A methodology for strategic planning of aircraft trajectories using simulated annealing

Supatcha Chaimatanan, Daniel Delahaye, Marcel Mongeau

A methodology for Strategic Planning of Aircraft Trajectories using Simulated Annealing

Supatcha CHAIMATANAN, Daniel DELAHAYE, and Marcel MONGEAU

Ecole Nationale de l'Aviation Civile (ENAC), Toulouse, France

Abstract

Global air traffic demand is continuously increasing and it is predicted to be doubled in the near future. In order to sustain such tremendous traffic volume, Air Traffic Management System (ATMS) has to reform its architecture and improve its performance. Since one of the airspace capacity limitations is the air traffic controllers' workload which is strongly related to the traffic situation in the control sector. A possible way to increase the airspace capacity is by mitigating the traffic situation by structuring the aircraft trajectories.

This article presents an optimization formulation of the strategic air traffic conflict reduction problem together with a methodology to structure the 4 dimensional aircraft trajectories (3 spatial dimensions and time). The number of interactions (potential conflicts) between aircraft trajectories are minimized using route-slot allocation techniques. We introduce a methodology that uses a Simulated Annealing (SA) algorithm combined with a local heuristic search method. Our preliminary results show that this approach can effectively reduce the interactions between aircraft trajectories, thereby decreasing the need of tactical intervention. Therefore reducing the controller's task load.

1 Introduction

According to [1], global air traffic demand has continuously increased during the past 20 years. Furthermore, it is expected to raise at 5% growth rate per year. Hence, the European airspace, for instance, will experience a traffic volume of 50,000 flights a day in 2030.

To maintain safety, punctuality and to reduce environmental impact under such a huge demand, it is necessary to enhance the Air Traffic Management System (ATMS) efficiency and expand its facility. Therefore, many ATMSs are currently being modernized. This is the concern of the Single European Sky ATM Research (SESAR) project which was launched in 2004 by the European community. Likewise, the National Airspace System (NAS) in the

United States is being transformed from a ground-based system to a satellite-based system called Next Generation Air Transportation System (NextGen).

One of the key concepts to reform the ATM architecture is to shift from ground-based operation to trajectory-based operation (TBO) where the airspace will not be constrained by artificial boundaries anymore. Instead, the airspace will be balanced by the influences of the trajectories respective to the airspace user's requirement. In this concept, the trajectories are precisely defined in four dimensions (three spatial dimensions and time). This 4D trajectory will be flown by the aircraft with very high longitudinal navigation accuracy (± 10 second errors).

Since TBO makes the air traffic becomes much more predictable and flexible, it allows to improve the conflict management by shifting from tactical intervention to strategic deconfliction. This permits us to significantly reduces the controller's task load by minimizing the level of intervention in the tactical phase.

During recent years, several methods have been proposed to find optimal aircraft trajectories in order to solve conflicts. For example, in [2], a constraint programming approach was used to address the conflicts on the upper airspace. An approach to find a conflict-free wind optimal route in real time is presented in [7]. Another method, using an integer programming model adapted for large-scale ATFM problem, is presented in [3]. A method to generate conflict-free based on a light propagation algorithm is introduced in [4]. Finally, a comparison of different optimization methods used for traffic flow management is provided in [9].

In this study, we propose a strategic potential conflict reduction algorithm. In this framework, the aircraft trajectories are four-dimensionally separated by imposing deviations to their nominal trajectories and by shifting their departure slots (route-slot allocation technique).

It is obvious that the solution space of the problem (route choices and departure slot choices) is highly combinatorial. This problem can easily be addressed by stochastic optimization such as Genetic Algorithm, Simulated Annealing, Tabu Search etc. In this study, we propose a methodology using two optimization approaches which we implement on two test problems consisting in 1,000 and 4,000 historical trajectories over France. The first optimization approach relies on a standard Simulated Annealing algorithm, while the second one is based on Simulated Annealing with a local search improvement.

This paper is organized as follows. We propose in Section 2 a mathematical formulation of the problem and a conflict-detection scheme. We describe our general simulated annealing algorithm in Section 3. We present preliminary numerical results in Section 4 and we conclude in Section 5.

2 Problem Modeling

In this section, we first introduce a mathematical formulation of our problem and then we propose a conflict-detection scheme.

