Air-Traffic Complexity Resolution in Multi-Sector Planning Using Constraint Programming

Pierre Flener*, Justin Pearson, and Magnus Ågren
Department of Information Technology, Uppsala University, 751 05 Uppsala, Sweden
FirstName.LastName@it.uu.se

Carlos Garcia-Avello[1], Mete Çeliktin[2], and Søren Dissing[3]
[1] Directorate of ATM Programmes, ATC Applications and Systems Division
[2] Directorate of ATM Strategies, SESAR and ATM Strategy Division
[3] Directorate of ATM Programmes, Communications Systems and Programmes
European Organisation for the Safety of Air Navigation (EuroControl), Brussels, Belgium
FirstName.LastName@eurocontrol.int

Abstract

Using constraint programming, we effectively model and efficiently solve the problem of balancing and minimising the traffic complexities of an airspace of adjacent sectors. The traffic complexity of a sector is here defined in terms of the numbers of flights within it, near its border, and on non-level segments within it. The allowed forms of complexity resolution are the changing of the take-off times of not yet airborne flights, the changing of the remaining approach times into the chosen airspace of already airborne flights by slowing down and speeding up within the two layers of feeder sectors around that airspace, as well as the changing of the levels of passage over way-points in that airspace. Experiments with actual European flight profiles obtained from the Central Flow Management Unit (CFMU) show that these forms of complexity resolution can lead to significant complexity reductions and rebalancing.

1 Introduction

The mission of the European Organisation for the Safety of Air Navigation (EuroControl) is to promote the harmonisation of the different national air-traffic-management (ATM) systems and to lead the development of the next-generation pan-European ATM system capable of handling the foreseen increase in traffic demand. It is the counterpart of the Federal Aviation Administration (FAA) of the USA.

Today, the air-traffic control operation within any control centre rests on a division of the airspace into *sectors*, that is three-dimensional, possibly concave polygonal regions of airspace that are stacked at various altitudes. Because of this fragmented mode of operation, the capacity of a control centre is limited by its sector with the minimum capacity. Indeed, controlling capacity is lost and the total capacity of a control centre could be raised by an early identification of the traffic complexity bottleneck areas and a reorganisation of the traffic patterns such that the traffic complexity is more evenly balanced between its sectors. This has triggered the development of concepts dealing with *multi-sector planning* (MSP), which is based on *traffic complexity management* (TCM) [8], where tools are developed to predict the traffic complexity over several sectors and to manage their overall complexity by anticipating peaks and proposing alternate plans. The *traffic complexity* of a sector is an estimate of the air-traffic controller (ATC) workload of that sector. Here, it is defined in terms of the numbers of flights within it, near its border, and on non-level segments within it. Indeed, each of these positions or segments of a flight requires special attention and procedures to be followed by the ATC. See Section 2.1 for more details on traffic complexity. The initial MSP TCM research and development (R&D) activities at EuroControl have focused on complexity measurement and complexity prediction [3, 4, 5].

This paper presents the first EuroControl R&D results on *complexity resolution*, that is the dynamic modification of flight profiles to reduce the predicted complexities over a given time interval of some sectors, thereby avoiding intolerable peaks of ATC workload, as well as, in this multi-sector framework, to balance the complexities of several adjacent sectors, thereby avoiding unacceptable dips of ATC workload and unfair discrepancies between ATC workloads in those sectors. We are only interested here in en-

*Currently on leave at Sabancı University in İstanbul, Turkey.

route flights in the upper airspace that follow standard routes, rather than performing free flight.

The tactical rolling-horizon scenario considered is as follows. At a given moment, suitably called now below, the (human) complexity manager queries the predicted complexities for an airspace of several adjacent sectors over a time interval that is some 20 to 90 minutes after now. Below a look-ahead of 20 minutes, there would not be enough time for the computation and implementation of the complexity-resolved flight profiles. Beyond a look-ahead of 90 minutes, there is too much uncertainty in trajectory prediction, and hence in complexity prediction. If there are ATC workload peaks, dips, or discrepancies over some time sub-interval $[m_1, \ldots, m_2]$ that warrant interference, then the complexity manager launches some complexity resolution process that suitably changes the current flight profiles over that sub-interval, whose length should be about 5 to 10 minutes. The average flight time through a European upper airspace sector is about 8 minutes, hence the proposed resolution window of 5 to 10 minutes is a trade-off between the calculation time required to find a reasonable resolution and the variability of complexity. Also, the implementation of a resolution strategy will have an impact on the evolution of the predicted complexity for later intervals. We thus came to the conclusion that a 5 to 10 minute interval may be a realistic value to start with. A minimum fraction *ff* of the number of flights planned to be in the chosen multi-sector airspace within $[m_1, \ldots, m_2]$ have to be there under the resolved flight profile as well. Indeed, complexity resolution would otherwise just re-plan a maximum of flights to be outside all those chosen sectors, at the expense of increased complexity in the adjacent sectors. This process is to be repeated around the clock approximately every 10 minutes. For this to work, the time spent on computing and implementing the complexity resolutions should not exceed those 10 minutes, and the implementation effort should be offset by the resulting complexity reductions and rebalancing among sectors.

