4D-trajectory deconfliction through departure time adjustment
Nicolas Barnier, Cyril Allignol

To cite this version:
Nicolas Barnier, Cyril Allignol. 4D-trajectory deconfliction through departure time adjustment. ATM 2009, 8th USA/Europe Air Traffic Management Research and Development Seminar, Jun 2009, Napa, United States. pp xxxx. hal-00938231

HAL Id: hal-00938231
https://hal-enac.archives-ouvertes.fr/hal-00938231
Submitted on 3 Apr 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
4D-Trajectory Deconfliction Through Departure Time Adjustment

Nicolas BARNIER
ENAC
7, Av. Édouard Belin - BP4005
F31055 Toulouse Cedex 4
barnier@recherche.enac.fr

Cyril ALLIGNOL
DSNA/DTI
7, Av. Édouard Belin - BP4005
F31055 Toulouse Cedex 4
allignol@tls.cena.fr

Abstract—As acknowledged by the SESAR programme, current ATC systems must be drastically improved to accommodate the predicted traffic growth in Europe. In this context, the Episode 3 project aims at assessing the performance of new ATM concepts, like 4D-trajectory planning and strategic deconfliction.

One of the bottlenecks impeding ATC performances is the hourly capacity constraints defined on each en-route ATC sector to limit the rate of aircraft. Previous works were mainly focused on optimizing the current ground holding slot allocation process devised to satisfy these constraints. We propose to estimate the cost of directly solving all conflicts in the upper airspace with ground holding, provided that aircraft were able to follow their trajectories accurately.

We present a Constraint Programming (CP) model of this large scale combinatorial optimization problem and the results obtained with the FaCiLe constraint library. We study the effect of uncertainties on the departure time and estimate the cost of improving the robustness of our solutions with the CATS simulator. Encouraging results were obtained without uncertainty but the costs of robust solutions are prohibitive. Our approach may however be improved e.g. with a prior flight level allocation and the dynamic resolution of remaining conflicts with one of CATS’ module.

Keywords: Slot Allocation, Conflict Resolution, Constraint Programming

I. INTRODUCTION

In an already saturated European sky, the predicted growth of air traffic volume urges to improve Air Traffic Management (ATM) efficiency, as attested by the ACARE Strategic Agenda 2 [1] and the European Single Sky programme SESAR. Current ATM optimization strategies, like reducing the size of control sectors or the distance of separation (RVSM, P-RNAV), seem to have reached the structural limits of the system, while the automation of Air Traffic Control (ATC) has known few significant improvements over the last decades [2].

In this context, the European Commission has launched the Episode 3 [3] research project to assess the concepts studied within SESAR definition phase. Among the key concepts identified to meet SESAR performance objectives, the planning of 4D-trajectories would allow to increase en-route capacity, while preserving the current level of safety. One of the goals of the WP4 of Episode 3 is to estimate how such regulations could benefit strategic deconfliction schemes over the current Air Traffic Flow Management (ATFM) process.

Currently, the Central Flow Management Unit (CFMU) in Brussels is in charge of optimizing the traffic by, among other strategic or tactical measures, delays the departure slots for the flights involved in overloaded en-route sectors. The purpose of this ground holding scheme is to respect the en-route capacity constraints provided by each ATC Centre (ATCC) as a number of aircraft per hour, according to their daily schedule. Former studies like [4], [5] aimed at improving this slot allocation over the greedy algorithm used at the CFMU. However, one of the limitations of this regulation model is that the definition of sectors capacities (hourly rate of aircraft entering the sector) is poorly related to the complexity of the traffic with respect to the controllers workload, as assessed by [6].

Instead of trying to satisfy en-route capacity constraints, we propose to directly solve the potential conflicts occurring between any two intersecting trajectories with departure time adjustments. A single delay would be associated with each flight such that all potential conflicts occurring above a given flight level would be avoided. This very fine grain model would of course
generate much larger constraints sets than the macro-
scopic (at the sector level) capacitated ones, but would
garantee conflict-free trajectories all along the flight
path... provided that aircraft were able to scrupulously
follow their predicted route in the four dimensions.

