Interactive Air Traffic Control automation in oceanic airspace[1]

Francisco Freitas[1]

Miguel Barão[2]

Rodrigo Ventura[1]

[1] Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal

[2] INESC-ID
Lisbon, Portugal

Abstract

Air traffic controllers workload limits impose upper bounds to the amount of traffic manageable in a given air sector for a given time frame. Air Traffic Control (ATC) automation methods open the possibility of reducing this workload by shifting to the machine the tasks of (1) detecting potential conflicts, and of (2) proposing to the controller ATC instructions that prevent such conflicts. We propose a decision support system based on a combinatorial optimization approach using a branch-and-bound method. Given a known traffic situation, we proceed by simulating the trajectories of traffic, taking into account possible instructions to separate traffic. In this study we considered only flight level change instructions, given at report fixes. The cost function employed includes both a measure of vertical deviation from the filed flight plan (FPL) and the total amount of ATC instructions. The multi-criteria problem is solved interactively, as the operator directs the algorithm towards the solution, indicating its preferences at intermediate points in the simulation. As a case study, we analyse the problem of oceanic airspace, where conventional ATC is used due to the lack of radar coverage.

1 Introduction

The current ATC system is already near its full capacity in some of the most busy sectors, and will eventually be overloaded if air travel grows as predicted [2]. Given that controllers are already working at the top of their capacities and that an increase in their workload would most likely be a threat to system safety, an increase in air traffic can only be safely handled by a future ATC system if solutions are found to reduce their workload.

A commonly studied approach to the problem focuses on redesigning the existing airspace organization, by creating functional airspace blocks designed to simplify air traffic control and by allowing aircrafts to fly direct routes between fixes (Free Route). An important field of research aims at decentralizing the ATC system, by passing some of the controller's tasks to aircrafts, allowing agents to self-organize themselves and to perform conflict detection and resolution autonomously. A study comparing the performances of centralized and decentralized strategies is presented in [3].

Better suited for short-term application is the approach of developing computerized decision support tools to improve the performance of the current centralized system (a conceptual scheme for such tool is shown in figure 1). Solutions derived from this approach may range from simple tools, that detect short and medium-term conflicts without suggesting any solutions to total automation, in which an automatic tool detects conflicts, searches for an optimal solution and issues the instructions to the aircrafts by means of some form of digital data format, replacing human controllers.

Our work follows this latter paradigm, and lies midway between a passive system and total automation. A breakdown of the developed tool is presented in figure 2.

1.1 Approach outline

The developed program receives as inputs the current situation of a set of aircrafts and their filed FPLs inside a certain flight region, and calculates their long-term predicted trajectories within a certain time window, using a point-mass dynamics model and a simplified Autopilot. A combinatorial search is carried out to analyse each plan within the state-space. Pairwise conflict detection is executed for each individual plan. A Cost Function taking into account the deviation from the filed FPL and the total amount of ATC instructions was created, allowing a branch-and-bound method to be implemented, pruning regions of the solution search and reducing simulation time, assuring optimality. A Monte Carlo simulation is ran as a robustness check to test each calculated plan. A set of particles is created for each aircraft, simulating possible trajectories.

At the end of the simulation, the algorithm returns a global plan, consisting of a set of instructions to be issued to each aircraft, and presents this plan to the human controller. The controller may accept the plan, or request the algorithm to search for other solutions, indicating which cost function criteria to improve. This iterative process continues until the controller accepts one of the proposed plans.

As a case study, we chose to analyse the operation in Oceanic Airspace, where conventional ATC is used due to the lack of radar coverage. Restrictions were created so that aircrafts fly their filed horizontal route at the requested airspeed, and trajectory changes are only allowed in the vertical plane, with instructions being issued only at report fixes. The algorithm is run every time a position report is received from an aircraft, ensuring the current advisory is based on the most updated information available.

