Satisficing Game Theory for Conflict Resolution and Traffic Optimization

Francesco Bellomi, Roberto Bonato, Vincenzo Nanni, and Alessandra Tedeschi

In the current, centralized approach to Air Traffic Control (ATC) air traffic controllers are responsible for the safe and efficient flow of aircraft. This situation would change with the introduction of Airborne Self-Separation as a distributed and scalable approach to ATC. The major technological challenge that must be tackled to make Airborne Self-Separation a viable alternative to the traditional controller-based approach is to devise a safe and reliable technology to solve conflicts and improve global performances in an uncontrolled environment. In this paper we introduce an algorithm that applies Satisficing Game Theory (SGT) to solve conflicts in the framework of an overall optimization of the traffic flow. This decentralized and cooperative algorithm is inspired by the work presented in [1]. The SGT provides a strategy that permits decision-makers to reach a compromise in the interest of achieving both individual and group goals, implementing altruistic behavior. The paper presents the first results we collected by running a software tool which simulates the behavior of the SGT algorithm in a 3D environment, using air traffic samples provided by the Italian air traffic service provider (ENAV). These results are the starting point of a further enquiry to explore the actual impact of the introduction of such a technology into a realistic ATC environment.

INTRODUCTION

In today's Air Traffic Management system, the responsibility for maintaining a safe and efficient traffic flow is entirely delegated to controllers, who issue flight instructions to pilots as well as grant or deny authorizations to apply specific procedures. Because of the ca-

---

Roberto Bonato, Francesco Bellomi, and Alessandra Tedeschi are with Deep Blue, Rome, Italy. Vincenzo Nanni is with ENEA, Rome.

Received November 2, 2007; accepted July 15, 2008.

Air Traffic Control Quarterly, Vol. 16(3) 211–233 (2008)
 CCC 1064-3818/95/030163-20

pacity limitations of this approach, several possible alternatives and complementary approaches are currently under investigation. *Airborne Self-Separation* [2] or *Free Flight* in less recent literature [3], [4], is an operating environment in which pilots are allowed to select their route in real time and without any external control by air traffic controllers, and therefore they bear more responsibility for the safe and efficient conduct of the flight. The advantages of Airborne Self-Separation broadly conceived will be twofold. On the one hand it should lead to reduced costs, lower fuel consumption and increased capacity. Self-optimization by the airlines could be more effective than any global optimization that can be performed by a human controller (see [5]), because different airlines might give higher priority to different parameters which depend on company strategy or other factors only known to the airline and crew. On the other hand, in the current centralized, controller-based approach there are difficulties in scaling up to cope with the increasing volume of future air traffic while keeping or improving air traffic safety (air traffic is predicted to grow exponentially at a rate of 5 to 6 percent per year [6]). If appropriately coupled with necessary improvements in automation, Airborne Self-Separation could reduce airline operating costs and controllers' workload, therefore resulting in an increase in efficiency without adversely affecting the safety of air travel.

The paper begins with a Section that lays out the scientific objectives; first, the motivation of our work on Airborne Self-Separation; followed by the introduction of the Collective Intelligence framework, as a mature and promising formal approach for Conflict Resolution and Traffic Optimization in ATC; and then we provide a brief review of conflict resolution techniques that have been proposed in the literature. A Section follows in which we present the multi-agent algorithm that was implemented in our simulator and that was based on the Satisficing Game Theory (SGT). We highlight the advantage in terms of flexibility and robustness of that approach. The following two Sections give the preliminary results of our work, with a detailed description of our prototype and its performance when subjected to a set of difficult traffic scenarios. Following the scenario performance is a brief description of open issues, with particular focus on the need for real-world testing, such as evaluating the techniques with imperfect data links and realistic traffic. Finally, in the Appendix, some details of the algorithm implementation are reported.

MOTIVATION OF THE PROBLEM

Conflict Resolution for Airborne Self-Separation

The single most important safety issue for air traffic in general, and for Airborne Self-Separation in particular, is conflict resolution. Current procedures almost entirely rely on the controller's ability to coordinate and organize the flow of air traffic, with the help of guidelines, best-practices, and individual experience. Automation plays today just a limited role, but with the steady increase of air traffic volume, demand for automated decision-support systems for ATM will grow. Decisions that are nowadays based on human interaction will have to be progressively replaced by more efficient forms of control with the development of more sophisticated and reliable Conflict Detection and Resolution algorithms.

In an Airborne Self-Separation environment, each aircraft is itself responsible for taking the measures necessary to avoid other aircraft. Each aircraft is simultaneously pursuing its own route, maximizing its own optimality function, be it fuel consumption, timeliness, or some other factor derived from airline business considerations. In a sparsely populated airspace, the action of a single aircraft seldom influences other aircraft's route, therefore the consequence of actions taken by a single aircraft are limited to the aircraft itself. On the contrary, in a crowded airspace any maneuver of a single aircraft (to avoid another aircraft, or to achieve its goal with respect to its own optimality function) may produce a domino effect of corrective or avoidance maneuvers by other aircraft. Thus, a single, "locally optimal" maneuver may produce an overall degradation of the system in terms of average delays, fuel consumption or some other parameter that measures a desirable property of the whole set of aircraft involved. Thus conflict resolution must guarantee both safe separation between aircraft, and the optimization of some parameter encompassing the whole set of aircraft in the affected airspace. The combination of these two tasks, especially in airspaces where tens of planes are flying, clearly lies beyond the possibilities of any human-centered centralized control.