Obviously, the latter hypothesis is far from being
met nowadays, but the accuracy of Flight Management
Systems will be a crucial issue for future ATFM and
ATC systems, as advocated by [7] and acknowledged
by the Airbus-driven “Technological Enablers” WP6 of
Episode 3. Nevertheless, we believe that our approach
may reduce air traffic complexity by “deconflicting” it
in advance. The remaining conflicts due to deviation
from the flight plan (or occurring in the lower airspace)
would then be dynamically solved either by automated
resolution systems as proposed by [8], [9], or by more
standard ATC procedures.

Several optimization paradigms are being evaluated
for this purpose, namely meta-heuristics, local search and
Constraint Programming (CP). We will focus here on
the CP approach as it offers to obtain proved bounds on
the maximal delays needed to solve the conflicts, which
can be used to draw conclusions on the feasibility
of this kind of regulations. Moreover, CP is a technology
of choice for implementing such preliminary work, as
it allows to easily refine the problem by adding new
constraints (e.g. connection constraints between flights
using the same aircraft) and to experiment with various
search strategies without changing the rest of the model.

In the following sections, we first briefly introduce
ATC and ATFM in Europe, focusing on ground holding
policies and related research projects. Then we describe
our model of a conflict-free slot allocation, starting
by the details of the constraints generation and search
strategy. Next, our first results on instances of the French
Traffic are presented, as well as the effect of small
takeoff time uncertainties. We end with planned further
works to enhance the approach before concluding.

II. CONTEXT AND RELATED WORKS

A. ATC and ATFM

Air Traffic Control (ATC) is a ground-based service
provided to ensure the safety and efficiency of the flow
of aircraft. The first goal of ATC is to maintain aircraft
separated: outside Terminal Areas (TMA) around air-
ports, two aircraft should remain distant from each other
at least by 5 NM horizontally and 1000 ft vertically, as
illustrated by the safety volume of figure 1.

The overall system currently implemented in Europe
to achieve this goal can be conceptually divided in

Fig. 1. Vertical and horizontal separation. Another aircraft cannot
be inside the cylinder at the same time.

several layers or filters with decreasing time horizon with
respect to the flight date of the traffic concerned:

1) Strategic (several months), ASM (Air Space
   management): design of routes, sectors and procedures
   (e.g. reduced separation RVSM since 2002, Area
   Navigation (RNAV) with fictive beacons...).

2) (Pre-)Tactical (a few days to a few hours), ATFM:
   ATC Centres opening schedules define hourly ca-
   pacities of each open sectors (or groups of sec-
   tors). To respect these capacity constraints, the
   CFMU computes and updates flow regulations and
   reroutings according to the posted flight plans and
   resulting workload excess.

3) Real time (5/15 min), ATC: surveillance, coordi-
   nation with adjacent centres, conflict resolution by
   various simple manoeuvres (heading, flight level,
   speed) transmitted to the pilots.

4) Emergency (less than 5 min), safety nets: ground-
   based (Short Term Conflict Alert, Minimum Safety
   Altitude Warning) and airborne (Traffic Alert and
   Collision Avoidance System, Ground Proximity
   Warning System).

We will focus in the following section on the kind of
regulations performed by the CFMU by postponing the
takeoff of aircraft.

B. Ground Delays

As aircraft obviously cannot be paused while airborne
whenever the traffic complexity becomes too high to be
safely handled by a controller, one of the simplest way
to leverage ATC workload is to postpone the takeoff
of aircraft\footnote{Note that flights might be delayed for other reasons than en-route
capacity violation, like bad weather or equipment failures.}. This kind of measure is however quite
unpopular among airlines, as it can be very costly and
may propagate in terms of missed correspondances and
aircraft rotation (see [10]), so the delays should be
minimized as much as possible.
1) Satisfying En-Route Sectors Capacity Constraints: The aim of CFMU regulations is to maintain the number of en-route aircraft entering a given subset of sectors below some bound over given time periods (usually one hour), according to the constraints declared by experts (FMP) in each ATCC for the day of traffic. The CFMU experts first identify the overloaded sectors and responsible flows with the PREDICT tool, then compute a slot allocation as ground delays for the involved flights with the CASA tool (cf. [11]).

CASA is able to take into account many operational constraints and updates to optimize its allocation process, but the algorithm used has greedy properties and thus cannot guarantee to find a correct solution (which satisfies all the constraints) or an optimal one. CP technology has been applied with good results to prove and optimize the allocation process with a relaxed model [4] or to smooth the resulting load profiles [5] with a tighter model.