In this work, the *allowed forms of complexity resolution* are as follows, for a flight that eventually enters the given airspace of adjacent sectors:

1. Changing the take-off time of a not yet airborne flight by an integer amount of minutes (not seconds), within the range $[-5, \ldots, +10]$. Today, if the ATM system is overloaded, the Central Flow Management Unit (CFMU) imposes slots on aircrafts of a 15 minute duration and with a $[-5, \ldots, +10]$ minute distribution along the slot time. However, a finer definition of the departure time within that slot allows a decrease in the predicted complexity peaks within the system. This is a powerful and economic means to manage traffic complexity whilst staying within flow control constraints.

2. Changing the remaining approach time into the chosen airspace of an already airborne flight by an integer amount of minutes, but only within the two layers of feeder sectors around that airspace, at a speed-up rate of maximum 1 minute per 20 minutes of approach time, and at a slow-down rate of maximum 2 minutes per 20 minutes of approach time. (Precise definitions of the notions of approach time and feeder sectors will be given in Section 2.2.) The present ATM system lets an aircraft fly its preferred speeds during the cruise phase and most of the descent phase. Typically, the first speed restriction for inbound flights happens below $10,000$ feet. There is quite some room on long flights to change the speed of some aircrafts in order to achieve a different future traffic distribution within sectors that will result in a lower traffic complexity. However, due to aircraft aerodynamics and airline cost index management, the speed control range during the cruise phase may be limited for long-haul flights but remains significant during the descent phase and interesting during the climb phase.

3. Changing the altitude of passage over a point in the chosen airspace by an integer amount of flight levels (hundreds of feet), within the range $[-30, \ldots, +10]$, such that the flight climbs no more than 10 levels per minute, or descends no more than 30 levels per minute if it is a jet, and 10 levels per minute if it is a turboprop. Aircrafts on crossing routes at the same level or in an overtake situation on the same route contribute significantly to traffic complexity. Separating the aircrafts vertically at an early stage may reduce drastically the traffic complexity perception of the controller by reducing the amount of time he needs to monitor a given pair of aircrafts. This technique is already used today to increase sector throughput and is named *level capping*. There are cost implications when changing the cruise level of a flight, but this aspect can be taken into account by reducing the levels of inbound flights before affecting outbound flights and overflights.

Many other forms of complexity resolution can of course be imagined, such as the horizontal reprofiling along alternative routes (from a list of fixed or dynamically calculated routes), or the introduction of even more time variables than just on the entry into the considered multi-sector airspace. Such additional forms of complexity resolution should only be introduced if they are warranted by additional gains and by computational feasibility.

The *objective* of the present work can now initially be stated as follows: Given a set S of adjacent sectors, given moments $now < m_1 < m_2$, and given a fraction *ff*, return a modification (according to the allowed changes above) of the profiles of the N flights that are planned at now to be inside S within the time interval $[m_1, \ldots, m_2]$ such that minimum $ff \cdot N$ of these flights are now planned to be inside S within $[m_1, \ldots, m_2]$ and such that the complexities of the sectors in S are minimised (and ideally better balanced). In practice, an allocated amount $timeOut$ of computation time is also given, and we want the best such flight profile changes that can be computed within $timeOut$ seconds.

Our *assumptions* and their justifications for realism and impact are as follows:

- We assume that times can be controlled with an accuracy of one minute. Indeed, the resolved flight profiles may have new take-off times for some of the flights originally planned to take off after now, or new approach times into the chosen airspace for flights already airborne at now, and the rest of their profiles are shifted accordingly, but the computed optimal complexity only holds if these resolved flight profiles are adhered to by the minute, which is a realistic accuracy nowadays. Semantically, the two kinds of time changes amount to a change of the *entry time* of a flight into the chosen airspace, in whose sectors the traffic complexities are actually balanced and minimised. Outside that chosen airspace, the flight profiles are of course to be updated differently according to the actual kind of time change, but our purpose here is only to *measure* the impacts of time changes within that airspace. Consequently, we will from now on only talk about entry-time changes. Although entry times are controlled with an accuracy of one minute, there are no theoretical difficulties with switching to a more fine-grained control of entry times.
- We assume that the flight time along a segment does not change if we restrict the flight-level changes over its end points to be "small", as realistically constrained in the third form of resolution above. Otherwise, we cannot shift the flight profile according to the new entry time and many more time variables would be needed, leading to combinatorial explosion.

The main *contributions* of this paper are as follows:

- We use a more sophisticated notion of air-traffic *complexity* than just the number of flights in a sector at a given moment, so that limiting the capacity of a sector is not just about limiting the number of its flights.
- We introduce air-traffic complexity *resolution* (namely minimisation and balancing) in a *multi*-sector-planning framework.
- We present the probably first *constraint program* [1, 9, 11] for such an airspace planning purpose, previous models being based on mathematical programming technology, such as [2, 10]. It is very efficient, highly readable, and is to serve as an early prototype for further experiments as well as additions and modifications of definitions and constraints. No claim is made that the designed program and the chosen underlying optimisation technology are optimal.

The rest of this paper is organised as follows. In Section 2, we introduce the necessary background information. In Section 3, we then summarise a constraint program modelling the complexity resolution problem in a multi-sector-planning framework. In Section 4, we report on the experiments we made with that program. Finally, in Section 5, we conclude, discuss related work, and outline future work.

2 Background

In Section 2.1, we define the air-traffic complexity of a sector as an estimate of the air-traffic-controller (ATC) workload of that sector. Next, in Section 2.2, we define various concepts arising for the considered forms of complexity resolution. Finally, in Section 2.3, we recall the notion of Pareto optimisation.

