircraft routing label plus the resource processed (i.e. (J2-1,3) means that the routing 1 of J2 goes over resource 3). The weight of each arc is reported for all arcs. Specifically, the fixed arcs are reported in solid color, while the alternative arcs are reported in dotted color. The latter arcs are given for holding circle resources 1 and 3 $\in A_{HC}$, for air segments resources 4, 10, 13, 15, 16 $\in A_{AS}$, for runway resources 16 and 17 $\in A_{RW}$, all reported in different dotted styles.

The selection of alternative arcs reported in Figure 11 is a solution to the ATC-TCA problem with the default routes. Regarding the aircraft sequencing decisions, J1 precedes J2 over all common resources; J3 takes off after J1 has landed; J4 goes after all others. No holding circle is performed by any aircraft.

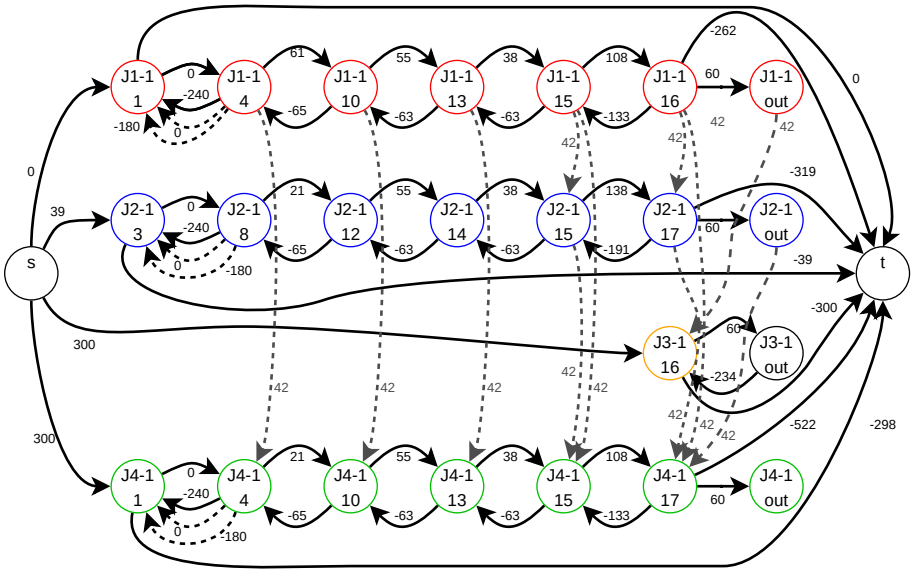


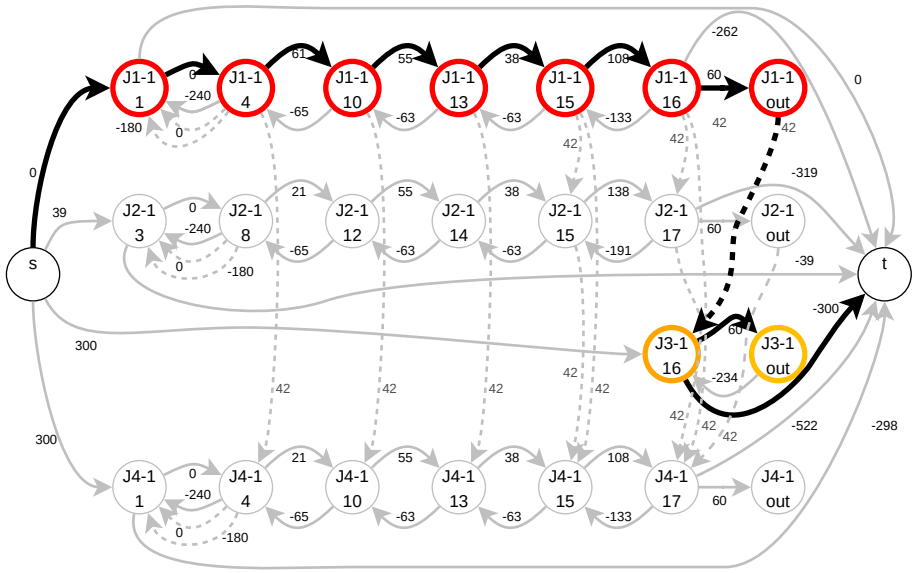
Figure 11: ATC-TCA solution for the example with aircraft default routes

We next discuss four neighbourhoods explained in Section 3.3 in terms of the aircraft ranking.