2.1 Mathematical Modeling

We consider a given set of trajectories $R = \{r_1, r_2, \ldots, r_N\}$, where N is the total number of flights, defined in four dimensions linking origin-destination (OD) pairs in the airspace. The objective is to allocate alternative routes and/or departure slots for each trajectory so that the overall interaction between trajectories is minimized.

The interaction between trajectories is defined by the total number of times that the separation rule is violated. The interaction encountered by a single aircraft i, denoted by Γ_i, depends on both the route and the departure slot choices of the other flights in the airspace.

In this study, a given direct trajectory is defined by a set of 4D coordinates (x_k, y_k, z_k, t_k) obtained from a simulator with sampling rate of 15 seconds ($t_{k+1} - t_k = 15s$). An alternative route r_i linking an OD pair can be defined by a set of virtual waypoints (p_1^i, p_2^i) along the route where $i = 1, \ldots, N$ is the flight number, and where p_1 and p_2 are as shown in Figure 1. The virtual waypoints will be our decision variables. The set of virtual waypoints for each trajectory must yield maximal possible deviation less than 10% of the distance of the nominal trajectory in order to avoid excessive extra fuel costs.

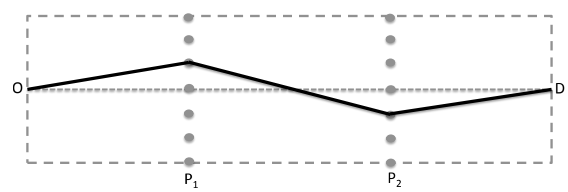

Figure 1: An example of horizontal deformation

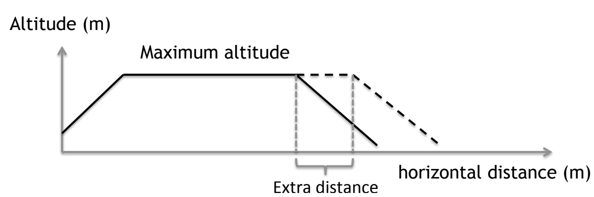

Figure 2: An example of vertical deformation

In this study, we suppose that the given flight levels are already optimal. Therefore, the virtual waypoints that allow to deform the nominal trajectory in the horizontal plane yield the altitude profiles that will be only extended according to their maximal altitude as shown in Figure 2.

Another possibility to avoid having two aircrafts in a potential conflict is by shifting them in the temporal space. Hence, the departure time of flight i can be expressed as $d_i = \tilde{d}_i + \delta_i$ where $\tilde{d}_i$ is the nominal departure slot of flight i, and δ_i is the time shift which can be either positive (delay the flight) or negative (advance the flight). These δ_i's complete our set of decision variables. In the sequel, p and δ will denote the vector of R^N where i^{th} component is $p_i = (p_1^i, p_2^i)$, and δ_i respectively.

Therefore, the route-slot allocation problem can be formulated in the form of an optimization problem as following;

$$\min_{(p,\delta)} \sum_{i=1}^{N} \Gamma_i$$

subject to

$$\Gamma_i = f(R, \delta) \tag{1a}$$

$$r_i = g(p_i) \qquad i = 1, 2, \ldots, N \tag{1b}$$

$$p_i = (p_1^i, p_2^i) \qquad i = 1, 2, \ldots, N \tag{1c}$$

$$d_i = \tilde{d}_i + \delta_i \qquad i = 1, 2, \ldots, N \tag{1d}$$

$$p_i \in p_1^1 \times p_2^i \tag{1e}$$

$$\delta d_i \in D \tag{1f}$$

where
Γ_i is the number of interactions aircraft i is involved;
f is a function that allows to compute the number of interactions encounter by aircraft i from a given set of flight plan (R, δ).
g is a function that allows to construct an alternative route r_i from given waypoints p_i.
N is the total number of flights;
n_i is the total number of possible virtual waypoint among which to choose for trajectory i;
m is the total number of time shift slots;
and the discrete search space for the virtual waypoints is determined by the sets $p_1^i := \{p_1^{i1}, p_1^{i2}, \ldots, p_1^{in_i}\}$, $p_2^i := \{p_2^{i1}, p_2^{i2}, \ldots, p_2^{in_i}\}$ while the time shifts must be chosen in the discrete set $D_i := \{d^{i1}, d^{i2}, \ldots, \dot{i}m\}$.

