Università degli Studi di Roma Tre
Dipartimento di Informatica e Automazione
Via della Vasca Navale, 79 – 00146 Roma, Italy

Optimal sequencing of aircrafts take-off and landing at a busy airport

Andrea D'Ariano[1], Paolo D'Urgolo[1], Dario Pacciarelli[1] and Marco Pranzo[2]

RT-DIA-173-2010 **June 2010**

(1) Università degli Studi Roma Tre,
Via della Vasca Navale, 79
00146 Roma, Italy.

(2) Università degli Studi di Siena,
Via Roma 56
53100 Siena, Italy.

This work is partially supported by the Italian Ministry of Research, Grant number RBIP06BZW8, project FIRB "Advanced tracking system in intermodal freight transportation".

ABSTRACT

This paper studies the problem of sequencing aircraft take-off and landing operations at congested airports. We introduce and analyze alternative detailed formulations and solution algorithms for scheduling arrival and departure times of the aircrafts, such that the delay with respect to the scheduled times is minimized. The aircraft scheduling problem (ASP) is viewed as an extension of the job shop scheduling problem with additional real-world constraints and formulated by using alternative graphs. Two alternative formulations model the required time separation among aircrafts in air segments and runways according to safety regulations and differ for the level of detail used to represent the holding circles. Scheduling rules, heuristic and exact methods are implemented and tested on practical size instances of the Fiumicino airport, the busiest airport in Italy. We show that two versions of an innovative branch and bound algorithm are always able to find good solutions in a few seconds and often improve the best solution computed by the scheduling heuristics. Optimality is proved in less than two minutes for more than half of the instances.

1 INTRODUCTION

Fiumicino airport (FCO) is one of the world's busiest airports by passenger traffic and the main airport in Italy. Passenger traffic increased of 73% between 1996 and 2006 [5] and a further increase of traffic is forecasted in the coming years. With the increase in air traffic, airports are becoming a major bottleneck in Air Traffic Control (ATC) operations. Aviation authorities are thus seeking methods to better use the existing airport infrastructure and to better manage aircraft movements in the vicinity of airports, improving aircraft punctuality while maintaining the required level of safety. In this context, the optimization of take-off/landing operations is a key factor to improve the performance of the entire ATC system. However, the real-time air traffic management is still mainly under the control of human controllers whose computer support is most often limited to a graphical representation of the current aircraft position and speed. ATC decisions can be broadly divided into:

- Routing decisions, where a route for each aircraft has to be chosen from its current position to its destination.
- Scheduling decisions, where routes are considered fixed, and feasible aircraft sequencing and timing have to be determined in each airway, such that safety regulations are satisfied and traversing times are minimized.

One of the main objectives of routing decisions is to balance the use of critical resources, e.g., alternative runways. The objective of real-time scheduling decisions is typically the delay minimization. We refer to the latter problem as the Aircraft Scheduling Problem (ASP). We deal with the development of a reliable decision support system for aircraft scheduling at a busy airport. In this paper, the routing problem is solved off-line in a preliminary step and the main focus is on real-time scheduling decisions with fixed routes.

We now briefly recall the general procedure for take-off/landing operations in the proximity of airports. For each Terminal Maneuvering Area (TMA), landing aircrafts move along predefined routes from an entry fix to a runway following a standard descent profile. During all the approach phases, a minimum separation distance between every pair of consecutive aircrafts must be guaranteed. This standard separation depends on the type and relative positions of the two aircrafts (at the same or different altitude). Separations between aircrafts are mandatory and recognized by international aviation regulations. By considering the different aircraft speeds, this safety distance can be translated in a Separation Time Interval (STI). Similarly, departing aircrafts leave the runway flying towards the assigned exit fix along an ascent profile, respecting separation standards. The runway can be occupied by only one aircraft at a time. Holding circles are used to stack the aircrafts until they can be guided into the landing sequence. A departing aircraft can leave a holding circle only after the traversing of (or a multiple of) half length of the circle.

Departing aircrafts are supposed to take-off within their respective assigned time slots. An aircraft is late whenever it is not able to accomplish the departing procedure within its assigned time slot. A landing aircrafts is late when landing after its scheduled arrival time. A conflict occurs whenever two or more aircrafts do not respect the minimum required distance at the same air segment or runway. Any potential conflict between

landing and departing aircrafts must be detected and solved in real-time by human air traffic controllers by rescheduling aircraft movements [2], i.e., by solving an instance of the ASP.