2.1 Air-Traffic Complexity

Let us first summarise the findings of the previous research and development work at EuroControl (and without our involvement) on air-traffic complexity measurement [3, 4, 5].

Definition 1 The complexity of a given sector s at a given moment m is based on the following terms:

- *Traffic volume*: Let N_{sec} be the number of flights in s at m.
- *Vertical state*: Let N_{cd} be the number of non-level (climbing or descending) flights in s at m. Indeed, such flights need more ATC monitoring than level flights.
- *Proximity to sector boundary*: Let N_{nsb} be the number of flights that are at most $h_{nsb} = 15$ nautical miles (nm) horizontally or $v_{nsb} = 40$ flight levels (FL) vertically beyond their entry to s, or before their exit from s, at m. Indeed, such flights require ATC interaction with the ATC of the previous or next sector.

The *moment complexity* of sector s at moment m is a normalised weighted sum of these terms:

$$C(s,m) = (a_{sec} \cdot N_{sec} + a_{cd} \cdot N_{cd} + a_{nsb} \cdot N_{nsb}) \cdot S_{norm}$$

where the *sector normalisation constant* S_{norm} characterises the airspace structure, equipment used, procedures followed, etc, of s. This is to ensure that complexity values have a relatively consistent meaning across a wide range of sectors.

The other parameters initially identified in [4], called 'data-link equipage' (indicating whether the ground-air data-link is digital or not), 'time adjustment' (necessary if the specific flight has a time constraint that will require controller action), 'temporary restriction' (the proportion of the normal capacity of the sector that is predicted to be available, considering the weather, equipment malfunction, military use of shared airspace, etc), 'potentially interacting pairs' (the number of flights that will violate horizontal or vertical separation constraints within the considered sector), and 'aircraft type diversity' (as the diversity in aircraft types has an impact on the speeds they fly, the levels they use, and the rates at which they change altitude, and thus introduces additional complexity by requiring more monitoring), are not used here. Indeed, the 'data-link equipage', 'time adjustment', and 'temporary restriction' parameters, though probably relevant, were not used in the study [5] that determined the weights and normalisation constants for the sectors of the multi-sector airspace considered in our experiments of Section 4, and this simply because there were no data to quantify the weights affecting them. Also, the 'potentially interacting pairs' parameter was used in [5], but (somewhat surprisingly) did not show a good correlation with the complexity value, as estimated by the COCA metric [6]. This has been explained by the fact that the 'traffic volume' and 'vertical state' parameters already capture this impact [5]. Finally, the 'aircraft type diversity' parameter was also used in [5], but also showed a poor correlation with the COCA complexity value, but this may be due to the limited amount of data used in the determination of the weights [5]. The resulting air-traffic complexity measure is maybe simple, but it correlates with a model of (European) ATC workload and yet it has more parameters than other such metrics actually deployed so far for complexity resolution [2, 10]. See Section 5 for a comparison with this related work.

However, as already observed in [5], moment complexity follows has a large variance, with steep rises and falls within just seconds. In order to reduce the probability that the complexity-resolved flight profile just falls into a dip of such a curve, we should define complexity so that its curve follows a smoother pattern. This can be achieved by a windows averaging technique, namely defining complexity over a given time interval rather than for a specific moment.

Definition 2 The *interval complexity* of a given sector s over a given time interval $[m, \ldots, m + k \cdot L]$ is the average of the moment complexities of s at the $k+1$ sampled moments $m + i \cdot L$, for $0 \leq i \leq k$:

$$C(s, m, k, L) = \frac{\sum_{i=0}^{k} C(s, m + i \cdot L)}{k+1}$$

where k is called the *smoothing degree*, and L the *time step* between the sampled moments.

Note that interval complexity reduces to moment complexity when $k = 0$, with the value of L being irrelevant in that case. From now on, all references to complexity are about interval complexity.

For the sake of complexity resolution, our experimentally determined good values of the parameters are $k = 2$ with $L = 210$ seconds, meaning that there are three sampled moments (namely m, $m + 210$, and $m + 420$), spanning a time interval of seven minutes. Under such values, interval complexity follows a much smoother curve than moment complexity. Any value $k > 2$ does not lead to any further significant smoothening (and would lead to impractically long computation times).

Any value of $k \cdot L$ very different from 420 seconds is not in tune with the average time flights spend on a given segment (within the airspace considered in our experiments of Section 4). Consider how the third form of complexity resolution works, in isolation from the other two forms, referring to Figure 1. For every point p_i on the planned route (drawn as a plain line) for a given flight f within the chosen airspace, there is a maximal range $[-30, \ldots, +10]$, denoted by a dashed vertical double arrow, of flight levels by which the altitude of passage of f over p_i can be changed, depending on the engine type of f and the distances to the previous point p_{i-1} and next point p_{i+1}, if any. Suppose, for $k = 2$, that the three sampled moments m, $m + L$, and $m + 2L$ fall as indicated on the horizontal time axis, namely on a climbing segment $[p_2, p_3]$, on a level segment $[p_3, p_4]$, and on another climbing segment $[p_4, p_5]$, respectively. Complexity resolution tries to level off the other two climbing segments, by making already $[p_1, p_2]$ reach the level h of $[p_2, p_3]$, or by making $[p_5, p_6]$ start only from h, if not by lowering or raising h for as many as possible of the points p_2 to p_5, depending on the lengths of $[p_1, p_2]$ and $[p_5, p_6]$. Indeed, the *sum* of the N_{cd} terms for the involved sectors would then decrease by the number of these steeper climbs that are compatible with the climbing performance of f. The dot-dashed alternative route in the figure assumes that both sampled climbing segments could be levelled off. Now, if L is too small, then several sampled