However, traffic complexity is very hard to define precisely, and sector capacities, expressed as a maximum number of aircraft entering the sector over a given time period, does not take into account many parameters relevant to accurately represent the performance of ATC. Observed actual capacities, as well as merging and splitting subset of sectors, symptomatically present very different profiles than the predicted ones.

To overcome this issue, recent works such as [12] use a much more precise and complex workload CP model to dynamically balance the traffic over the sectors of an ATCC in the upper airspace. Other works, like [13] uses CP technology as well to optimize the ATCC opening schedules to match the predicted traffic more closely, or even attempt to redesign airspace sectorisation with better balancing like [14].

2) Solving the Conflicts: One of the key ATM operational concept of the SESAR programme that Episode 3 should validate is the design of conflict-free 4D-tubes within crowded airspace (whereas separation could be delegated to aircraft in less dense areas). So instead of only respecting sectors capacities macroscopically, we propose to evaluate the cost of precisely solving all potential conflicts, only by ground holding, while minimizing the worst allocated delay to maintain equity among airlines.

Optimality proofs for the overall sum of the delays can be exponentially harder than our max criterion, and therefore out of reach for such large problems. However, our search strategy will focus on maintaining the overall amount of delay as low as possible, while the use of CP technology will provide proved maximal delay bounds that other optimization techniques (e.g. local search or meta-heuristics) cannot produce.

This conflict-free model will of course yield much larger problem instances as all the conflicting trajectories intervals above a given flight level will be taken into account as constraints. The resulting problem is intrinsically disjunctive as for each potential conflict between two flights $i$ and $j$, either $i$ must precede $j$ or $j$ precede $i$ at each pair of points concerned (see section III-B).

Other approaches have been presented to solve conflicts in real-time, automating the task of controllers. Some of the most promising ones are centralized techniques that compute simple horizontal or vertical manœuvres [15] and small speed adjustments as proposed by the ERASMUS project [9]. These solvers, based around a meta-heuristic (Genetic Algorithm), can take uncertainties on ground and vertical speed into account and repeatedly compute solutions for a sliding time window.

III. CONFLICT-FREE SLOT ALLOCATION

A. Conflict Detection

Our data are provided by the CATS\textsuperscript{2} simulator [16], which takes as input all filed flight plans concerning the French airspace for a given day of traffic and the relevant airspace configuration (sectors, waypoints...), and outputs the corresponding 4D-trajectories. Trajectories are sampled with a 15 s time step, which is the largest interval to guarantee that at least two points of the trajectories of facing aircraft at the highest possible speed will be closer than one separation norm, i.e. even the shortest conflicts will be detected.

![Fig. 2. Conflicting Points Detection](image)

Trajectories are then probed pairwise for potential conflicts, taking the maximal allowed delay into account. The separation norm is thus tested for each pair of points

\textsuperscript{2}The Complete Air Traffic Simulator developed at DSNA/DTI.
of the two probed trajectories (up to \( p = 900 \) points per trajectory for up to \( n = 9500 \) flights in \( \mathcal{O}(n^2p^2) \)) as illustrated on figure 2 in the horizontal plane.

Though the maximal allowed delay can be seen as a parameter of the search algorithm only, it also affects the conflict detection. Actually, when the maximal allowed delay is increased, the size of the problem grows as well, as more and more flights will tend to be in potential conflict. Ultimately, if a 24 h-delay will be allowed, the conflict detection could be done regardless of time, as any two geometrically conflicting trajectories would generate a constraint. So, whenever a particular instance has been proved inconsistent, it has to be generated again with higher values of the maximal delay, which will capture later potential conflicts on the trajectories pairs.

Operationally, flights originating outside the Eurocontrol countries cannot be delayed, so their delay variable will be fixed to 0 in our constraint model, reducing the number of variables but tightening the constraints as well and offering less opportunities for optimization. Constraints corresponding to conflicts occurring between two such flights will of course be discarded as we cannot delay the flights to solve them. Such remaining conflicting cases would have to be taken care of by other ATC or ATFM techniques that we will not address in this study.

### B. CP Model

1) **Conflicts Constraints:** To compute the constraints of our model, the trajectories are pairwise probed for couples of conflicting points. Given a flight \( i \), we note the input data:

- \( \{p_i^k\} \) the chronologically ordered sequence of the 3D-points of its trajectory;
- \( t_i^k \) the time at which the flight will be at point \( p_i^k \), should it not be delayed.