The ASP has been the subject of many papers (see, e.g., the reviews in [2, 10]). Two cases of ASP can be distinguished: the static and the dynamic ASP. In the *static* ASP one wants to sequence landing/departing aircraft when all the information is known in advance. Beasley et al. [3] present a mixed-integer zero-one formulation for the static ASP in the single and multiple runways cases. The problem is then solved using exact and heuristic algorithms. Ernst et al. [7] tackle the static ASP of aircraft landings by using a specialized simplex algorithm for the single runway case and extend it to the multiple runway case. Hansen [8] proposes a genetic algorithm for the routing and sequencing of landing aircraft on a multiple runway case. Other methods are based on airlines' priorities [14]. In the *dynamic* ASP aircrafts enter the system one at the time. A new sequence of take-off/landing aircraft has to be recomputed every time a future incoming aircraft is known. In such cases, an effective solution approach consists of constraining the set of feasible positions in the sequence for the new aircraft to avoid excessive perturbations to the schedule [4]. With this constraint, known as Constrained Position Shifting (CPS), no aircraft can be sequenced forward or backward more than a specified number of positions with respect to the First In First Out (FIFO) sequence. CPS is also used by Psaraftis [13] that develops a dynamic programming-based approach for the static ASP case. Venkatakrishnan et al. [15] generalize the work of [13] to the dynamic case.

Most scheduling models in literature model the TMA as a single resource, typically the runway, so that the ASP is a single machine scheduling problem with some additional constraints. These models, that ignore the air space in the TMA, are not realistic since bottleneck situations may happens not only on the runways but also in the air segments. A recent stream of research includes airways in the ASP formulation and presents the problem as a job shop scheduling problem plus additional constraints [1, 4]. With these models, airways are divided into air segments and runways and each air segment/runway is a machine. The ASP can be viewed as a job shop since aircrafts corresponds to jobs, each traversing a prescribed sequence of machines. The traversing of a air segment/runway by a specific aircraft is known as an operation. The processing time of the operation is equal to the traversing time of the associated air segment/runway machine, which depends on the aircraft characteristics. Each operation has to be executed without interruption, under aircraft speed constraints. The constraints on the separation time intervals between consecutive aircrafts are represented as sequence-dependent set-up times.

This paper describes two new variants of the microscopic formulation of the ASP proposed by [1], based on the alternative graph model of [11]. Differently from others approaches based on job shop scheduling, the alternative graph enables an accurate representation of the air traffic regulations to be taken into account when solving the ASP. Our models are similar to the one of [4] but differ for the inclusion of additional real-world constraints such as holding circles, time windows of [minimum, maximum] allowed aircraft speeds, multiple capacity of air segments and blocking constraints at runways. We describe and test innovative exact algorithms for the ASP on practical size instances of the FCO airport and compare them with heuristics and dispatching rules similar to those described in [1, 4]. We show that optimal/near-optimal solutions can be found in short computation time. These solutions can be used to assess the quality of fast heuristics

(in terms of delays, travel time spent and delayed aircrafts) and to support air traffic controllers.

The next section introduces our two alternative graph formulations of the ASP, Section III describes heuristic and exact solution methods, Section IV presents the computational results on the FCO airport and Section V gives conclusions and further research directions.

2 ALTERNATIVE GRAPH FORMULATIONS

This section describes the TMA of Fiumicino airport and the formulations of ASP. At FCO there are three runways (16L, 16R, 25), two of which (16R, 25) are intersecting and cannot be used simultaneously. Each aircraft flies through a prescribed sequence of machines and overtaking is not allowed within an air segment in the TMA. Air traffic regulations impose that two consecutive aircrafts flying on the same air segment must respect a minimum longitudinal separation depending on their respective size (heavy, medium or light). This distance translates into a STI between the entrance/exit of consecutive aircrafts in the same air segment. A runway is a blocking resource, since the presence of an aircraft on the runway imposes that no other aircraft can use it. The intersecting runways act as a single blocking resource and are represented as a single machine. Figure 1(a) shows the TMA of FCO divided into 3 holding circles (TAQ, CMP, CIA; numbered from 1 to 3), 7 air segments for landing procedures (4 to 10), 3 air segments for take-off procedures (14 to 16) leading to the exit points of the TMA (ELVIN, RAVAL, BOL), two runways (12 and 13) and a common glide path (11). The latter includes the last air segments before the two runways. Since these two air segments are parallel and quite close, besides the minimum longitudinal distance between consecutive aircrafts on the same air segment, traffic regulations also impose a minimum diagonal distance between aircrafts flying on different air segments. The two air segments act thus as a single special machine in which setups depend not only on the sequence but also on the route chosen for consecutive aircrafts.