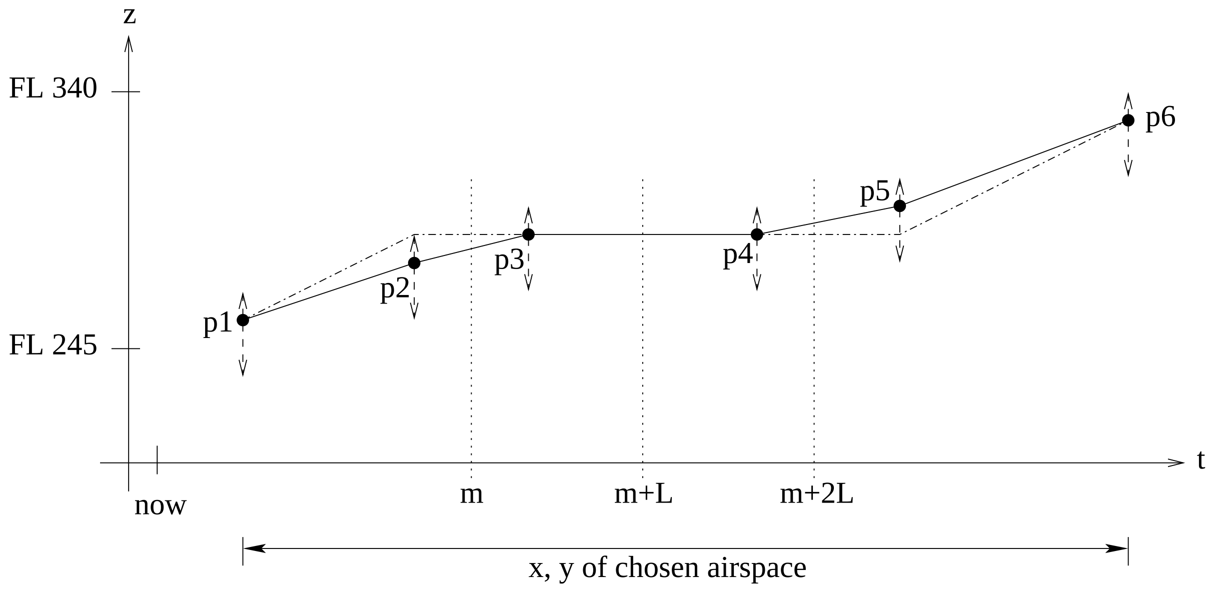

Figure 1: Planned profile (plain line) and resolved profile (dot-dashed line) that minimises the number of climbing segments for a flight at the three sampled moments m, m+L, and m+2L.

moments might fall onto the same non-level segment, thereby making a single change achieve a lot of complexity reduction. Conversely, if L is too large, then the sampled complexity values concern segments that are too far apart for their average to be meaningful.

2.2 Feeder Sectors and Approach Times

The second considered form of complexity resolution is the slowing down and speeding up of already airborne flights, but only within the two layers of sectors around the chosen multi-sector airspace, so as to delay or advance their entries into that airspace. We call those surrounding sectors the feeder sectors.

Definition 3 A *feeder sector* s of a flight f is a sector that is a neighbour of a sector of the chosen airspace, or a neighbour of a neighbour of such a sector, such that f is planned to fly through s before entering that chosen airspace.

Essentially, the time still to be spent in the feeder sectors is called the approach time, defined next.

Definition 4 For a given moment, called now, the *approach time* of a flight f that has taken off at or before now but that has not yet entered the chosen airspace is the duration that f is planned to fly within its feeder sectors between now and its entry into the chosen airspace. If f has not yet entered its feeder sectors at time now, then the approach time of f is the entire planned duration of flight within its feeder sectors; otherwise, the approach time of f is the remaining duration of flight within its feeder sectors. Formally, if a flight f with take-off time $f.timeTakeOff$ is planned to enter the chosen airspace at time $t_C[f]$ and its first feeder sector at time $t_F[f] \leq t_C[f]$ such that $f.timeTakeOff \leq now < t_C[f]$, then the approach time of f is

$$a[f] = t_C[f] - \max(now, t_F[f]) \qquad (1)$$

The restriction to two layers of feeder sectors around the chosen airspace aims at propagating only a reasonable number of updates to the upstream air-traffic control centres for each updated flight profile, as well as at leaving sufficient time to implement those updates. Note that the actual complexity will be measured over a time interval $[m, \ldots, m+k\cdot L]$, with $now < m$. Furthermore, there is too much uncertainty involved to propagate changes more than two sectors upstream and still hope the new entry time into the chosen airspace will actually be as intended. Finally, at the moment, there is no practical way for an ATC centre to modify a flight long before it reaches its own airspace.