Free-Net Waiting Operations Jobs neighbourhood $\mathcal{N}_{FNWJ}$: The $\mathcal{N}_{FNWJ}$ ranking is based on the sum of the consecutive delays collected at each waiting node in the graph of the incumbent routing solution in which all alternative arcs are unselected (i.e. free-net traffic situation) but the one generating the waiting node. For example, let's consider the pair (((J1-1,out), (J3-1,16)),((J3-1,out), (J1-1,16))). If the arc ((J1-1,out), (J3-1,16)) is selected, the consecutive delay at node (J3-1,16) is 364-300=64. In fact, the weight of the longest path from s to (J3-1,16) in the graph of Figure 10 is 300 (i.e. $w^{F_{rt}}_{(s),(J3-1,16)}$), while the weight of the longest path from s to (J3-1,16) in the graph of Figure 10 with the only alternative arc ((J1-1,out), (J3-1,16)) selected is: 0 (related to $w^{F_{rt}}_{(s),(J1-1,1)}$) + 0 (related to $w^{F_{HC}}_{(J1-1,1),(J1-1,4)}$) + 61 (related to $w^{F_{AS}}_{(J1-1,4),(J1-1,10)}$ + 55 (related to $w^{F_{AS}}_{(J1-1,10),(J1-1,13)}$) + 38 (related to $w^{F_{AS}}_{(J1-1,13),(J1-1,15)}$) + 108 (related to $w^{F_{AS}}_{(J1-1,15),(J1-1,16)}$) + 60 (related to $w^{F_{RW}}_{(J1-1,16),(J1-1,out)}$) + 42 (related to $w^{A_{RW}}_{(J1-1,out),(J3-1,16)}$) = 364. From similar calculations, if the arc ((J3-1,out), (J1-1,16)) is selected, the consecutive delay at node (J1-1,16) is 402-262=140. The $\mathcal{N}_{FNWJ}$ ranking considers the minimum consecutive delay generated on the two waiting nodes related to each alternative pair, i.e. min(64, 140)=64 is assigned to J1-1 and J3-1 due to the pair (((J1-1,out), (J3-1,16)),((J3-1,out), (J1-1,16))). The

complete score of each aircraft is computed by summing up the minimum consecutive delay generated by each alternative pair. In this example there are other two alternative pairs that generate a score: ((J1-1,15, J2-1,15),(J2-1,17, J1-1,16)) with a score equal to the maximum between 196-153=43 and 333-262=71; ((J2-1,15, J1-1,15),(J1-1,16, J2-1,17)) with a score equal to the maximum between 195-154=41 and 304-291=19. The final ranking values for $\mathcal{N}_{FNWJ}$ are: J1-1: 64+43+19 = 126; J2-1: 43+19 = 62; J3-1: 64; J4-1: 0.

Ramified Critical Path Operations neighbourhood $\mathcal{N}_{RCPO}$: For the solution of Figure 11, the ramified critical path is highlighted in Figure 12. The operations on the critical path are as follows: (J1-1,1),(J1-1, 4),(J1-1, 10),(J1-1, 13),(J1-1, 15),(J1-1, 16),(J1-1,out), (J3-1,16). The ramified critical path is an extention of the critical path including also operation (J3-1,out). In $\mathcal{N}_{RCPO}$, the ranking values for each aircraft are computed as the maximum value $l^{S'(F^R)}(s, krp) + l^{S'(F^R)}(krp, t)\ \forall\ (krp)$ in the ramified critical path of the graph of the incumbent solution. This value corresponds exactly to the makespan, i.e. 64, for all nodes in the critical path. Regarding the extra node on the ramified critical path, the value can be computed as $l^{S'(F^R)}(s, (J3-1, out)) + w^{F_{rt}}_{(J3-1,out),(J3-1,16)} + l^{S'(F^R)}((J3-1, 16), t)$ = 424 - 234 - 300 = - 110. The score of J3-1 is the maximum between 64 and -110, the score of J1-1 is 64, the score of J2-1 and J4-1 is 0. The ranking of $\mathcal{N}_{RCPO}$ is reported in Figure 12.