Collective Intelligence: A New Conceptual Framework

We plan to address the problem of conflict resolution in an Airborne Self-Separation airspace with an approach inspired by the emerging framework of Collective Intelligence (COIN, see [7]). This formal framework is designed to address situations where there is no centralized control and where there is a clear global objective function that needs to be optimized. More specifically, we might isolate the following characteristics for problems which can be tackled by Collective Intelligence techniques:

—the problem can be modeled by means of a collection of agents;
—agents do not need centralized control;
—each agent acts independently from each other, pursuing its own

specific objective, and maintaining only a limited vision of the overall situation;

—there is a well-specified (i.e. mathematically defined) global objective;

—agents need to coordinate to achieve this objective;

—agents are adaptive, i.e. they dynamically react to changing conditions of the environment.

Within this framework, the "intelligent" behavior of the whole system does not result from careful top-down off-line algorithmic planning; it is rather an emergent property of the system as a whole, in which each agent, while pursuing its own limited goals and locally interacting with other agents, maximizes the given optimality function for the system. This new emerging paradigm has already been proved successful in a number of other technological domains (e.g. packet routing, transport logistics, automated car driving, see [7] for a survey on different applications). Benefits resulting from this approach could be manifold. First of all, by relying on computation distributed among several independent agents, the risk of catastrophic failures for the system is largely subdued. In the traditional, centralized approach to control, a single failure occurring in the central control unit might "leave in the dark" any process (in our case, any plane) which depends on it. On the contrary, the intrinsically distributed and adaptive nature of Collective Intelligence agents would allow for a higher degree of robustness and dependability for the whole system. The failure of a single node does not entail the failure of the whole system, and other agents can more or less quickly adapt to the new configuration. Distributed computing also could allow for more efficient computation of possible solutions to the optimization problem. Although the solution provided might be only suboptimal, it seems to outperform approaches purely relying on human intervention, and it can be achieved with cheap computational resources in real time.

It is apparent that the problem of conflict resolution for Airborne Self-Separation is particularly well suited to be tackled within the Collective Intelligence paradigm. In an Airborne Self-Separation environment, every aircraft acts as an agent pursuing its own goals (shortest route, fuel consumption, timeliness) as independently as possible from other agents engaging the same airspace. However, when a potential conflict occurs, the planes involved must coordinate among themselves and with others in their immediate vicinity to agree on conflict resolution maneuvers that are minimally expensive for the whole set of aircraft. As we will see in the next section, it is possible to rigorously address this issue within the framework of Collective Intelligence, by encoding some form of "altruistic behavior" among the agents, therefore allowing for a compromise between local optimal choices and overall performance of the system, separation assurance still being primary.

A brief Review of some Conflict Resolution Techniques

A variety of interesting and effective techniques have been proposed to resolve conflicts in ATC. Most of them are described in the following reviews [8], [9] and [10]. There are also other very innovative automated approaches: Krozel et al. [11] describe three different algorithms, one centralized and two distributed.

The centralized or ground based approach determines the set of conflicts that would occur in the next eight minutes if no corrective actions were taken. Aircraft are then partitioned into clusters; all pairs of aircraft with a conflict will be in the same cluster. The aircraft within each cluster are then ranked using some permutation sequence. The highest ranking aircraft is allowed to continue on its path without heading change. A conflict-free trajectory must then be found for the second aircraft in the sequence, and this continues until a conflict-free path has been found for every aircraft in the cluster; or until the search fails to find an acceptable flight path, in which case the algorithm restarts with a different ranking and permutation sequence.

In the two decentralized strategies, each aircraft resolves its own conflicts as they are detected. Multiple conflicts within the eight-minute look-ahead window are resolved in a sequential pair-wise fashion, either passing in front of or behind the conflicting aircraft. A so called 'myopic' strategy selects the alternative that requires the smallest heading change. Then, a second look-ahead strategy further examines the selected maneuver to ensure that it does not produces a conflict that is earlier in time than the original conflict. If such a conflict is detected, the strategy tries the alternative maneuver, and then small heading offsets from the original choice if needed.

Dugail, Feron, and Bilimoria analyze a conflict resolution scheme in a scenario consisting of two perpendicular flows of air traffic that intersect at a fixed point [12]. Upon entering the airspace, each aircraft makes a single instantaneous heading change—the minimum required to avoid conflicts with those aircraft already present. After the maneuver, each aircraft moves in a straight line to its destination. Simulations modeled regular and random aircraft arrivals. For this scenario, the authors prove that this conflict resolution scheme does not result in arbitrarily large avoidance maneuvers and is therefore stable.