2.3 Pareto Optimisation

Essentially, we are dealing with a multi-objective minimisation problem. We are given a number of sectors $s_1, s_2, \ldots, s_n$ and wish to minimise the vector of their complexities with respect to a resolution R:

$$\langle C_R(s_1, m, k, L), \ldots, C_R(s_n, m, k, L)\rangle$$

Pareto efficiency is a concept that originates from economics but is now a widely used concept for multi-objective minimisation problems within engineering. A vector of complexities is said to be *Pareto optimal*, or *Pareto efficient*, if no element can be made less complex without making some other element more complex. More formally, a vector

$$\langle C_{R^*}(s_1, m, k, L), \ldots, C_{R^*}(s_n, m, k, L)\rangle$$

is *Pareto optimal* for a resolution R^* if and only if there is no resolution R such that for all j we have

$$C_R(s_j, m, k, L) \leq C_{R^*}(s_j, m, k, L)$$

but for some j we have

$$C_R(s_j, m, k, L) < C_{R^*}(s_j, m, k, L).$$

In general, there is more than one Pareto-optimal solution to a problem.

One of the standard techniques for solving Pareto optimisation problems is by combining the multiple objectives into a single objective using a weighted sum:

$$\sum_{i=1}^{n} \alpha_i \cdot C_R(s_i, m, k, L)$$

for some weights $\alpha_i > 0$. Minimising the weighted sum produces a solution that is Pareto optimal. Different weights α_i produce different Pareto-optimal solutions with different tradeoffs. In practice, one often takes $\alpha_i = 1$. Although the weighted sum only guarantees to minimise convex parts of the set of Pareto-optimal points, in practice non-convex sets are seldom found. Further, in this work, we are only interested in a resolution that reduces complexity, but not in the structure of the set of Pareto-optimal points, therefore any Pareto-minimal resolution is sufficient.

We have experimented with other Pareto weights and with other (non-Pareto) optimisation criteria as well, such as minimising the worst resolved complexity among the sectors of the chosen airspace, or minimising the worst discrepancy among the resolved complexities of the sectors of the chosen airspace. However, the results were not as good, including often a poorer balancing of the resolved complexities across those sectors, or even took much more computation time to get.

3 The Constraint Program

Our constraint program is fully parameterised and constitutes an ideal vehicle for experimenting with various settings of the following parameters:

- $maxEarly$ (respectively $maxLate$) is the maximum amount of minutes that a flight can take off before (respectively after) its planned time; a typical value is 5 (respectively 10).
- $maxSlowDown$ (respectively $maxSpeedUp$) is the maximum amount of minutes that a flight can be slowed down (respectively sped up) per 20 minutes; a typical value is 2 (respectively 1).
- $maxDown$ (respectively $maxUp$) is the maximum amount of flight levels by which the altitude of a flight over a point can be decreased (respectively increased); a typical value is 30 (respectively 10).
- $maxDownJet$ (respectively $maxUpJet$, $maxDownTurbo$, and $maxUpTurbo$) is the maximum amount of flight levels that a jet (respectively turbo-prop) can descend (respectively climb) per minute; a typical value is 30 (respectively 10, 10, and 10).
- $lookahead$ is a non-negative-integer amount of minutes; a typical value is a multiple of 10 in the range $[20, \ldots, 90]$.
- now is the time, given as hour:minute, at which a resolved scenario is wanted with a forecast of $lookahead$ minutes.
- $m = now + lookahead$ is the start moment for complexity resolution.
- k is the smoothing degree, that is the number of time steps into the future from m where the (original) complexity of the considered sector is measured; a good value is 2.
- L is the length, in seconds, of those time steps; a good value for complexity resolution is 200.
- *ff* is the minimum fraction of the sum of the numbers of flights planned to be in the chosen multi-sector airspace at the sampled moments $m + i \cdot L$, for all $0 \leq i \leq k$, that have to be there in the resolved flight profile as well.
- $timeOut$ is the maximal amount of seconds that should be spent on computations before returning the currently best feasible solution.

For each flight f, for each pair (f, p) of a flight f and one of its waypoints p within the chosen airspace, for each sector s, and for each index i of a sampled moment, the constraint program determines values of the decision variables $\Delta T[f]$ (denoting the entry-time change of f into the chosen airspace), $\Delta H[f, p]$ (denoting the flight-level change of f over p), $N_{sec}[i, s]$, $N_{nsb}[i, s]$, and $N_{cd}[i, s]$, subject to the allowed forms of complexity resolution, such that the sum of the interval complexities of the chosen sectors is minimised (that is, it performs a Pareto optimisation with unit weights). See [7] for full technical details. The constraint program tries to 4D-position each flight f to be never near a boundary of any sector s, so as not to increase any $N_{nsb}[i, s]$, and never on a climbing or descending segment, so as not to increase any $N_{cd}[i, s]$. If it cannot avoid positioning a flight near a sector boundary or on a climbing or descending segment, then it prefers a sector with a low a_{nsb} or a_{cd} value, unless the *ff* parameter even allows it to re-schedule the flight such that it is not in the chosen airspace during the sampled moment, giving $N_{sec}[i, s] = N_{nsb}[i, s] = N_{cd}[i, s] = 0$.

We implemented this model in OPL, the *Optimisation Programming Language* [11]. As the resulting OPL model has non-linear constraints, the OPL compiler translates the model into code for ILOG Solver, rather than for CPLEX, and constraint solving [1] takes place at runtime.