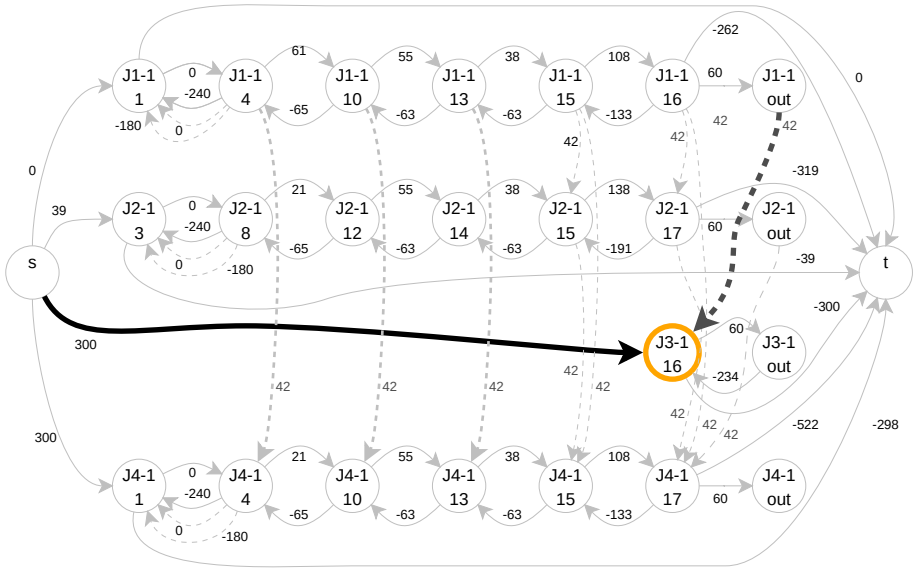


Aircraft	Ranking
J1-1	64
J2-1	0
J3-1	64
J4-1	0

Figure 12: Ramified critical path

Waiting Operations Critical Path neighbourhood $\mathcal{N}_{WOCP}$: The $\mathcal{N}_{WOCP}$ ranking is based on the sum of the consecutive delays collected on the waiting operations on the critical path of the incumbent solution. In the example, there is only one waiting operation (J3-1,16) in which a consecutive delay of value 64 is collected for J3-1. The consecutive delay (64) of J3-1 is computed as the weight of the path $l^{S'(F^R)}(s, (J1-1, out)))$ plus the weight $w^{F_{RW}}_{(J1-1,out),(J3-1,16)}$ (42) minus the weight of $l^{S'(F^R)}(s, (J3-1, 16)))$ (300). The other aircraft has no waiting operation on the critical path. The ranking of $\mathcal{N}_{WOCP}$ is reported in Figure 13.

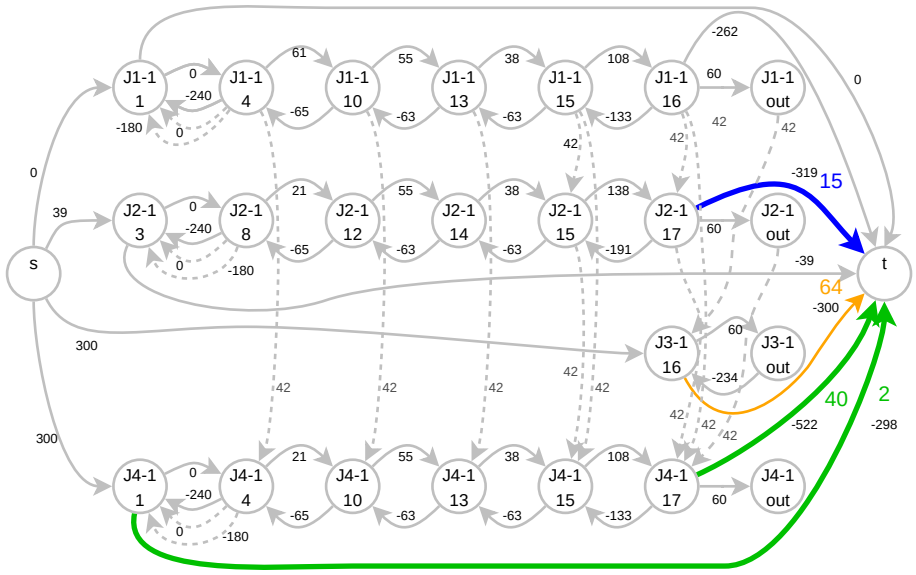
Delayed Jobs neighbourhood $\mathcal{N}_{DJ}$: The $\mathcal{N}_{DJ}$ ranking is based on the maximum consecutive delays on the due date arcs for each aircraft (job). Figure 14 highlights the



Aircraft	Ranking
J1-1	0
J2-1	0
J3-1	64
J4-1	0

Figure 13: Waiting operations critical path

due date arcs in which some consecutive delays are collected. J1-1 is not delayed and its score is thus 0. The other aircraft have the following consecutive delay on their last operation: J2-1 15 seconds, J3-1 64 second, J4-1 40 seconds. J4-1 has also a consecutive delay in its first operation, but the largest delay is collected on its last operation. The ranking of $\mathcal{N}_{WOCP}$ is reported in Figure 14.



Aircraft	Ranking
J1-1	0
J2-1	15
J3-1	64
J4-1	40

Figure 14: Delayed jobs