The search space of this discrete optimization problem is highly combinatorial and the problem is NP-Hard (relative to the number, N, of flights involved).

2.2 Potential Conflict Detection Scheme

Let us now present our potential conflict detection scheme to evaluate the number of interactions between trajectories. Two aircrafts are considered to be in a potential conflict if their horizontal separation is less than 5 Nautical miles (Nm) or if vertical separation is less than 1000 feet (ft). We can imagine that the aircraft has a protection zone defined by a three-dimensional cylinder as shown in Figure 3.

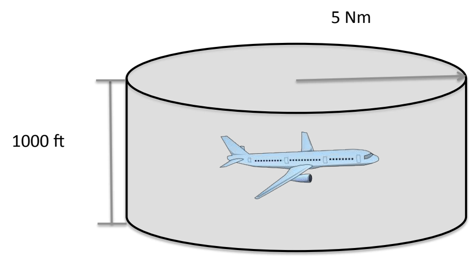

Figure 3: The separation-norm cylinder

To detect a conflict, one can compare the aircraft position along its trajectory with the other aircraft in the airspace (pairwise comparison). There are several researches addressing this aircraft conflict detection problem. For instance, the method presented in [5]. is efficient for detecting conflicts when dealing with small number of aircraft. However, for many aircrafts this exhaustive process would yield a very large number of comparisons (up to $(k.N)^2$ where k is the number of sampled coordinates on each trajectory). The probabilistic approach of [10] predicts the future position of aircrafts and computes the probability of conflict. However, this method only addresses conflict detection in short-term mid-term and is limited to a small number of aircraft.

The conflict grid method presented in [8] is a conflict detection method that can be used when a large number of aircrafts are involved. In [8], the binary value stored in each cell is initialized to zero. The N trajectories are considered sequentially. If an aircraft meets this cell, the value will be set to one. If another aircraft tries to occupy the same cell, then this cell becomes forbidden (a constraint) for the following trajectories.

In this paper, the potential conflict can be detected by first associating each 4D coordinate (x, y, z, t) of each discretized trajectory to a specific cell in a four dimensional grid. The size of a cell is defined by the separation norm. In each of these 4D cells, instead of storing a binary value, we propose to store a list of aircrafts occupying the corresponding cell. This data structure enables us to update easily the total number of interactions, when we modify the routes or the departure slots (through a small change in our decision variables p and δ). The idea is that there is no need to recompute the number of interactions encountered by an aircraft whose route and departure slot are not modified and not affected by the (other aircraft's) modified trajectories.

Then, for each non-empty cell, we checks all the neighboring cells . The potential conflict is detected if the neighboring cells are occupied.

An example of conflict detection grid in two dimension is illustrated in Figure 4. Consider the m^{th} point of trajectory j. The conflict-detection scheme checks all the neighboring cells (in gray in Figure 4). In this case, the m^{th} point yields three conflicts locations due to three points from trajectory k. By repeating this operation along trajectory j, the total number of interactions encountered by trajectory j can easily be evaluated by summing up the number of interactions over each point of the trajectory.

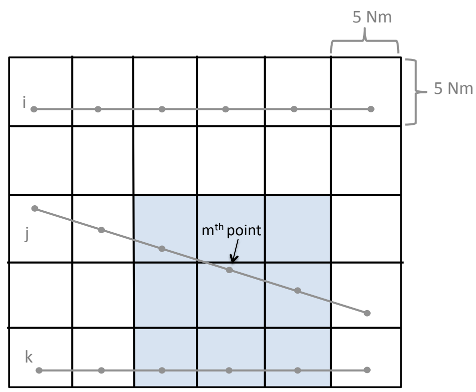

Figure 4: A 2D (space only) example of our conflict-detection scheme

3 Simulated Annealing Algorithm

Simulated Annealing (SA) is a so-called metaheuristic optimization method that was introduced by S. Kirkpatrick et al. in 1983 and V. Cerny in 1985 [6]. Its popularity comes from its ability to avoid getting trapped in local minima and to find a near-global optimal solution for NP-hard combinatorial optimization problems.