Resmerita, Heymann, and Meyer describe a resource allocation approach to collision detection and resolution that partitions the airspace into distinct cells that may be occupied by only one aircraft at a time, ensuring separation [13]. These cells become the vertices of an undirected graph whose edges are paths between cells. Agent trajectories are directed, timed graphs that overlay the airspace graph. Before an aircraft enters the system, it registers itself with a central controller that maintains a list of all aircraft and their optimal trajectories (an aircraft may have more than one). The controller then distributes resources (timed access to cells) as aircraft request them. The algorithm assumes non-cooperative, greedy agents that do not communicate during conflict resolution. Conflicts arise when an agent requests a resource that has already been allocated. The agent first tries alternate optimal paths, and then the controller requests other agents holding disputed resources to free them by choosing alternate paths. The approach has two possible outcomes: either the resources for an optimal path can be obtained, or the agent is not allowed to enter the system. This algorithm is optimal with respect to agent path quality (as determined by each agent), but it is computationally intensive and requires a centralized controller.

Pallottino et al. [14] describe a geometric approach to collision detection and resolution. Path planning is modeled as a set of linear constraints on either velocity or heading changes to be optimized with respect to total flight time and course deviations, respectively. The authors prove that a decentralized adaptation of the algorithm is possible, given a proper look-ahead distance.

Blom et al. [15] evaluate through Monte Carlo simulation an airborne self separation concept that has been developed for use in en route traffic conditions, such as encountered over the Mediterranean area. For three different encounter scenarios, probabilities for violating minimum separation and for near-mid-air and mid-air events are estimated by applying powerful novel Monte Carlo simulation approaches for rare event estimation. The conflict detection and resolution approach is mainly procedural and it is intentionally designed to solve multiple conflicts one by one rather than by applying a joint potential field resolution. Crew decides which option to execute. All aircraft use the same resolution algorithm, and all crews apply the same procedures. The paper shows several quantitative risk estimates and presents an analysis of these results in terms of safety. In [16], an exact solution based on geometrical concept is found for some simple scenarios and then generalized to more general situations (with some restrictive hypothesis). This technique is mathematically very well grounded, but the applicability in more realistic and in completely arbitrary situations is not discussed.

In [17] very interesting and innovative concepts are developed.

Finally, in [18] are described simulations of integrated air and ground operations conducted at NASA Ames Research Center to evaluate three Distributed Air/Ground Traffic Management (DAG-TM) Concept Elements : En Route Free Maneuvering, En Route Trajectory Negotiation, and Terminal Arrival Self-Spacing.

Excluding valuable contributions such as the ones in [19], [20] and [21], the recent research mainstream in ATM has focused on solutions for conflict avoidance that are based on fixed sets of rules that dictate actions based on situational geometry. This approach can achieve surprising performances in a fixed scenario, such as two intersecting flows of aircraft described in [12], but good performance in arbitrary situations may not always be guaranteed. A suitable distributed solution should have several characteristics: (1) the aircraft must coordinate their decisions in avoiding collisions; (2) the avoidance maneuvers must be generally applicable and not limited to specific geometric situations; (3) the avoidance maneuvers must ensure an overall traffic optimization, in terms of aircraft trajectories and global delays; (4) the approach must be realistic and not oversimplify the problem; (5) the solution must scale to high traffic densities.

SATISFICING GAME THEORY BASICS

From seminal works, it appears that Collective Intelligence algorithms can effectively and efficiently resolve most conflicts, even with high traffic densities. In particular, Satisficing Game Theory (see [22]) is based upon the following key concepts: (1) rather than seeking the optimal solution for the group and all agents simultaneously, satisficing agents simply obtain an adequate solution; (2) agents determine the adequacy of a choice by comparing two different utility functions representing respectively the benefits (selectability) and the costs (rejectability) of the considered choice. Thus, a choice is acceptable if the benefits outweigh the costs. This approach allows individuals or groups to condition their own preferences on the preference of others, therefore enabling a truly collaborative approach to conflict resolution. By exchanging the information and goals of each agent, cooperation among agents can be realized even in a completely distributed system.

The 2D Case

First we describe in detail the simple 2D model, used to test the Satisficing approach. We assume that all aircraft fly at the same altitude and at the same constant speed. The first step in applying SGT is to create influence flows that describe relationships between aircraft. In order to do that, at each time step, each aircraft exchanges information with all other aircraft within a 50 nmi radius. This information includes: current position, destination, actual heading, flight time and delay (relative to an unobstructed straight line flight).

Each aircraft ranks the set of viewable aircraft (within 50 nmi) according to the broadcasted information, with a pre-defined rule. In our simulations we used different definition. One possible is to use the three parameters: proximity to destination, delay (greater delay, higher rank), and flight time (longer flight time, higher rank). We also used a 'random' order, or a smaller number of parameters. Thus we have different concepts of 'priority.'

For each aircraft X_i, we define the *priority set* P_i, i.e., the set of all viewable aircraft with higher ranking than X_i, that could conflict with X_i itself (for some heading choice). This set is also known as the set of X_i's *parents*.

At each second, each aircraft has to choose one of five directional options: flying straight, moderate turn to the left, sharp turn to the left, moderate turn to the right and sharp turn to the right.