The ASP is now formulated in terms of alternative graphs. The alternative graph model introduced in [6, 11] has already been effectively used to model complex real-world job-shop scheduling problems arising in public transportation and manufacturing. An alternative graph is a triple $\mathcal{G} = (N, F, A)$, where $N = \{0, 1, \ldots, n, *\}$ is the set of nodes, F is a set of directed arcs (*fixed*) and A is a set of pairs of directed arcs. Arcs in the set A, *alternative*, are the decision variables. If $((i, j), (h, k)) \in A$, arc (i, j) is the alternative of arc (h, k). Each arc (i, j) is either fixed or alternative and has an associated weight w_{ij}. The nodes are associated to relevant events, such as the starting/completion of the schedule (nodes $0/*$) or the starting of an operation (nodes $1, \ldots, n$), e.g., the entrance of an aircraft in an air segment/runway. We call t_i the starting time associated to event i. A *selection* S is a set of alternative arcs, at most one from each pair. A selection, in which exactly one arc is chosen from each pair in A, is a *feasible schedule* (i.e., a solution to the ASP) if the graph $(N, F \cup S)$ has no positive length cycles. Given a feasible schedule, a timing t_i for operation i is the length of a longest path from 0 to i. A feasible schedule is an *optimal* solution if the length of a longest path from 0 to $*$ is minimum over all the

solutions. The general formulation of the ASP is therefore:

$$\begin{array}{ll} \min t_* - t_0 & \\ s.t. & \\ t_j - t_i \geq w_{ij} & (i,j) \in F \\ (t_j - t_i \geq w_{ij}) \vee (t_k - t_h \geq w_{hk}) & ((i,j),(h,k)) \in A \end{array}$$

An illustrative example with the two formulations of ASP is given for two landing aircrafts (A and B) and a departing aircraft (C) whose routes are depicted in Figure 1(a). Each node of the graph represents an operation, e.g., $A9$ is aircraft A entering air segment 9. Fixed (alternative) arcs are depicted with solid (dotted) arrows. We assume infinite capacity at the holding circles, and there are thus no conflicts on the holding circle machine between A and B (machine 1). For the aircraft routes of Figure 1(a), there are potential conflicts at two air segments (4 and 11) and at one runway (12), since these machines are used by more than one aircraft.

For each landing aircraft, its entrance time in the TMA is taken as a constraint for the problem in order to guarantee compatibility of the solution with its actual position. This is formulated in Figure 1(b), with a pair of fixed arcs for each aircraft: $(A0, A1)$ and $(A1, A0)$ for aircraft A, $(B0, B1)$ and $(B1, B0)$ for B, with weights $w_{(A1,0)} = -w_{(0,A1)}$ and with $w_{(B1,0)} = -w_{(0,B1)}$. The departing aircraft C is constrained to leave after its minimum departure time, which is formulated with arc $(0, C12)$.

The holding circle permits an aircraft to run around the airport before starting the landing procedure. An aircraft can leave a holding circle only after a multiple of the time needed to complete a half circle. In the example of Figure 1(b), two holding circle constraints are shown, one per landing aircraft. Each landing aircraft can either skip the holding circle or travel a half circle of weight γ. The two alternative decisions are formulated with two fixed arcs and a pair of alternative arcs. For aircraft A, the two fixed arcs are the solid arrows $(A1, A4)$ and $(A4, A1)$ with weights 0 and $-\gamma$. The alternative arcs $((A1, A4), (A4, A1))$, weighted γ and 0, are depicted with dotted arrows. In a solution, one of the two alternative arcs must be chosen. Choosing $(A1, A4)$ introduces a zero length cycle and forces the aircraft to spend exactly γ time units in the holding circle. Choosing $(A4, A1)$ introduces another zero length cycle and forces the aircraft to skip the holding circle.

The pairs of fixed arcs after the holding circle machine constrain each aircraft to traverse a given air segment in a window of [minimum, maximum] traversing times. For example, in Figure 1(b), $w_{(A4,A9)}$ is the minimum traversing time of air segment 4 for aircraft A and $w_{(A9,A4)}$ is the maximum traversing time but negative, so that the arcs $(A4, A9)$ and $(A9, A4)$ represent the constraints $t_{A4} + w_{(A4,A9)} \leq t_{A9} \leq t_{A4} + |w_{(A9,A4)}|$.

The sequencing of aircrafts on shared air segments is formulated with suitable alternative pairs. In Figure 1(b), aircrafts A and B share air segment 4. The sequencing is imposed by the two pairs of alternative arcs $((A4, B4), (B7, A9))$ and $((B4, A4), (A9, B7))$, weighted with the STI at the entrance/exit of the air segment. These pairs also model the non-overtaking constraint, e.g., if $(A4, B4)$ is selected from the first pair, then $(A9, B7)$ must be selected from the second pair in order to avoid a positive length cycle. In other words, if A precedes B at the entrance of air segment 4 it must precede B also at its exit.