4 Experiments

For our experiments, the chosen air-traffic control centre is Maastricht, in the Netherlands. The chosen multi-sector airspace within the Maastricht airspace consists of five sectors, covering the upper airspace of the three BeNeLux countries and some airspace of northern Germany, depicted in Figure 2, and characterised in Table 1. They are all high-density, en-route, upper airspace sectors (above FL 245). The sector identified by $sectorId$ stretches vertically between flight levels $bottomFL$ and $topFL$. Unfortunately, none of these sectors is below any other one, so that our traffic complexity resolutions cannot consider re-routing a flight through a lower or higher sector in the chosen airspace. The weights a_{sec}, a_{cd}, a_{nsb} and sector normalisation constants S_{norm} of the complexity metric are taken from [5]. There are an additional 34 feeder sectors (not listed here), for which we only need to know the $bottomFL$ and $topFL$ values. Since constraint solving is faster on integers than floats, we actually multiplied all the a_{sec}, a_{cd}, a_{nsb}, and S_{norm} values by 100. As their original accuracy is 2 decimal positions, we will not lose any precision, as long as any obtained complexity is divided by $10,000$ before it is displayed.

The chosen day is 23 June 2004. At the moment (5 August 2004) of choosing this day, it was the busiest day that far of the year 2004. The chosen hours are the peak traffic hours, that is from 07:00 to 22:00 local time. The chosen flights follow standard routes (no free flight) and are of the turbo-prop or jet type. The chosen point profiles are so-called model-3 (radar-corrected) profiles, as these better reflect actual traffic than model-1 (filed) profiles. The Central Flow Management Unit (CFMU) provided us with the flight profiles for these choices. There are $1,798$ flight profiles, after statically repairing 761 flight profiles that included impossibly steep climbs or descents (which would otherwise have to be repaired dynamically during complexity resolution) and discarding 26 flights whose profiles were not repairable.

The maximum approach time is 78 minutes for this traffic sample. By formula (1), this means that approach times can be changed by integer amounts of minutes within the range $[-4, \ldots, +8]$. This is a sufficiently large range, almost of the magnitude of the range of take-off-time changes, so that the second form of complexity resolution can be expected to lead to significant changes in its own right.

Our experiments show that the described forms of complexity resolution lead to systematic reductions of the sum of the complexities of the sectors of the chosen airspace.

In Table 2, every line summarises the results on the 180 instances obtained by taking now at every 5 minutes between 07:00 and 22:00 on the chosen day. The average reduction in the average complexity over the five sectors is shown over these instances for various values of the smoothing degree k, the length L (in seconds) of the time steps, and $lookahead$. We kept *ff* $=$ 90% of the planned flights in the chosen airspace, and used $timeOut = 120$ seconds. With larger values of $lookahead$, it is possible to get better complexity reductions, simply because more flights are not airborne yet and thus offer more opportunities for resolution. With larger values of k (and thus lower values of L), it is possible to get a slightly better complexity reduction, but with $k = 2$, it is possible already to get nearly a 50% complexity reduction. The experiments were done with OPL 3.7 under Linux 2.6.4-52 on an Intel Pentium 4 CPU with 2.53GHz, a 512 KB cache, and a 1 GB RAM. Most of the computations finished before timing out, or were retrospectively seen (upon a larger value for $timeOut$) to have found (near-)optimal solutions at the moment of timing out, which means that the proofs of optimality were more time-consuming than finding the optima. Other experiments confirmed that reducing *ff* or increasing $timeOut$ gives better results.

In practice, complexity resolution in an MSP context will not be a constraint optimisation problem (COP), as here, but rather a constraint satisfaction problem (CSP). There will be many additional constraints, which simply have to be satisfied, such as requiring the resolved complexities to be within prescribed intervals. Since CFMU flight profiles derived from the flight plans introduced by the airlines are not very accurate (witness the amount of necessary repairs we had to perform) and currently incorporate only an attempt at balancing the numbers of flights (the N_{sec} term of our traffic complexity metric) between sectors, we cannot impose such maximal or even minimal bounds on the resolved complexities, as feasible solutions might then not exist. Indeed, there