On the basis of the exchanged information, each aircraft computes the rejectability and selectability of each choice, and selects the directional option that maximizes the difference between the selectability and rejectability utilities:

$$u^i_{l*} = \arg\max_{u^i_l \in U}(p_{S_i}(u^i_l) - p_{R_i}(u^i_l)) \tag{1}$$

Loosely speaking, each "satisficing agent" is looking for the highest gain, with lowest risk. In the ATC context, the rejectability function reflects concern about the safety of the considered aircraft: it indicates the degree of 'risk' of each directional option, i.e. how much it is likely to lead to conflicts with higher priority aircraft.

In order to compute the rejectability, each aircraft compares the linear extension of each of its directional options with linear projections of current headings of all aircraft in its priority set P_i. Each projected conflict adds a weight to that option, depending on distance in time and severity of the conflict (collision or near miss). After all the *parents* are considered, the weight of each option is normalized over the option space.

The mathematical definition of the rejectability increases the (negative) value of the option that leads to conflicts or small separations, with more weight for incidents closer in time. The selectability function reflects goal achievement, namely in our ATC context reaching the destination. In contrast with rejectability, selectability is influenced by the preferences of other agents. Thus, we have two distinct components for the selectability: the base selectability σ_{S_i}, which accounts for agent X_i heading preferences, and ρ_{S_i} which accounts for other viewable agents' preferences.

The base selectability of each directional option is determined by the difference with the desired heading of the aircraft: if an option takes the aircraft more directly to its destination, it will have a higher base selectability.

The parent selectability ρ_{S_i} is computed using the base selectability of all higher-ranked aircraft to approximate their full selectability. We take into account (in a simplified, but effective way) the preference and the 'desires' of all the parents of our aircraft: if one of them strongly prefers a direction that will lead to a separation violation, the current aircraft will lower its selectability for its own directional options that will lead to a conflict. This allows for cooperation and 'conditional altruism,' as previously mentioned. For more details on the computation of these quantities, see Appendix 1.

Finally, the selectability mass function is formed by the convex combination of parent selectability and base selectabilty

$$P_{S_i}(u_l^i) = \lambda\sigma_{S_i} + (1-\lambda)\rho_{S_i} \tag{2}$$

where, in our simulation, we use $\lambda = 1$ if $|P_i| = 0$, otherwise $\lambda = 0.001$.

After computing rejectability and selectability, each agent chooses the heading change to perform, following the decision rule described above. In order to test the decentralized Satisficing algorithm, we consider different performance measures: the most important is the absence of missed separations, called *Separation Assurance*. We compute it as a function of varying traffic densities and of different traffic scenarios.

We also take into account the *System Efficiency* which is defined as the degree to which an aircraft is able to follow its ideal flight path. In order to compute it for the whole system, we perform the overall average

$$SE = \frac{1}{N}\sum_{i=1}^{N} \frac{t_i}{t_i + t_{d_i}} \tag{3}$$

where t_i is the ideal flight time for aircraft i and t_{d_i} is added delay time.

Conflict resolution maneuvers will cause each aircraft to deviate from each ideal path, increasing costs and adding delays. For an algorithm to be successful, conflicts must be avoided while maintaining high efficiency.

Variations and Generalizations

In order to achieve a better realism, we are implementing several generalizations of the original algorithm developed by Hill et al. [1]: the most important is the possibility of performing changes in altitude, modeling a real 3D space. The aircraft can change level at each time step, by adding two other directional options: up and down.

We are still testing the effect of that important change on the algorithm performance: on one side, allowing changes of level is obviously an advantage in avoiding conflicts; on the other side, adding a degree of freedom leads to a higher level of complexity in the interaction among aircraft. Another important variation in our simulator is the introduction of different velocities for different aircraft models, as in the real world. We also include the capability of managing the presence of bad weather conditions or no-flight zones by introducing some forbidden areas and adding a higher weight in the rejectability function for the directional options that will lead to them. We model actual airspace realistically, introducing airport scenarios, real traffic flow data, and interfacing with ENAV simulators.

Finally we will test more deeply the stability and the scalability of this algorithm, considering also another important measure of performance, often used in literature [1]: the System Stability (SS)—a measure of the extent to which conflict resolution maneuvers create new conflicts that will require additional resolution maneuvers.

The *SS* is defined as

$$SS = \frac{|A_1|}{|A_2|} \tag{4}$$

where A_1 is the set of conflict alerts if all aircraft were to fly straight-line paths to their destination and A_2 is the set of conflict alerts arising where conflict resolution maneuvers are employed.