The sequencing of aircrafts on a runway is modeled by introducing a pair of alternative arcs for each pair of aircrafts sharing the runway. In Figure 1(b), only A and C share a runway. The resulting alternative pair is $((Aout, C12), (C16, A12))$. If A precedes C, C

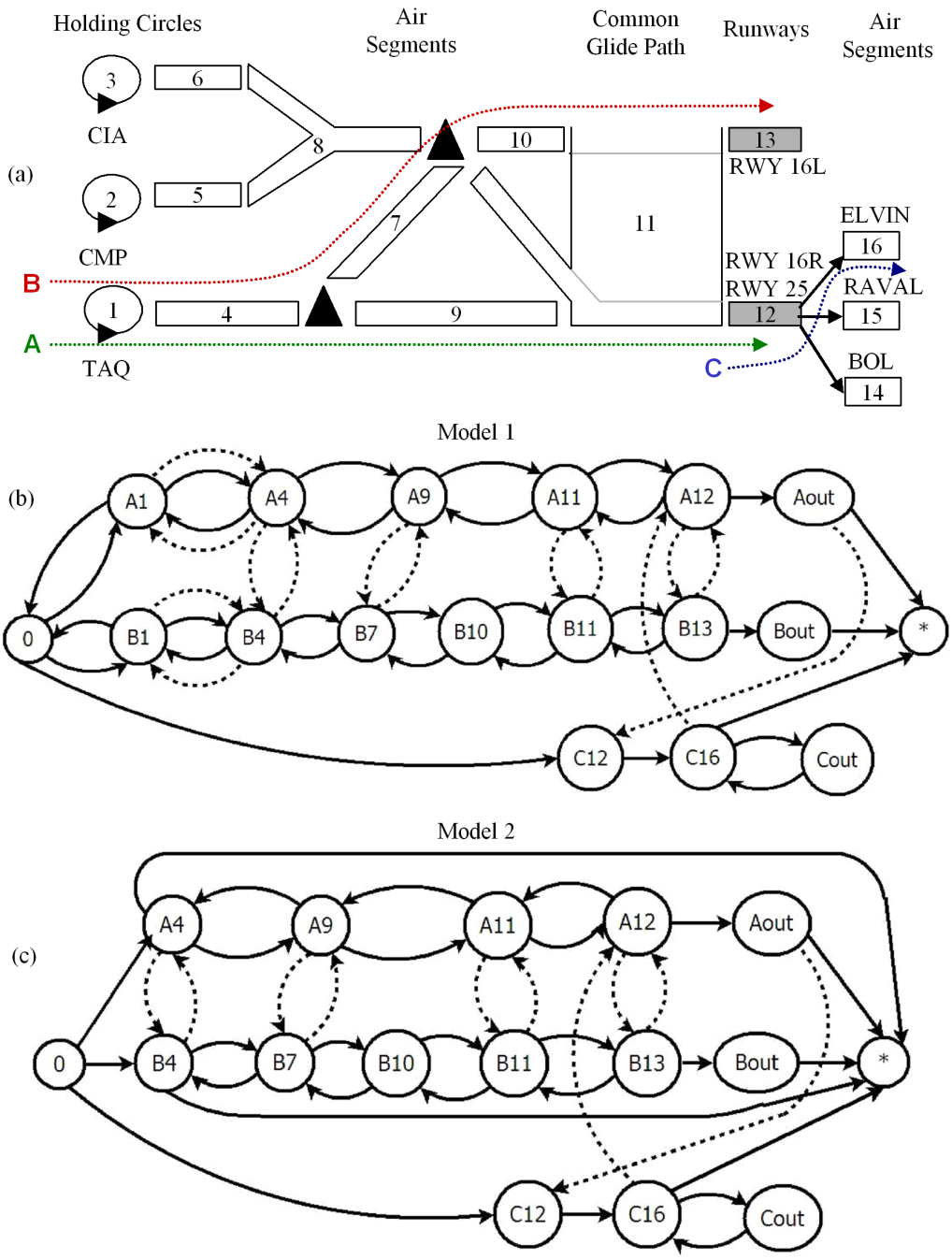

Figure 1: The FCO airport (a) and two alternative graph formulations, Model 1 with holding circles (b) and Model 2 without holding circles (c).

can take-off only after that A leaves the runway, i.e., after the starting time of operation $Aout$ plus the required STI between A and C represented by $w_{(Aout,C12)}$. Similarly, if C precedes A, A can land only after $t_{C16} + w_{(C16,A12)}$.