We define a set \( D \) of decision variables:

\[
D = \{\delta_i, \forall i \in [1, n]\}
\]

of finite domain \([0, \text{max\_delay}]\) that represent the delay associated with each of the \( n \) flights, and we will describe our model using the following auxiliary variables:

- \( \theta_i^k = t_i^k + \delta_i \) the date at which flight \( i \) will be at point \( p_i^k \) if it is delayed by \( \delta_i \);
- \( d_{ij} = \delta_j - \delta_i \) the difference of the delays of flight \( j \) and \( i \).

For any geometrically conflicting points \( p_i^k \) and \( p_j^l \) such that the separation norm is violated (\( d_h \) being the distance in the horizontal plane and \( d_u \) in the horizontal plane):

\[
d_h(p_i^k, p_j^l) < 5 \text{NM} \quad \text{and} \quad d_u(p_i^k, p_j^l) < 1000 \text{ft}
\]

we must temporally ensure that:

\[
\theta_i^k \neq \theta_j^l
\]

which can be rewritten with the difference variables \( d_{ij} \):

\[
d_{ij} \neq t_i^k - t_j^l
\]

Starting at the first such point \( p_i^k \) that conflicts with a point of flight \( j \), we take into account the whole continuous segment of trajectory \( j \) conflicting with \( p_i^k \):

\[
\{p_j^l, \forall l \in [l_k, l_{k+r}]\}
\]

for some \( r \), and we impose that:

\[
d_{ij} \not\in \{t_i^k - t_j^l, \forall l \in [l_k, l_{k+r}]\}
\]

\[
d_{ij} \not\in [lb^k, ub^k]
\]

with \( lb^k \) and \( ub^k \) being respectively the lower and upper bound of the set of \( t_i^k - t_j^l \).

If the next point \( p_i^{k+1} \) of the trajectory of flight \( i \) conflicts with a further segment of flight \( j \), we will obtain another forbidden segment for \( d_{ij} \):

\[
d_{ij} \not\in [lb^{k+1}, ub^{k+1}]
\]

taking part in the same potential conflict. To ensure separation we must then impose:

\[
d_{ij} \not\in [\min(lb^k, lb^{k+1}), \max(lb^k, lb^{k+1})]
\]

as the conflicting segments of flight \( j \) overlap.

So if we take into account all the successive points of flight \( i \), starting at \( p_i^k \), that conflict with overlapping segments of flight \( j \), up to some last point \( p_i^{k+r} \), with \( lb_1 = \min\{lb^{k+u}, u \in [0, s]\} \) and \( ub_1 = \max\{ub^{k+u}, u \in [0, s]\} \) being the overall lower and upper bounds of the corresponding forbidden intervals for \( d_{ij} \), we can define the first conflict between flights \( i \) and \( j \):

\[
d_{ij} \not\in [lb_1, ub_1]
\]

Note that we take as parameters of the problem instance the maximal allowable delay \( \delta_i \in [0, \text{max\_delay}] \), therefore the domain of \( d_{ij} = \delta_j - \delta_i \) is the interval \([-\text{max\_delay}, \text{max\_delay}] \). We simply discard the conflict whenever \( ub_1 < -\text{max\_delay} \) or \( lb > \text{max\_delay} \).

A pair of flights may conflicts several disjoint times over their entire trajectories (as illustrated on figure 3), so several such disjoint intervals may be forbidden for
the difference of their delays. For two flights $i$ and $j$ conflicting $\sigma$ times over their entire trajectories:

$$d_{ij} \not\in [lb_1, ub_1] \cup \cdots \cup [lb_\sigma, ub_\sigma]$$

or, rewritten as a disjunctive constraint over the decision variables:

$$(-w \leq \Delta_j - \Delta_i < lb_1) \lor$$

$$(ub_1 < \Delta_j - \Delta_i < lb_2) \lor \cdots \lor$$

$$(ub_\sigma < \Delta_j - \Delta_i \leq w)$$

provided that $lb_1 > -w$ and $ub_\sigma < w$, otherwise the first or last part of the disjunction is removed.

The cost of a solution is then defined as:

$$\text{cost} = \max \{\delta_i, \forall \delta \in [1, n]\}$$

2) Further Instance Processing: The takeoff and landing part of trajectories are truncated around airports within a given radius (usually 10 NM) as the traffic is considered handled with specific procedures by the TMA control services in these zones.