A 3D SOFTWARE DEMONSTRATOR

In order to perform a sequence of quantitative simulations, running the algorithm on a set of test cases, we developed software capable of rendering a three-dimensional animation of arbitrary flight scenarios. As shown in Figure 1, the user can describe a flight scenario by specifying both the global parameters of the simulation (such as proximity radius, near miss radius and collision radius) and the features of each involved flight (such as the starting point, the destination point and the speed). The computation of the simulation is performed offline (i.e. not in real time), and then graphically rendered in playback mode. This allows for both a more precise rendering in terms of timing accuracy and more freedom in terms of different playback options (such as reverse-order or fast-forward at arbitrary speed, or random jump across the whole time-line). A step-by-step playback mode is also available. During the playback, besides the graphical visualization, it is also possible to display the complete state of the ongoing flights in numerical terms (position, flight time,

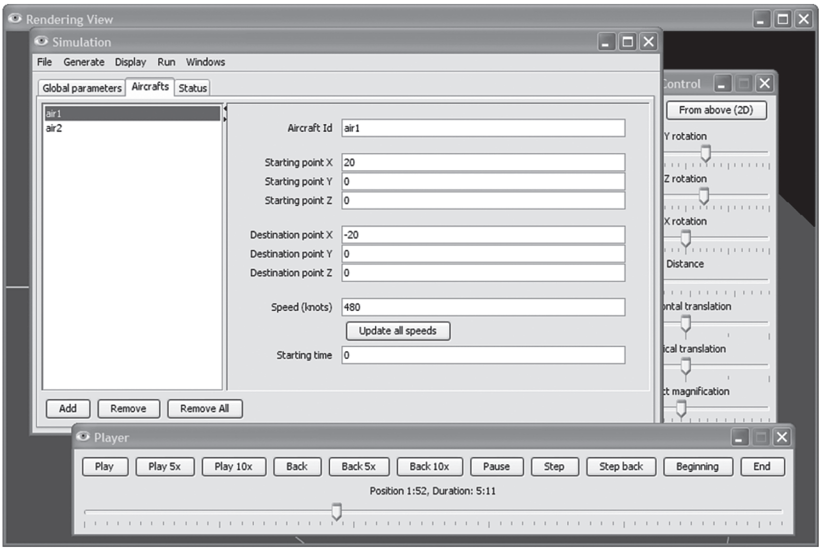


Figure 1. Prototype control panel.

distance from the target, delay, priority). The time resolution of the rendering is the same as the simulation: one second in the simulation time.

The 3D visualization can be rotated along each axis, and uniformly scaled; a special predefined setup display a bi-dimensional view "from above," which resembles a traditional "radar view," and may be clearer for the viewer for those simulation scenarios where the flight height is less significant (see Figure 2). Optionally, proximity areas,

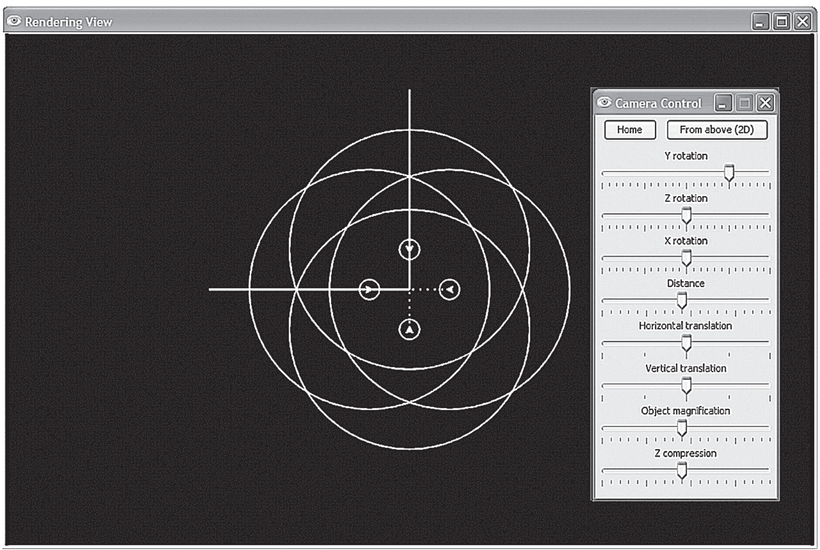


Figure 2. Prototype simulation window (2D view).

near miss areas, and conflict areas may be depicted as colored circumferences; optimal and actual path may be displayed as well. Conflict sets may be highlighted when they arise.

Some typical starting test configurations (such as a random distribution on a spherical surface, or different cross patterns) can be automatically generated.

The software is written using the Java programming language (Java SE 6) in order to ensure maximum portability among different computing platforms. 3D rendering, shown in Figure 3, is performed using open source software libraries able to interface any OpenGL-capable graphic hardware. The program architecture allows for different algorithms to be implemented and tested side-by-side.

Some simple statistics, such as the number of occurred near-misses, the number of collisions, and the global efficiency (as defined above) are collected and can be displayed and exported. A simulation configuration can be stored on a file and later reloaded and re-rendered.

CURRENT RESEARCH

Test Scenarios

In these preliminary simulations, we still concentrate on the 2D case, deliberately forcing high airplane densities at the same altitude in order to stress the agents by increasing the complexity of interaction between them.