The delay of an aircraft is computed as follows. Departing aircrafts are considered late if leaving the airport after 10 minutes from their scheduled departure time. Arriving aircrafts are late if landing after their scheduled arrival time. The delay of an aircraft is partly due to the entrance delay in the TMA and partly due to the additional delay caused by the resolution of potential aircraft conflicts in the TMA. In this paper we minimize this second part, that we call *consecutive delay*. Let α_{ar} be the scheduled time of an aircraft a arriving/departing at/from a runway r, and let τ_{ar} be the earliest possible time of a at r compatible with its current position, i.e., computed by adding to the actual entrance time of a (ψ_a) the minimum traversing time of its route and disregarding the presence of other aircrafts. Given a solution, the total delay of a at r is the difference between its actual time t_{ar} and α_{ar}. We divide the total delay into two parts as follows. If $\tau_{ar} > \alpha_{ar}$, then $\tau_{ar} - \alpha_{ar}$ is an *unavoidable delay* that cannot be recovered by rescheduling the aircrafts. The difference between the total delay $t_{ar} - \alpha_{ar}$ and the unavoidable delay $\max\{0, \tau_{ar} - \alpha_{ar}\}$, is the consecutive delay of a at r, i.e., $\max\{0, t_{ar} - \max\{\tau_{ar}, \alpha_{ar}\}\}$. The objective of ASP is the minimization of the maximum consecutive delay over all aircrafts. In the graph, we use the arcs ingoing node $*$ to model this objective, so that $t_* - t_0$ means the maximum consecutive delay of a solution. In Figure 1(b), arcs $(Aout, *)$, $(Bout, *)$ and $(C16, *)$ are weighted with $w_{(Aout,*)} = -\max\{\tau_{Aout}, \alpha_{Aout}\}$, $w_{(Bout,*)} = -\max\{\tau_{Bout}, \alpha_{Bout}\}$ and $w_{(C16,*)} = -\max\{\tau_{C16}, \alpha_{C16}\}$.

In our computational experience, the holding circle constraints make the problem particularly hard to solve. We therefore developed Model 2 in which these constraints are relaxed and each aircraft can spend an arbitrary amount of time in the holding circle. However, the time spent in the holding circle is penalized in the objective function. In Figure 1(c), holding circles are therefore replaced by the arcs $(A4, *)$ and $(B4, *)$ with weights $w_{(A4,*)} = -\max\{\tau_{A4}, \psi_A\}$ and $w_{(B4,*)} = -\max\{\tau_{B4}, \psi_B\}$.

3 HEURISTIC AND EXACT ALGORITHMS

This section briefly describes the five aircraft scheduling algorithms evaluated in this paper. The First In First Out (FIFO) rule, often used in the traffic control practice, solves aircraft conflict situations by assigning each conflicting machine to the first aircraft requiring it.

The other heuristics take decisions based on global information taken from the alternative graph formulations. The Arc Greedy Heuristic (AGH) is a family of heuristic algorithms. The family is 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 solution is found or a positive length cycle is detected. The heuristics in the AGH family differ from each other for the evaluation criterion applied to choose the next arc from A. In this paper we study two evaluation criteria. Heuristic AMCC (Avoid Most Critical Completion time) chooses the pair containing the alternative arc which would cause the largest increase in consecutive delay [11] and selects its alternative arc. Heuristic AMSP (Avoid Most Similar Pair) chooses the pair with the largest sum of consecutive delays and selects the arc causing the smallest delay [12].

We next present the new Job Greedy Heuristic (JGH), also based on an iterative arc selection procedure. At each step, JGH selects all the alternative arcs involving a chosen job, so that the sequencing of the operations of this job with respect to the others is fixed. All the remaining jobs are then evaluated and the position of the one minimizing the objective function is actually fixed in the partial schedule. Note that at each step the relative order of remaining jobs with previously selected jobs is already fixed, so it only remains to decide the relative order of unselected jobs. The evaluation of an unselected job requires to solve a scheduling problem restricted to the unselected pairs between the current candidate job and the partial schedule. This restricted problem is solved by applying the two AGH heuristics described above. Once a job has been positioned in the current schedule, a new iteration begins. The procedure halts when all jobs have been positioned in the schedule or an unfeasibility is detected.