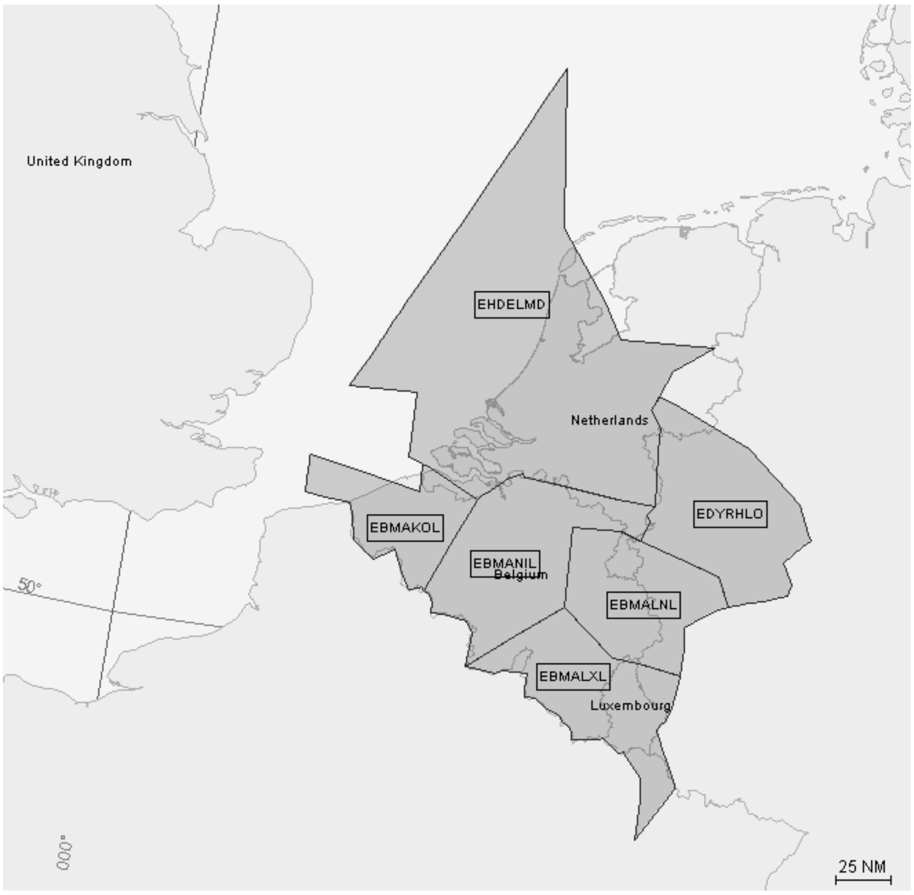

Figure 2: The chosen multi-sector airspace over Western Europe. On the chosen day, the sectors EBMAKOL and EBMANIL were collapsed into the sector EBMAWSL.

$sectorId$	$bottomFL$	$topFL$	a_{sec}	a_{cd}	a_{nsb}	S_{norm}
$EBMALNL$	245	340	7.74	15.20	5.69	1.35
$EBMALXL$	245	340	5.78	5.71	15.84	1.50
$EBMAWSL$	245	340	6.00	7.91	10.88	1.33
$EDYRHLO$	245	340	12.07	6.43	9.69	1.00
$EHDELMD$	245	340	4.42	10.59	14.72	1.11

Table 1: Characterisation of the chosen multi-sector airspace

are enormous discrepancies among the planned complexities, and even optimal complexity resolution can often not sufficiently reduce them. We ought to get better flight profiles for such additional constraints and experiments, so that we can then switch from a COP to a CSP. The reasons why we report here on optimisation experiments are that they give an upper bound on the runtime performance of the model and that this upper bound is already very good.

This is also why we do not illustrate the differences between some planned and resolved flight profiles, and rather just compiled our many experiments into the single two-dimensional Table 2. What that table does not show, however, is that the resolved complexities were much more evenly balanced among the chosen sectors than the planned complexities.

5 Conclusion

Constraint programming offers a very effective medium for *modelling* and efficiently *solving* the problem of minimising and balancing the traffic complexities of an airspace of adjacent sectors. The traffic complexity of a sector is here defined in terms of the numbers of flights within it, near its border, and on non-level segments within it. The allowed forms of complexity resolution are the changing of the take-off times of not yet airborne flights, the changing of the approach times into the chosen airspace of already airborne flights by slowing down and speeding up within the two layers of feeder sectors around that airspace, as well as the changing of the levels of passage over points in that airspace. Experiments with

lookahead	k	L	Average planned complexity	Average resolved complexity
20	2	210	87.92	47.69
20	3	180	86.55	50.17
45	2	210	87.20	45.27
45	3	180	85.67	47.81
90	2	210	87.29	44.67
90	3	180	85.64	47.13

Table 2: Average planned and resolved complexities in the chosen airspace

actual European flight profiles obtained from the Central Flow Management Unit (CFMU) show that these forms of complexity resolution can lead to significant complexity reductions and rebalancing.

A lot of related work is about dealing with potentially interacting pairs (PIPs) of flights. However, as the number of PIPs had a very low correlation with the traffic complexity, due to the fact that this element of moment complexity is already captured by the total number of flights in the sector and by the number of non-level flights in the sector [5], we did not have to resolve them away, nor even worry about their number in the resolved flight profiles.

There is also a large literature on complexity metrics for estimating workload, and we refer to [3] for a thorough recent survey thereof, as doing so here is beyond the scope of this paper.

The work closest to ours was done for the airspace of the USA in [10]. The main differences with our work are as follows. They have (at least initially) static lists of alternative routes to pick from for each flight and do not consider changing the time plans, whereas we dynamically construct (only vertically) alternative routes and new time plans. Their sector workload constraints limit the average number of flights in a sector over a given time interval (like the N_{sec} term of our complexity metric) and the number of PIPs (recall that our intended N_{pip} term was eliminated for lack of statistical significance [5]), but these terms are not part of a complexity metric. No multi-sector planning is performed to reduce and rebalance the workloads of selected sectors. However, a notion of airline equity is introduced toward a collaborative decision-making process between the FAA and the airlines.

Another important related work aims at minimising costs when holding flights (on the ground or in the air), if not re-routing them, in the face of dynamically changing weather conditions [2]. The main differences with our work are as follows. Their objective is cost reduction for airlines and airports, whereas our work is airspace oriented. They only consider ground holding and air holding, whereas we also consider the planning of earlier take-offs and the speeding-up of airborne flights. Their dynamic re-routing is on the projected two-dimensional plane, whereas ours is in the third dimension. Their sector workload constraints limit the number of flights in a sector at any given time, but there is no complexity metric and no multi-sector planning.

As this little overview of related work shows, the whole problem of optimal airspace and airport usage by the airlines is very rich, and only facets thereof are being explored in each project. Our work intends to reveal some new facets, such as complexity metrics and multi-sector planning.