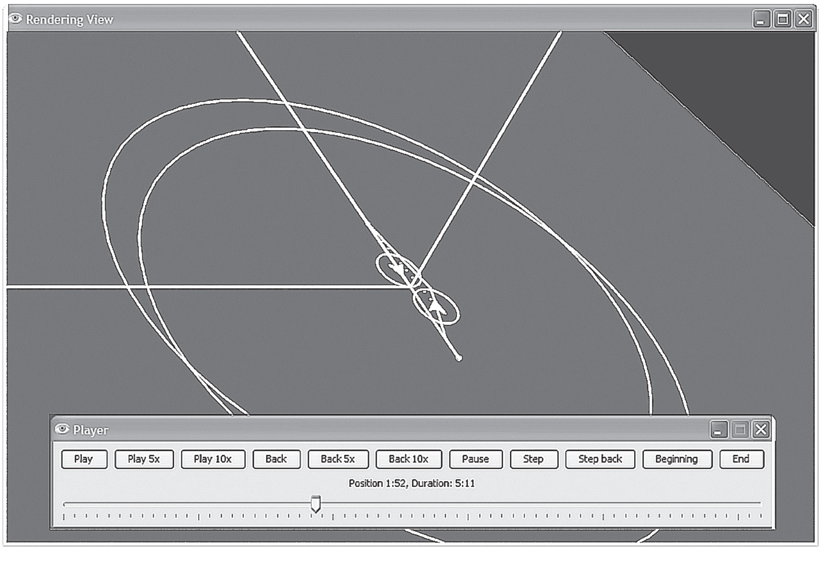


Figure 3. Prototype 3D visualisation and 'player' functionality.

We test the robustness of our algorithm on a very unrealistic and 'pessimistic' scenario: all the flights go through a choke point at the same time. A predetermined number of aircraft are distributed around a circle with a radius of 50 miles. Each aircraft has its destination at the exact opposite side of the circle, and it will disappear when reaching its destination. This scenario represents a challenge for any conflict resolution algorithm: as the number of aircraft increase, there is a corresponding increase in the traffic density (because the circle has a fixed radius). Applying SGT we obtained good results both in terms of efficiency and in absence of missed separation. In Figure 4, we present 4 snapshots of the choke point scenario for 12 aircraft. The results are noted in Figure 5 (14 aircraft, zero missed separations, 0.674 efficiency). We notice in Figure 6 that, as density increases the SGT exhibits a graceful degradation with respect to efficiency.

Another test is on a 'perpendicular flow scenario', where two linear traffic flows intersect at right angles, one moving from left to right and the other moving from top to bottom, as in Figure 7. For this scenario, we use a proximity radius of 100 nmi. As the flows approach the intersection point, the aircraft change their direction to avoid violations of the 5 nmi separation distance. There were no missed separations for an arbitrary large number of aircraft (keeping a fixed initial distance between aircraft of 8 nmi), but increasing the number of aircraft, there is a consequent loss of efficiency.

In Figure 8 we report some results about the frequency of separa-

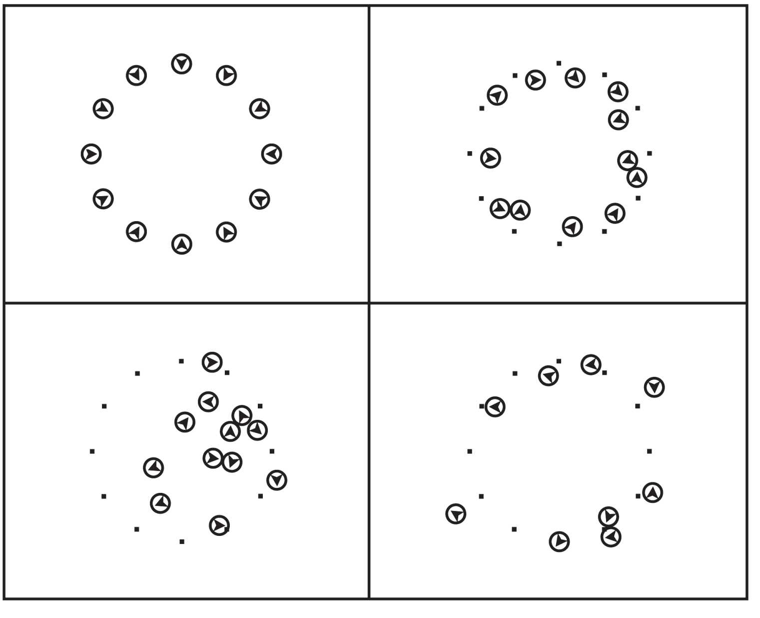
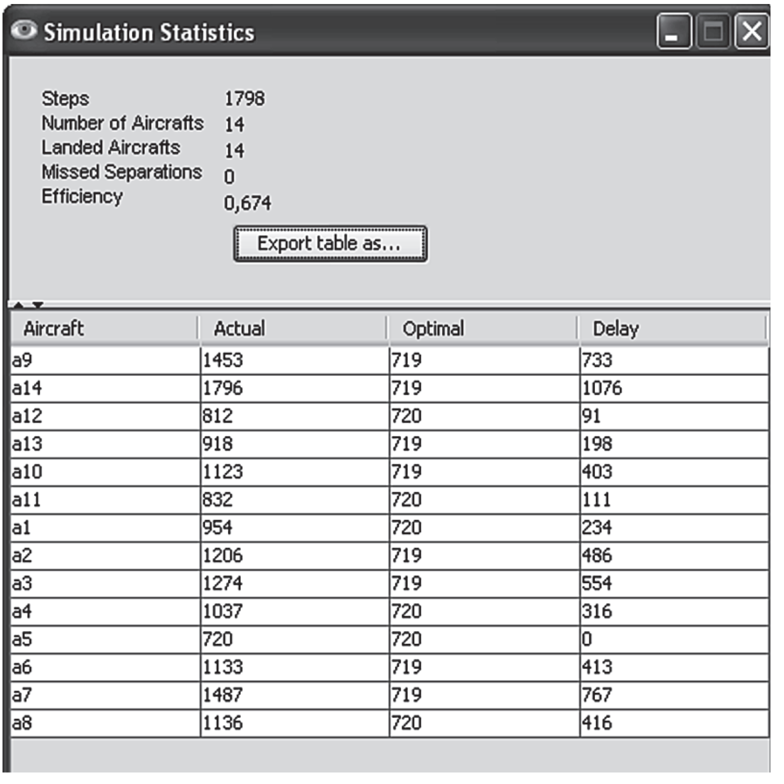