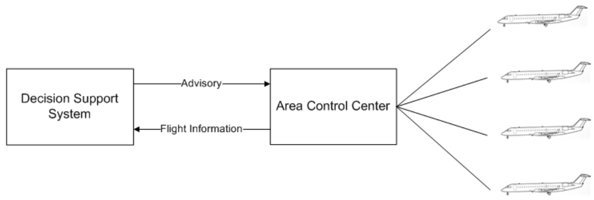

Figure 1: Decision support system integration

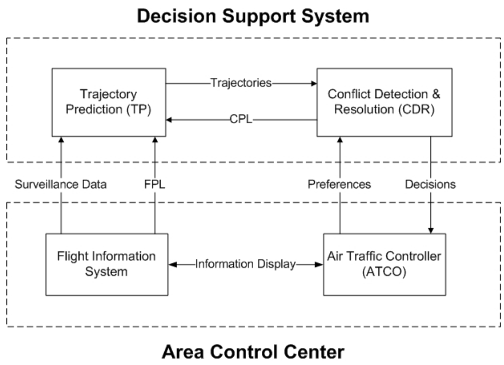

Figure 2: Decision support system breakdown

2 Trajectory prediction

The method for the prediction of the future trajectory of an aircraft is based on the one presented by Glover and Lygeros in [1]. Having at its disposal the current state of an aircraft – position, speed and attitude – its future intention – in FPL format – and an aircraft-specific model – loaded from Eurocontrol's Base of Aircraft Data (BADA) – the program is able to predict the trajectory of that aircraft using a point-mass model.

A six-state control system is implemented, with its states being: geographical position (Latitude φ and Longitude λ), altitude (h), true airspeed (TAS), heading angle (ψ) and mass (m). The aircrafts are assumed to have three control inputs: engine thrust (T), bank angle (ϕ) and flight path angle (γ).

Being expressed in a cartesian orthogonal frame, the model presented in [1] is unsuited to deal with large length trajectories, as it does not account for the curvature of the Earth. This limitation was overcome using

[1]This work was supported by the FCT project [PEst-OE/EEI/LA0009/2013]

quaternion algebra to calculate each successive position of an aircraft by applying a 3D rotation about an axis containing the center of the Earth to the aircraft position at the last time step. This way, the aircraft horizontal position ceases to be expressed as a (x,y) pair, but rather as geographic Latitude (φ) and Longitude (λ).

The three control inputs form a basic Autopilot, ensuring the aircraft follows the trajectory and airspeed required by its FPL. These control inputs result from proportional-integral controllers for the Thrust and the Flight Path Angle in Climb/Descent phase, and from proportional controllers for the Bank Angle and the Flight Path Angle in Cruise phase.

This continuous-time system is discretized using a fourth-order Runge-Kutta method to solve the ordinary differential equations. From now on, system variables are indexed with a discrete index k, *i.e.* t = kT, where T is the integration time step.

3 Conflict detection and resolution

3.1 Cost Function

Several different criteria may be used to evaluate a plan issued to an individual flight – trajectory change relative to FPL, number of instructions issued, total time inside FIR, fuel consumption – and there are also criteria suited to qualify a global plan – how much certain flights are penalized relative to others, controller's workload peak at a given period.

In this work, two criteria are used in the Cost Function: (1) vertical deviation from the filed FPL (denoted D) and (2) total amount of ATC instructions (denoted N). The cost function is obtained by multiplying both criteria by weight coefficients (denoted λ_N and λ_D) and then summing them:

$$f = \lambda_N N + \lambda_D D \tag{1}$$

3.2 Combinatorial Optimization

The algorithm runs a branch-and-bound method to reduce computation time, assuring optimality. Starting at Current Time (the real time being observed by the human controller), the search advances progressively through time, generating predicted trajectories for each aircraft. Every time an aircraft reaches a waypoint, a node is created and several branches are considered, corresponding to all possible flight levels that aircraft can be assigned to. A lower bound of the cost is computed for each branch, measuring the minimum cost each one will add to the total cost function. The branch with the lowest heuristic is chosen and the corresponding action is executed.

At each node, conflict detection is executed in the time window between the previous node and the current one. If a conflict between a pair of aircraft A and B is detected, the algorithm backjumps to the last node (the most advanced in time) where a decision was made involving either A or B, discarding every node in between, and selects a different branch to proceed the search. If the conflict is resolved, the search continues. If not, the algorithm backjumps again to a node involving A or B, repeating the process until resolution is achieved. A solution plan is found when the simulation time window limit is reached, or when every flight in the scenario has already abandoned the airspace being controlled. After the first solution plan has been found, an upper bound is available and may be used to prune branches in the middle of the search tree, whenever a node has a cost higher than the current upper bound. This greatly reduces the run time of the algorithm, requiring the expansion of fewer nodes. The simulation is concluded when the whole search tree has been explored. Following the branch-and-bound method, branches are pruned whenever their lower bound exceeds the current upper bound (i.e., the minimum cost among all solutions found so far). This plan is selected to be presented as a suggestion to the controller.