The simulated annealing method is inspired by the annealing process in metallurgy where the state of a material is modified by controlling the cooling temperature. The process consists in heating up a material to bring it to a high energy state. Then, the material is slowly cooled down by decreasing the temperature according to a properly pre-defined *cooling schedule*, keeping at each temperature step a sufficient duration such that the material reaches its thermal equilibrium before further reduction of temperature. As the temperature tends towards zero, the material reaches a crystallized solid state where the energy is at the absolute minimum (decreasing too rapidly the temperature yields states at locally-optimal energy levels).

When solving an optimization problem using the simulated annealing method, the cost function to be minimized is analogical to the energy (E) of the physical problem, and a control parameter controlling the exploration of the solution space plays the role of the "temperature" in the physical system.

The simulated annealing method can be summarized as shown in figure 5.

Simulated Annealing Algorithm
1. At $T = T_0$, start from a random solution S_{init}
2. Evaluate $y := E(S_{init})$
3. Set the current best-known solution $S_{best} := S_{init}$ and $y_{best} := E(S_{init})$
4. for (i = 0 to nbTransitions)
    Generate randomly a perturbed candidate solution S_i
    If $E(S_i) < E(S_{init})$
        accept S_i as current solution
      else
        if $r < \exp((E(S_{init}) - E(S_i))/T)$, where r = random number
          accept S_i as a current solution
    If $E(S_i) < y_{best}$
      set $S_{best} := S_i$ and $y_{best} := E(S_i)$

5. Reduce the temperature $T_{k+1} = \alpha\, T_k$
6. Repeat step 4 and 5 until termination criterion = true

Figure 5: Standard simulated annealing algorithm

The efficiency of the simulated annealing algorithm, with respect to convergence speed and quality of the solution depends strongly on the following user-defined control parameters:

- The initial temperature, T_0.
  The initial temperature should be sufficiently high to ensure a random walk to diversify the search at high temperatures.

- The cooling schedule.
  The temperature should be reduced slowly and smoothly. Lowering abruptly the temperature yields convergence towards a local minima. A widely-used cooling schedule is the geometrical law of decrease (used in the implementation of Figure 5): $T_{k+1} = \alpha.T_k$, where $\alpha < 1$ is a pre-defined constant set by the user.

- The rules of acceptance.
  Generally, the *Metropolis rule* (used in the implementation of Figure 5) is the rule of acceptance. The idea is, first to accept occasionally a solution which increases the energy of the current state, to escape from a local minima; second, such an acceptance worsening the value of the cost function get less and less frequent as the temperature decreases.

- The length of the homogeneous Markov chains.
  The *length of the Markov chains*, or the duration of each temperature step, should be long enough to ensure that the *thermodynamic equilibrium* is reached.

- The termination criterion.
  In the physical annealing, the process terminates when the temperature reaches zero.

However, in the simulated annealing algorithm, the rate of acceptance then becomes very low. Therefore, the method becomes ineffective at low temperatures. A widely-used termination criterion is when no more solution are accepted during a certain number of consecutive temperature decrease steps.

Practical suggestions for choosing the user-defined control parameters are given in [6].

3.1 Local Heuristic Search

In the simulated annealing method, at high temperatures most of the computation time is spent on exploring the state space, and at low temperatures most of the computation time is spent on evaluating solutions that will not be accepted. Therefore, to improve the efficiency of this method, we introduce a greedy search within the simulated annealing process. This search method is summarized in Figure 6.

Greedy search method
1. Let $S_{current}$ be the current state solution.
2. Generate a set of neighbors solutions.
3. Let S' be the best neighbor of $S_{current}$.
4. If (E(S') <= E($S_{current}$))
   set $S_{current}$:= S'
   else repeat step 2.
5 Repeat step 2 until termination criterion is satisfied.

Figure 6: Greedy search method

The idea is to switch from simulated annealing to a greedy search from time to time in order to accelerate convergence. The probability of switching to the greedy search depends on the temperature. At high temperatures, the probability is low in order to preserve diversity of the search. At low temperature, when simulated annealing becomes less effective, this probability is higher in order put more emphasis on intensification of the search around the best solutions found so far.

4 Numerical Experiments

In order to implement simulated annealing, we must first determine how to generate a perturbation to a point in the search space. In this study, after the initial state is randomly generated, the perturbation is generated first by randomly choosing a trajectory to be modified. Then, for each chosen flight, a new route and/or a new departure slot choice is randomly generated.