Figure 4. Snapshots of the choke point scenario with 12 aircraft.



Aircraft	Actual	Optimal	Delay
a9	1453	719	733
a14	1796	719	1076
a12	812	720	91
a13	918	719	198
a10	1123	719	403
a11	832	720	111
a1	954	720	234
a2	1206	719	486
a3	1274	719	554
a4	1037	720	316
a5	720	720	0
a6	1133	719	413
a7	1487	719	767
a8	1136	720	416

Figure 5. Table summarizing statistical results for choke point scenario with 14 aircraft.

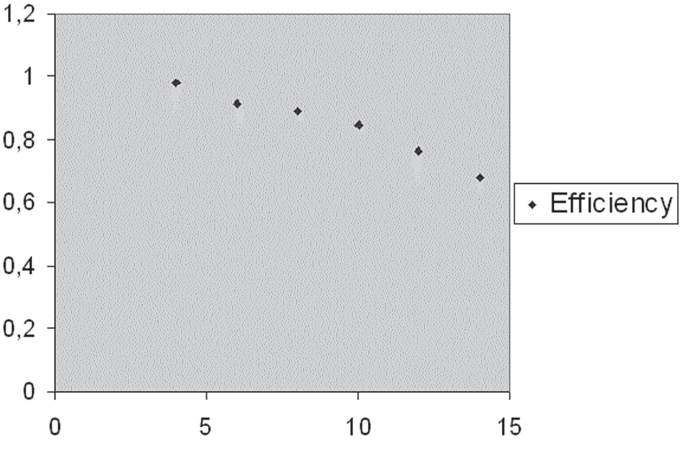


Figure 6. Efficiency vs number of aircraft.

tion violations, efficiency, and individual paths and delays for a ‘perpendicular flow’ scenario with 12 aircraft.

Finally, evaluate the algorithm using randomly generated traffic with high densities (e.g. more than twice the actual European aver-

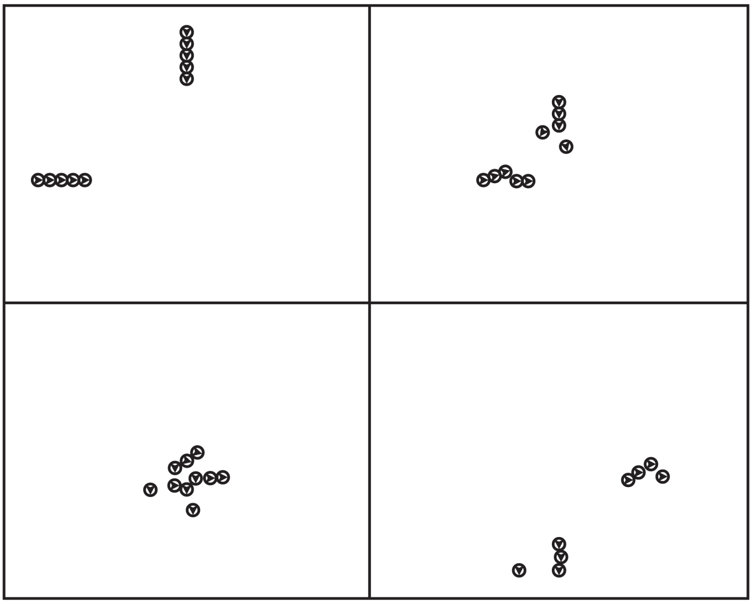
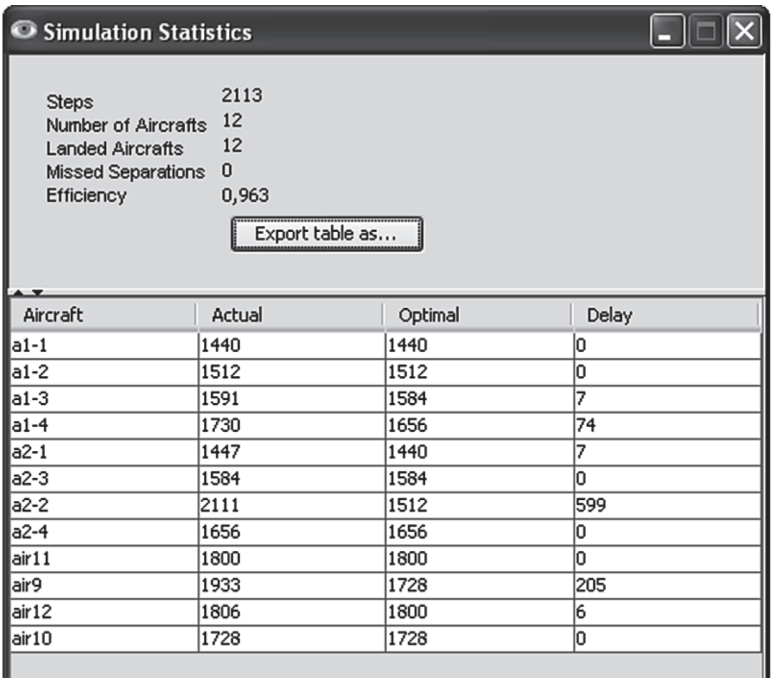