The fifth algorithm is an adaptation of the Branch and Bound (denoted as BB) algorithm introduced in [6] for a train scheduling problem and applied in this paper to the ASP. The initial upper bound is obtained by running the four heuristics proposed above. At each node of the enumeration tree, for the current selection S, the BB algorithm uses the single machine Jackson Preemptive Schedule $JPS(S)$ [9] for computing a lower bound on each air segment and on each runway and possibly pruning the enumeration tree. Static and dynamic implication techniques are also utilized to speed up the computation. A main modification of the BB algorithm concerns with the extension of the static implications to cope with the air segment machines, while only single capacity machines where considered in [6]. The same constraint propagation techniques are utilized during the execution of the four heuristics, as well as during the exact search procedure. In order to identify optimal BB configurations, we performed a preliminary set of experiments on 24 instances of the ASP. Each instance corresponds to a graph with a number of alternative pairs up to 2867. We tested several branching rules, branching schemes and search strategies of the enumeration tree. The two best BB configurations, in terms of computation time and number of branches, exhibit the following similarities. The branching rule is to choose the unselected pair with criteria AMSP and branch on this pair. The search strategy is to alternate four repetitions of the depth-first visit with the choice of a selection S with the smallest value of $JPS(S)$ among the last five generated selections. The two configurations differ for the branching scheme. In the first configuration BB branches with priority on sequencing aircrafts on the runways. In the second configuration all alternative pairs have the same priority. In both cases, AMSP is applied to choose among alternative pairs with the same priority.

4 COMPUTATIONAL EXPERIMENTS

The tested instances are based on a busy hour of aircraft arrivals and departures at the FCO international airport. We consider an entry fix for each aircraft in the TMA and the routing problem is solved off-line, so that the workload of the runways is well balanced and there is no conflict at runways when aircrafts are on time. We also fix a limit of three round trips for each aircraft in a holding circle. The experiments are executed on a processor Intel Core2 T7200 (2 GHz), 2 GB Ram and Linux operating system.

4.1 Description of the instances

Table 1 presents the main characteristics of 80 instances we use to test the scheduling algorithms. Each row reports average data over 20 instances, 10 instances for the two alternative graph models of Section II. Column 1 presents 4 time horizons of traffic prediction, ranging from 15 minutes to 60 minutes and Column 2 shows the number of arriving and departing aircrafts. The resulting graphs are described in Columns 3-5 in terms of the number of nodes ($|N|$), the fixed arcs ($|F|$) and the pairs of alternative arcs ($|A|$) .

Table 1: Perturbed traffic situations and prediction horizons

Time Horiz	Arr/Dep Aircrafts	Graph Nodes	Fixed Arcs	Altern Pairs	Max Delay	Avg Delay	Delayed Aircrafts
15	6/1	44	254	85	450	99	4
30	16/4	118	1509	640	900	115	10
45	22/7	166	2774	1226	1350	163	14
60	32/16	259	6233	2867	1800	226	24

For each time horizon, we generate 5 Uniform and 5 Gaussian (random) entrance delays in the airport area. Columns 6 and 7 of Table 1 report on the perturbation characteristics as the maximum and average entrance delays (in seconds) and the last column shows the number of aircrafts delayed at their entrance in the area under study (see Figure 1(a)).

4.2 Effects of scheduling the runways first

Table 2 shows the performance of the two BB versions, i.e., with or without scheduling first the aircrafts at the runways. Both versions are truncated after 120 s of computation. Each row presents the average behavior over the 40 instances of Model 1 (with holding circles) or Model 2 (without holding circles). Columns 1 and 2 report the type of model and the algorithmic version. Columns 3-5 present the average over all the instances of the computation time (in seconds), the number of feasible schedules and the number of optimal solutions. For the instances solved by both versions of the branch and bound, Columns 6 reports the value of the objective function (i.e., the maximum consecutive delay in seconds), while Columns 7 and 8 show the time (in seconds) and the number of iterations to compute the best solution.

Table 2: Different versions of the branch and bound algorithm

Holding Circles	Runways First	Comp Time	Feas Sched	Opt Solut	Max Delay	Time Best	Iter Best
Model1	On	89	31/40	11/40	384	26.6	28150
Model1	**Off**	80	40/40	16/40	338	17.1	20940
Model2	**On**	39	40/40	28/40	430	0.6	401
Model2	Off	55	40/40	24/40	430	3.3	4447

From Table 2 we conclude that Model 2 is easier to solve than Model 1. None of the two BB versions outperforms the other, since the best versions (in bold in Table 2)

depend on the model being used. Giving no priority to scheduling first the aircrafts on the runways allows to solve best Model 1, whereas giving priority to the runways allows to solve best Model 2. In the next subsections, when referring to BB we mean the best BB version for each model.