Also, since the complexity resolution happens for a time interval $[m, \ldots, m + k \cdot L]$ in the future, constraints will be needed to make sure no unacceptable complexity is generated before m.

Another issue is the implementation of the calculated complexity resolutions. Additional constraints are needed to make sure that the proposed flight-profile updates can be implemented sufficiently quickly, and that doing so is still offset by the resulting complexity reductions and rebalancing among sectors. For instance, the number of flights affected by the changes may have to be kept under a given threshold.

There is a lot of other future work to do before an early prototype like ours can be deployed in a tactical context. Its main current objective is therefore strategic, namely to provide a platform where new definitions of complexity can readily be experimented with, and where constraints can readily be changed or added. This motivated the choice of constraint programming (CP) as implementation technology, since the maintenance of constraint programs is simplified. Furthermore, the likely addition of many more side-constraints will make the problem less and less purely combinatorial, and this is the typical scenario where CP is expected to be faster or to find better solutions than rival technologies [9].

Acknowledgements

Many thanks to Bernard Delmée, Jacques Lemaître, and Patrick Tasker at EuroControl DAP/DIA, for pre-processing the chosen CFMU raw data into the extended data and format we needed. This research project has been funded by EuroControl grant

C/1.246/HQ/JC/04 plus amendments 1/04 and 2/05.

References

[1] Apt, Krzysztof R. *Principles of Constraint Programming*. Cambridge University Press, 2003.

[2] Bertsimas, Dimitris and Stock Patterson, Sarah. The traffic flow management rerouting problem in air traffic control: A dynamic network flow approach. *Transportation Science*, 34(3):239–255, INFORMS, 2000.

[3] EuroControl, Directorate of ATM Strategies, Air Traffic Services division (DAS/ATS). *Complexity algorithm development: Literature survey and parameter identification*. Edition 1.0, 9 February 2004.

[4] EuroControl, Directorate of ATM Strategies, Air Traffic Services division (DAS/ATS). *Complexity algorithm development: The algorithm*. Edition 1.0, 7 April 2004.

[5] EuroControl, Directorate of ATM Strategies, Air Traffic Services division (DAS/ATS). *Complexity algorithm development: Validation exercise*. Edition 0.3, 10 September 2004.

[6] EuroControl Experimental Centre (EEC). *Pessimistic sector capacity estimation*. EEC Note Number 21/03, 2003.

[7] Flener, Pierre; Pearson, Justin; Ågren, Magnus; Garcia Avello, Carlos; and Çeliktin, Mete. *Air-Traffic Complexity Resolution in Multi-Sector Planning*. Technical Report 2007-003 at Department of Information Technology, Uppsala University, Sweden, January 2007. Available at `http://www.it.uu.se/research/publications/reports/2007-003/`.

[8] Garcia Avello, Carlos. *Decomplexing Traffic in a Multi Sector Planning Environment*. EuroControl DAS/ATS, 11 May 2004.

[9] Milano, Michela (editor). *Constraint and Integer Programming: Toward a Unified Methodology*. Kluwer Academic Publishers, 2004.

[10] Sherali, Hanif D.; Staats, Raymond W.; and Trani, Antonio A. An airspace planning and collaborative decision-making model: Part I — Probabilistic conflicts, workload, and equity considerations. *Transportation Science*, 37(4):434–456, INFORMS, 2003.

[11] Van Hentenryck, Pascal. Constraint and integer programming in OPL. *INFORMS Journal on Computing*, 14(4):345–372, 2002.

Keywords

Air traffic complexity, complexity resolution, multi-sector planning, constraint programming.

Biographies

Pierre Flener is an Associate Professor (Docent) at the Department of Information Technology of Uppsala University, Sweden. He received his PhD in Computer Science in 1993 from the Université Catholique de Louvain, Belgium. He is the founder and leader of the ASTRA research group in constraint technology at Uppsala University. He is a co-founder of SweConsNet, the Network for Sweden-based researchers and practitioners of Constraint technology.

Justin Pearson is a Senior Lecturer at the Department of Information Technology at Uppsala University, Sweden. He received his PhD in Electronic Engineering in 1996 from the University of Kent at Canterbury, UK. He is a member of the ASTRA research group in constraint technology at Uppsala University, and a co-founder of SweConsNet, the Network for Sweden-based researchers and practitioners of Constraint technology.

Magnus Ågren is a PhD student at the Department of Information Technology of Uppsala University, Sweden. He is a member of the ASTRA research group in constraint technology at Uppsala University.

Carlos Garcia Avello works for the EuroControl agency on R&D activities dealing with the development of advanced decision support tools in the area of air-traffic complexity management and arrival flows management. Other areas include aircraft performance modelling, trajectory prediction, and 4D trajectory management. He graduated in 1982 from the Department of Mathematical Mechanics of the Université Catholique de Louvain, Belgium.

Mete Çeliktin works for the EuroControl agency on R&D coordination and validation activities. He received a PhD in Computer Science in 1994 from the Université Catholique de Louvain, Belgium.

Søren Dissing works as an air traffic management expert for the EuroControl agency, where he oversees the operational implementation of data link in controller-pilot data link communications. He has previously been involved in the development of controller decision support and air traffic complexity management. Until 1996, he worked for 22 years as an air traffic controller at the Copenhagen Area Control Centre in Kastrup, Denmark.