After the computation of the conflict constraints, the whole instance is scaled down to a more reasonable time step (e.g. 1 min) than the 15 s used during conflict detection, ensuring that the original forbidden intervals are strictly included in the scaled ones.

Moreover, the flight level of the detected conflicts can be filtered, for example to only take into account conflicts occurring within the upper airspace (from FL290 and above). The minimal and maximal altitude of each conflict is recorded during the detection stage and a conflict is discarded if it entirely occurs below or above the specified airspace slice.

We allow as well to filter the time interval during which the conflicts may occur, taking the time bounds of the allowable delay into account. Any conflict strictly occurring outside the given time interval is then discarded.

Eventually, all the flights that do not have any conflict with any other flight are withdrawn from the instance.

3) Conflict Extension: To improve the robustness of our solutions towards uncertainty on the departure times of the flights, we add an extra parameter $ext$ that extends conflicting intervals by a fix amount of time. Such an extension of $ext$ minutes stretching each end of a conflict will allow to manage uncertainties of $\pm \frac{ext}{2}$ minutes on the departure slot (see section IV-D), at the price of an increase in the cost of the solutions. We expect that this extension scheme may as well be able to diminish the effects of other sources of uncertainty (e.g. vertical and ground speed) regarding the number of remaining conflicts.

C. Search Strategy

The constraints of the problem are reminiscent of the disjunctive mutual exclusion constraints modelling scheduling problems. At a coarse grain, we could consider each conflicting area as a machine on which to process two tasks of different lengths (depending on the speed of the aircraft). Several conflicts along a trajectory could be seen as the ordered tasks of a given job, as in the Jobshop Scheduling Problem (JSP).

However, the comparison does not hold much further. First, the time intervals between any two conflict tasks of the same trajectory is fixed, as only one delay variable is associated with each flight (unlike the JSP where all tasks are only related with precedence constraints). Second, to consider a potential conflict in three dimensions only, as the transitive closure of the overlapping conflicting segments, with task lengths proportional to the time spent by the aircraft within the area, is misleading. In this setting, the conflict associated with two catching-up flights on the same route would be the entire trajectory, preventing them from being airborne at the same time! Obviously, our model is much more precise and allows two aircraft on the same route being only separated by 5 NM. Third, the number of “conflict machines”, if not quadratic in the number of “flight jobs” as it could ultimately grow for arbitrary instances, is quite huge anyway as shown on figure 5.

Nevertheless, the branching scheme of our search strategy is inspired by standard scheduling techniques,
because the essentially disjunctive nature of our problem shares some issues with scheduling ones. Trying to start the search by labelling the delay variables $\delta_i$ would be highly inefficient, because the constraints are expressed over the differences $d_{ij}$. Much more filtering is obtained by feeding the propagation of the arithmetic constraints with new domain bounds for the $d_{ij}$ auxiliary variables.

Similarly to some scheduling branching schemes, where tasks performed on the same machine are ordered pairwise (either task $A$ precedes task $B$ or $B$ precedes $A$), we either add the constraint $d_{ij} < lb$ or $d_{ij} > ub$ in the case of a single conflicting interval. If there are several holes in the domain of $d_{ij}$, branching is repeated with the bounds of the remaining holes. The variable $d_{ij}$ with highest sparsity, i.e. the smallest ratio between the domain size and the difference of the domain bounds, is chosen first for branching.

To compensate for the cost being defined as the maximal delay only, disregarding the total amount of time, we choose to branch first within the $d_{ij}$ interval corresponding to the minimum potential increase for its delay variables $\delta_i$ and $\delta_j$. Such an interval would be the closest to 0. Whenever the search backtracks over such a decision, this interval is discarded and we branch on the next one recursively.

When all conflicts are ordered and there is no more holes in the domain of the $d_{ij}$, we start labelling the decision variables $\delta_i$ with a standard dom/deg selection heuristic and the values closest to 0 are probed first so as to reduce the total amount of delay.

After the first solution is found, the branch and bound algorithm then proceeds by dichotomy on the cost domain to find the optimal solution with respect to minimization of the maximal allocated delay.

IV. RESULTS

We have implemented this CP model with the FaCiLe library [17] and obtained the following results on various day of traffic in 2007, with up to 9500 flights and 600 000 intersecting pairs of trajectories taken into account for the largest instances we could optimally solve. About 10% of the flights are non-European flights, their delays will therefore be fixed to 0 as aforementioned.