4.3 Branch and bound versus heuristics

Table 3: Comparison of the scheduling algorithms

Sched Algo	Comp Time	Feas Sched	Opt Solut	Max Delay	Avg Delay	Delayed Aircrafts	Total DTTS
15 min							
BB	0.3	20/20	20/20	80	16.3	3.6	193
AMCC	0.1	12/20	12/20	80	16.3	3.6	193
AMSP	0.1	12/20	9/20	94	18.3	3.6	207
JGH	0.1	10/20	10/20	80	17.9	3.8	212
FIFO	0.1	13/20	7/20	182	14.3	3.4	173
30 min							
BB	19.8	20/20	16/20	278	81.6	22.7	1116
AMCC	0.3	9/20	2/20	320	92.1	23.8	1205
AMSP	0.3	10/20	0/20	371	122.8	24.4	1184
JGH	0.3	10/20	0/20	339	104.1	23.0	1153
FIFO	0.3	10/20	0/20	540	146.9	25.3	1404
45 min							
BB	20.5	20/20	8/20	289	85.4	31.4	1643
AMCC	0.4	10/20	3/20	356	125.3	35.0	1755
AMSP	0.4	10/20	0/20	454	147.6	37.2	1853
JGH	0.5	10/20	0/20	380	128.3	36.1	1698
FIFO	0.5	8/20	0/20	721	242.2	39.3	2168
60 min							
BB	18.2	20/20	0/20	1012	297.0	59.8	2551
AMCC	0.9	10/20	0/20	1029	272.4	61.6	2471
AMSP	0.9	10/20	0/20	1069	337.0	63.2	2614
JGH	1.1	10/20	0/20	1018	347.3	59.4	2638
FIFO	1.3	5/20	0/20	1912	803.7	57.2	2850

Table 3 presents the average results for the four time horizons of Table 1 (15, 30, 45 and 60 minutes) and the five algorithms of Section III (BB, AMCC, AMSP, JGH and FIFO). Columns 2 to 4 of Table 3 show the average results obtained for all the 20 instances of each time horizon in terms of the computation time (in seconds), the number of feasible schedules and the number of optimal solutions. For BB the computation time refers to the time to compute the best solution. In order to obtain a fair comparison of the results, the values of the last four columns are averaged over those instances for which a solution is found by all the five algorithms. Columns 5–8 report the maximum and average consecutive delays (in seconds), the number of delayed aircrafts, and the total increase of travel time spent by all aircrafts in the TMA besides their minimum traversing

time (Delta Travel Time Spent (DTTS), in seconds). For an aircraft a arriving/departing at/from a runway r, DTTS is equal to $t_{ar} - \tau_{ar}$. The last indicator is an interesting factor for energy consumption reasons. The smaller is the DTTS of an aircraft in the TMA, the smaller is its energy consumption.

It follows from Table 3 that the heuristics are very fast but often fail in computing a solution, due to the inherent complexity of the ASP. On the other hand, BB is always able to find a good solution within a few seconds. For time horizons of 30 and 45 minutes, BB outperforms all the other algorithms for the given performance indicators. As for the heuristics, AMSP is the most robust in terms of feasible schedules found. AMCC and JGH are the most performing in terms of consecutive delays while FIFO shows the largest consecutive delays for time horizons of 60 minutes.

4.4 Limits of the branch and bound algorithm

We now study the optimality gap of BB. We classify an instance as *open* if BB is not able to prove optimality within 120 s. From Table 3 it follows that there are 4 open instances for the time horizon of 30 minutes, 12 open instances for 45 minutes and all the 20 instances for 60 minutes. We next try to close these 36 open instances by enlarging the time limit of BB up to 7200 s.

Table 4 reports the performance of BB on the 36 open instances. Each instance is coded in Column 1 with the following *A*-*B*-*C*-*D*-*E* fields: Under A we denote the number of delayed aircrafts at the entrance of the TMA; B indicates the distribution type: Uniform (U) or Gaussian (G); C represents the average entrance delay in seconds; D indicates the time horizon of traffic prediction, from 15 to 60 minutes; E represents the model used for the ASP.

Columns 2 of Table 4 shows the lower bound, Column 3 reports the best solution found by AMCC, AMSP, JGH and FIFO if any, Column 4 and 5 present the final solution found by BB after 120 s and 7200 s of computation time, respectively. An asterisk in Column 5 indicates proven optimality of the solution. Finally, Columns 6 and 7 show the computation times (in seconds) to reach the best solution and to terminate the algorithm, respectively.

Out of the 16 open instances with time horizons up to 45 minutes, BB is able to close 12 instances within the two-hour time limit and to improve the solutions at 120 s for other 2 instances. For 7 instances, the optimal solution is the same obtained at 120 s. All the 20 instances with the time horizon of 60 minutes remain open after 2 hours, but BB is able to improve the solutions at 120 s for 4 instances. We finally observe that, Model 2, though simplified, enables BB to solve a larger number of instances to optimality within 120 seconds with respect to Model 1.