The initial temperature are determined according to recommendation given in [6]. The initial rate of acceptance and the number of transitions in each temperature step are experimentally tuned (with respect to minimum computation time criteria).

In the cooling process, the temperature is decreased according to the geometrical law $T_{k+1} = 0.99 \cdot T_k$ (with $\alpha = 0.99$). The algorithm is terminated when the optimal solution (routes and departure slot choice with zero conflict) is reached or after a pre-defined maximum number of transitions is reached (in our test, the algorithm is terminated when the final temperature T_f satisfied $T_f = T_{init}/1000$).

In the simulated annealing with local heuristic search method, the probability of selecting the local search method is set to be less than 0.1 % at initial temperatures, and it is then continuously increased up to 1% at the final temperature.

We implemented our methodology in Java on a Core2Duo 2.4 GHz computer with 8 GB DDR3 RAM on a Unix platform. The methodology was tested on two test problems involving 1,000 and 4,000 historical-data trajectories over France respectively. The results show that our approach is able to find a very good solution and to provide a conflict-free flight plan for these test problems. The results and the computation time of the proposed algorithm using the simulated annealing alone and the one with local search is shown in Table 1.

Instance	Method	Computation time		Initial no. of interactions	Final no. of interactions	avg. departure time shifting (mins)	avg. extra distance (km)
		average (mins)	σ (mins)				
1,000	Simulated Annealing	9.07	0.47	13,140	0	23.5	15.8
	Simulated Annealing with local search	0.70	0.19		0		
4,000	Simulated Annealing	227.7	3.15	74,972	0	30.8	21.9
	Simulated Annealing with local search	19.05	7.67		0		

Table 1: Numerical results

It is obvious that for these particular problems, the integration of local search method within the simulated annealing algorithm can significantly decreased the average computation time while providing good results.

5 Conclusion

We introduced in this paper, an optimization formulation of the strategic air traffic potential-conflict reduction problem together with a methodology aiming at reducing interactions between aircraft trajectories. Our methodology has two options. One using simulated annealing and the other combining simulated annealing with a local search method to reduce the total computing time. The algorithm was tested on two test problems involving 1,000 and 4,000 flights. The results show that both options are viable to reduce effectively the number of trajectory interactions. Adding the local search reduces significantly the computing time.

References

[1] Worldwide market forecast for commercial air transport 2010-2029. Technical Report YGR-5067, Marketing, Japan Aircraft Development Corporation, 2010.

[2] Nicolas Barnier and Cyril Allignol. 4d-trajectory deconfliction through departure time adjustment. In 8^{th} *USA/Europe Air Traffic Managment Research and Development Seminar (ATM 2009)*, 2009.

[3] Dimitris Bertsimas, Guglielmo Lulli, and Amedeo Odoni. An integer optimization approach to large-scale air traffic flow management.

[4] Nour Elhouda Dougui, Daniel Delahaye, Stephane Puechmorel, and Marcel Mongeau. Light propagation algorithm for aircraft trajectory planning. In *American Control Conference (ACC)*, 2011.

[5] Gilles Dowek and César Munoz. Conflict detection and resolution for 1,2,...,n aircraft. In *7th AIAA Technology, Integration and Operations Conference*, 2000.

[6] Johann Dréo, Alain Pétrowski, Patrick Siarry, and Eric Taillard. *Métaheuristiques pour l'optimisation difficile*. EYROLLES, 2 edition, 2005.

[7] Matt R. Jardin. Real-time conflict-free trajectory optimization. In *5th USA/Europe Air Traffic Managment Research and Development Seminar (ATM 2003)*, 2003.

[8] Matthew R. Jardin. *Towards Real Time En Route Air Traffic Contol Optimization*. PhD thesis, Standford University, 2003.

[9] Jason Lohn and Jeseph Rios. A comparison of optimization approaches for nationwide traffic flow management. In *AIAA Guidance, Navigation and Control Conference*. IEEE International, 2009.

[10] Prandini M., Hu J., Lygeros J., and Sastry S. A probabilistic approach to aircraft conflict detection. *IEEE Transactions on Intelligent Transportation System*, 1(4):199–220, 2007.