A. Data Processing

The resulting constraint graphs typically exhibit only one single large connected component of maximal degree that can be greater than 300, i.e. a single flight may conflict with more than 300 other flights. Large cliques can also be found, as large as involving more than 60 flights, which indicates the presence of very entangled and dense traffic areas. The hardness of the conflict constraints is quite unevenly distributed, with two peaks for respectively very small and very large forbidden intervals.

We mainly tune the size of our instances by choosing the minimal flight level for which the conflicts are taken into account, aiming at the upper airspace (above FL290) where most of the cruising traffic occurs. However, we were able to optimally solve one instance with more than 6 600 flights and 630 000 conflicting pairs, taking all conflicts into account for the whole day, regardless of the flight level. Figure 4 shows the number of flights of our instances as a function of the minimal FL. The plots present two parts, one very steep from the maximal FL to FL300, then flights add up at a slower rate in the lower airspace – note that plots are labelled with their date and the type of routes (standard or direct), e.g. “070123s” for the day of traffic on the 23rd of january 2007 with standard routes.

The number of conflicting pairs is not quite quadratic with the number of flights, as mentioned in section III-B and shown on figure 5, but is quite huge anyhow, reaching 630 000 for our largest instance as aforementioned.

B. Numerical Results

We were systematically able to obtain optimal solutions within affordable computation times for all the instances where the 4 GB memory of our Core 2 Duo at 2.4 GHz were not exhausted. However, the search strategy exposed in section III-C was not always the best, depending on the kind of the instances.

As shown in figure 6, small instances are solved in a few seconds whereas the biggest ones could take almost
one minute, growing only quadratically with the number of flights. We plan to address larger instances, hopefully European ones, on a computer with more memory.

However, the cost of this conflict-free slot allocation can be quite high for the busiest days (our worst case is 182 min above FL350), but may be more reasonable (around 60-90 min) for less crowded days. Figure 7 shows that the cost grows steadily for small instances (i.e. at high minimal FL), but jumps as soon as we add the main flows of traffic around FL350 for bigger instances. The optimal cost then seems to be stable for larger instances, triggered only by the flights added around FL350.

The corresponding overall delay sum (figure 8) and percentage of delayed flights (figure 9) exhibit of course a more steady behaviour, dramatically increasing with the largest instances only. Even if we do not optimize these criteria, the effectiveness of our search strategy can be observed as our figures remain within typical CFMU bounds – note that Eurocontrol reported a mean delay of 25-30 min per delayed flights and a percentage of about 40-55% delayed departures for September 2008.

C. Standard and Direct Routes

For one of the days of traffic (plots labelled “070123s” and “070123d”), we have also tested our model on direct routes, i.e. aircraft fly in straight line at the requested flight level from origin to destination, disregarding the waypoints of their flight plan (which we call standard routes). Direct routes are the ideal trajectories for airlines, with respect to operational cost, but such a traffic would be hardly controllable for human operators and ATC would have to be fully automated in this context.
However, they tend to generate constraint graphs with a lower tightness (see figure 5), and it is interesting to observe that overall delay sums are smaller than the standard ones on figure 8. Flights following standard routes tend to be on closer trajectories, suitable for the efficiency of current ATC procedures, but not using airspace to its full capacity. The max cost can be greater with direct routes though, depending on the day of traffic, as observed on figure 7.

**D. Robustness Towards Uncertainty**

Our model and algorithm were validated by taking the generated solutions as input delays for the CATS simulator. We observed that the few remaining conflicts occur below the chosen minimal flight level only, or involve flights originating outside the Eurocontrol area, which cannot be delayed in our model (see section III-A).

However, this first validation does not take into account any uncertainty w.r.t. departure time or aircraft navigation. To assess the robustness of our solutions, we have added an uncertainty parameter \( err \) on departure times within CATS. Takeoffs are randomly shifted by a bounded amount of time uniformly\(^3\) chosen in the interval \([-\frac{err}{2}, +\frac{err}{2}]\).

New validation tests were carried out with various values of parameters \( err \) and \( ext \) to compensate for the uncertainties. As expected, figure 10 shows that for \( ext \geq err \), no conflict remains: all points below the \( ext = err \) dashed line on the xy-plane exhibit a conflict percentage equal to zero. Above this line, the ratio of remaining conflicts increases with \( err \) for a given \( ext \) and when \( ext \) diminishes for a given \( err \), reaching 75% for the highest point (\( err = 6 \) and \( ext = 0 \)).