5 CONCLUSIONS AND FUTURE WORK

This paper presents heuristic and exact algorithms for solving two alternative graph formulations of the ASP. Computational results for the FCO airport demonstrate the effectiveness of our branch and bound algorithm that is able to compute good solutions in limited time for instances up to 60 minutes of traffic prediction. Optimal/near-optimal solutions are found in a few seconds of computation for most of the instances and BB

Table 4: Open instances

Perturbation Identifier	Lower Bound	Solut Init	Solut 120s	Solut Final	Time Best	Time Tot
10-U-237.7-30-1	308	-	537	420*	228.6	236
10-U-258.2-30-1	351	-	528	388*	336.6	456
10-U-253.1-30-1	351	-	419	419*	46.7	233
8-G-216.9-30-1	193	-	387	335*	480.6	1766
14-U-342.8-45-1	328	-	676	420*	2059.4	2088
14-U-355.8-45-1	255	-	420	420	32.0	7200
14-U-375.3-45-1	259	-	691	363	3252.0	7200
14-U-356.7-45-1	299	-	772	653	569.4	7200
14-U-356.2-45-1	306	-	781	420*	1391.4	4391
14-G-328.8-45-1	170	-	244	244*	31.7	908
14-G-334.4-45-1	180	-	227	227*	19.6	269
14-G-323.1-45-1	217	-	273	273*	25.8	361
14-G-321.7-45-1	167	-	265	265	78.0	7200
14-G-298.9-45-1	237	-	274	274*	16.4	183
24-U-467.5-60-1	918	-	1245	1177	146.7	7200
24-U-466.9-60-1	910	-	1207	1207	14.6	7200
24-U-477.1-60-1	955	-	1230	1230	7.7	7200
24-U-467.6-60-1	906	-	1253	1169	1176.0	7200
24-U-474.7-60-1	932	-	1183	1183	129.4	7200
22-G-431.5-60-1	708	-	1074	1074	6.4	7200
22-G-428.4-60-1	708	-	1098	1098	114.2	7200
22-G-431.2-60-1	708	-	972	967	492.5	7200
22-G-441.2-60-1	713	-	1135	1132	655.3	7200
22-G-437.5-60-1	770	-	1061	1061	6.2	7200
14-U-359.0-45-2	299	447	376	376*	0.7	189
14-U-358.2-45-2	341	443	384	384*	0.5	568
24-U-464.4-60-2	963	1211	1211	1211	0.6	7200
24-U-462.8-60-2	939	1167	1167	1167	0.6	7200
24-U-470.2-60-2	954	1187	1187	1187	0.8	7200
24-U-477.1-60-2	952	1180	1180	1180	1.2	7200
24-U-460.9-60-2	890	1103	1103	1103	0.6	7200
22-G-443.9-60-2	766	1016	1016	1016	0.8	7200
22-G-424.2-60-2	708	932	932	932	0.6	7200
22-G-442.6-60-2	725	945	945	945	0.8	7200
22-G-435.9-60-2	767	987	987	987	0.8	7200
22-G-438.6-60-2	708	932	932	932	0.8	7200

often outperforms the heuristics, including a practical scheduling technique.

Further research is still needed in order to develop an on-line decision support system for air traffic control at TMAs. To this aim, the scheduling models and algorithms presented in this paper should be part of the system core. Other research directions concern with the development of algorithms for computing optimal aircraft routes and speeds.

References

[1] Adacher, L., Pacciarelli, D., Paluzzi, D., Pranzo, M. (2004) Scheduling arrivals and departures in a busy airport. Preprints of the 5th Triennial Symposium on Transportation Analysis, Le Gosier, Guadaloupe.

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

[3] Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M., Abramson, D. (2000) Scheduling aircraft landings – The static case. *Transportation Science* **34 (2)** 180–197.

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

[5] CENSIS (2008) Rapporto sulla situazione sociale del Paese. Research Report **42**, Fondazione CENSIS, Roma, Italia (in Italian).

[6] 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.

[7] Ernst, A.T., Krishnamoorthy, M., Storer, R.H. (1999) Heuristic and exact algorithms for scheduling aircraft landings. *Networks* **34 (3)** 229–241.

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

[9] Jackson, J.R. (1955) Scheduling a production line to minimize maximum tardiness. Research Report **43**, Management Science Research Project, University of California, Los Angeles, USA.

[10] 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.

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

[12] Pranzo, M., Meloni, C., Pacciarelli, D. (2003) A new class of greedy heuristics for job shop scheduling problems. *Lecture Notes in Computer Science* **2647** 223–236.

[13] Psaraftis, H.N. (1980) A dynamic programming approach for sequencing identical groups of jobs. *Operations Research* **28 (6)** 1347–1359.

[14] Soomer, M.J., Franx, G.J. (2008) Scheduling aircraft landings using airlines' preferences. *European Journal of Operational Research* **190 (1)** 277-291.

[15] Venkatakrishnan, C.S., Barnett, A., Odoni, A.M. (1993) Landings at Logan airport: describing and increasing airport capacity. *Transportation Science* **27 (3)** 211–227.