3.3 Interactive decision-making

The choice of a two criteria cost function deters the use of the conventional optimization techniques commonly used for single-variable problems. When more than one objective is considered, a multi-criteria decision making problem must be solved. In multi-criteria problems, the concept of solution *optimality* is replaced by those of *efficiency* and *Pareto optimality*. In general, there is not a single optimal solution for a given problem, but the goal of a multi-criteria problem is rather to calculate the set of solutions that are *Pareto optimal*. This set of solutions may be called *Pareto front* or *efficient set*.

In this work, an interactive approach is chosen to solve the multi-criteria problem. The algorithm searches for efficient solutions one at a time, using a fixed ratio $\frac{\lambda_D}{\lambda_N}$. Each time it finds an efficient solution, the algorithm presents it to the controller. The controller decides whether he accepts the proposed plan or he requests the algorithm to proceed the search. In the latter case, he must inform which criteria to improve. The cycle ends when the controller accepts one of the proposed plans.

4 Results

The algorithm was tested for a time window of 4 hours with traffic scenarios generated randomly. From a pool of nominal horizontal trajectories based on real commercial flight plans, a certain number of flights *NF* is created and placed randomly at different flight levels and entering the controlled airspace at different times. The chosen controlled airspace was Santa Maria FIR, Portugal's oceanic airspace. The algorithm's ability to calculate optimal solutions was verified for the teste traffic densities. To test the algorithm computational performance, running time t_{run} was measured as a function of *NF*.

Figure 3 shows total running time t_{run} to increase in an approximately exponential manner as the scenario number of flights *NF* is increased. This clearly indicates the complexity of a given scenario – *i.e.* the computational effort it requires from the algorithm – grows much faster than the state-space dimension, which increases linearly with the number of flights *NF*. This may attributed to the combinatorial effect as more aircraft are added to a scenario, increasing the number of conflicting aircraft, and forcing the algorithm to explore a much larger percentage of the search tree.

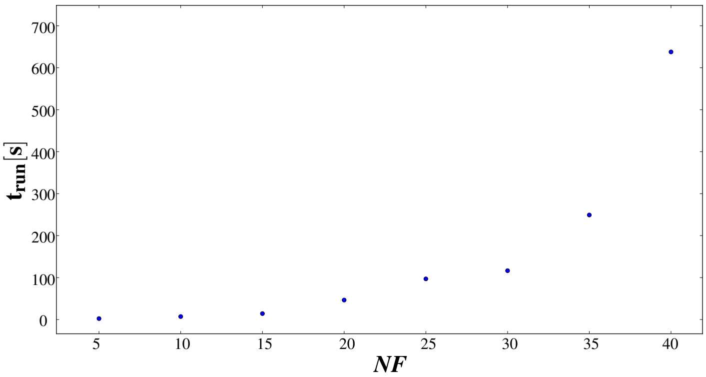

Figure 3: Running time as a function of NF

5 Conclusions

The developed program has proved its capacity in solving the scenarios that it was subjected to. More exhaustive testes are being conducted, especially focusing on the gain achieved by requesting the algorithm to be optimal, i.e. on how much the cost function improves as a consequence of the algorithm not 'settling' for the first leaf node of the search tree (know as the *greedy solution*) and rather exploring the whole tree to guarantee the *optimal solution* is found.

References

[1] W. Glover and J. Lygeros. A stochastic hybrid model for air traffic control simulation. In *Hybrid Systems: Computation and Control, 7th International Workshop*, 2004.

[2] B. Kirwan and M. Flynn. Towards a controller-based conflict resolution tool - a literature review. *EUROCONTROL document ASA.01.CORA.2.DEL04-A.LIT*, 2002.

[3] J. Krozel et al. System Performance Characteristics of Centralized and Decentralized Air Traffic Separation Strategies. In *4th ATM R&D Seminar*, 2001.