However, increasing the \( ext \) parameter leads to an increase in the total amount of delay as illustrated in figure 11. The added delays can be far too costly for higher values of \( ext \), especially for large instances. So we cannot hope to solve all conflicts with this technique alone within CFMU (−5/+10 min) or even SESAR (±3 min) time slot objectives, as operational delay cost would be prohibitive. As presented in the next section, other regulation or dynamic resolution techniques may be used to overcome this issue.

**V. Further Work**

These first results are encouraging but we have only addressed so far the resolution of conflicts within the French airspace. However, in a unified European ATC context, all conflicting traffic throughout the Eurocontrol

---

\(^3\)A better approach would involve a statistical analysis to approximate the probability distribution of the discrepancy between scheduled and actual takeoff times.
countries should be taken into account. Such instances would comprise up to 30,000 flights per day. We plan to experiment with various refinements of our algorithm to address such large scale problems.

A. Prior Flight Level Allocation

To be able to address such large instances as aforementioned, while maintaining reasonable maximal delay figures, we also plan to combine our slot allocation algorithm with a prior flight level allocation, possibly using CP technology as described in [18].

This first step, computed to minimize horizontally conflicting flows by separating them vertically (trying as well to deviate as little as possible from requested FLs), is expected to deconflict the traffic in a substantial amount before time slot allocation. The optimal cost should then remain within much better bounds than with the raw traffic.

B. Sliding Time Windows

For large European instances, we also plan to adapt our algorithm to repeatedly solve slices of the problem on a limited time window \( T_w \), then to keep only the earliest part of the solution over a smaller interval \( \lambda \) and to slide the resolution window by \( \lambda \). Parameters \( T_w \) and \( \lambda \) must be carefully chosen according to the computation time of the resolution and the cumulative effects of uncertainties. Similar approaches are used for dynamic conflict resolution in the CATS solver as described in [8].

C. Resolution with CATS Modules

Various real-time resolution algorithms have already been implemented within the CATS simulator [15], [8], [9] with very good results. We would like to assess our regulation schemes further by observing the workload of these algorithms when provided with our solutions and various kind of uncertainties, and whether the automatic resolution can cope with our delays or if the resulting manoeuvres will warp our plan entirely.

VI. CONCLUSION

We have presented a new ground holding approach to solve all potential conflicts occurring above a given flight level for a day of traffic in the French airspace. Rather than trying to respect sector capacity constraints, we model each possibly conflicting situations between any two aircraft and impose adjustments of departure times to keep them separated, with the hypothesis that aircraft could precisely follow their planned 4D-trajectories.

The resulting problem size is huge, but our CP algorithm is able to reach optimal solutions for all conflicts occurring inside the upper airspace. The resulting maximal delay, overall delay sum and ratio of delayed flights can be comparable to delays allocated by the CFMU, but for the busiest days, solving all conflicts by ground delaying can be far too costly. Nevertheless, our solutions were validated with the CATS simulator, checking that no conflict under the given flight level for a delayable aircraft remains.

We have also presented a first step toward taking uncertainties into account by extending the forbidden intervals of conflicting flights. However, an extension as small as 4 min, which is able to cope only with a \( \pm 2 \) min-uncertainty on the departure time generates tremendous amounts of delays, far above SESAR performance objectives.

We plan to overcome these issues and further assess the possible outcomes of 4D-trajectory planning in the context of Episode 3 WP4 and address larger (European) instances with various techniques like combining our delay algorithm with a prior flight level allocation, repeatedly solving the problem on a sliding time windows or solving the remaining conflicts with a CATS resolution module.

REFERENCES


AUTHORS

Dr Nicolas Barnier is a lecturer and research assistant at ENAC. He graduated from ENAC as an engineer in 1997, and received a PhD in computer science in 2002 from the University of Toulouse. He is one of the authors of FaCiLe, an open source Constraint Programming library for the functional language OCaml. His research interests focus on Constraint Programming, Local Search and Combinatorial Optimisation in general.

Cyril Allignol is currently a PhD student in the Planning, Optimization, and Modelling team of the DSNA/DTI R&D domain. He received his engineer’s degrees (IEEAC in 2006) from the french civil aviation academy (ENAC) and his M.Sc. (2006) in computer science from the “Institut National Polytechnique de Toulouse” (INPT).