Figure 7. Snapshots of the perpendicular flows scenario with 10 aircraft.



Simulation Statistics

Steps	2113
Number of Aircrafts	12
Landed Aircrafts	12
Missed Separations	0
Efficiency	0,963

Export table as...

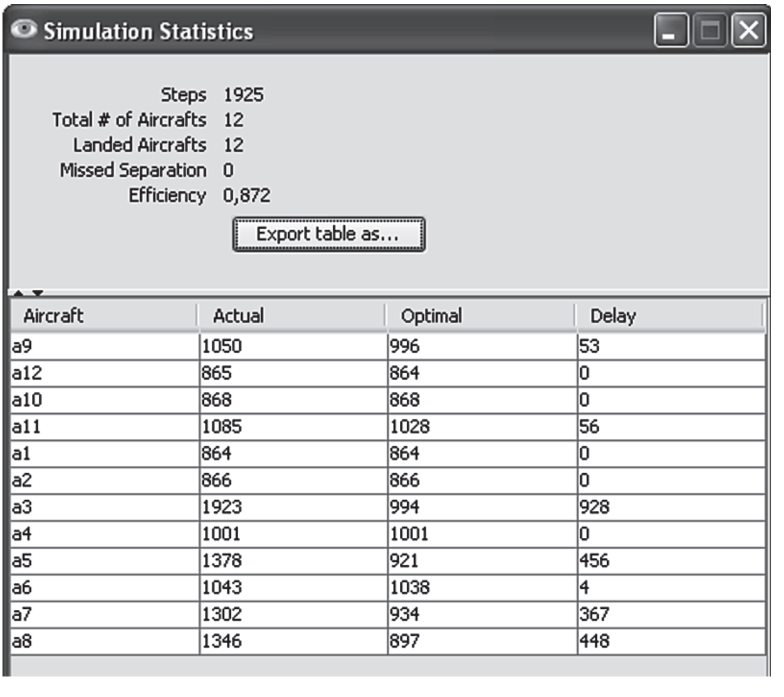
Aircraft	Actual	Optimal	Delay
a1-1	1440	1440	0
a1-2	1512	1512	0
a1-3	1591	1584	7
a1-4	1730	1656	74
a2-1	1447	1440	7
a2-3	1584	1584	0
a2-2	2111	1512	599
a2-4	1656	1656	0
air11	1800	1800	0
air9	1933	1728	205
air12	1806	1800	6
air10	1728	1728	0

Figure 8. Table summarizing statistical results for the perpendicular flows scenario with 12 aircraft.

age density). In the Random Flight scenario, that reflects open airspace with no obstacles other than other aircraft, airplanes appear at random points on a square. They are assigned a random destination point on the opposite side of the square itself. As we can see in Figure 9, there are no missed separations up to 12 aircraft in 10,000 nmi squared (for 5 different generated samples). In Figure 10 there is snapshot of the random scenario with 12 aircraft, with the actual paths, generated by SGT algorithm, in grey. Later we will compare the performance of our conflict resolution software tool with other tools based on different algorithms, e.g. Hoekstra [7] and Krozel [11], in order to identify the strengths and weaknesses of our algorithm for the random scenario with 12 aircraft.

Real Traffic Datasets

ENAV has already provided access to some realistic traffic flows, in particular data from the Fast Time Simulation of a whole day with a very high traffic density, i.e. 09/07/2006, for the Padova Area Control Center. The input files are in csv format and they contain, for each flight, the following information: flight ID, airports of departure and arrival, aircraft model, average level during the flight and then all fixed points in aircraft route, with level and time at which the aircraft reached them. We will use this first set of data provided by ENAV as a baseline to test our algorithm, first concentrating on interesting and typical cases involving a small number of aircraft.



Simulation Statistics

Steps 1925
Total # of Aircrafts 12
Landed Aircrafts 12
Missed Separation 0
Efficiency 0,872

Export table as...

Aircraft	Actual	Optimal	Delay
a9	1050	996	53
a12	865	864	0
a10	868	868	0
a11	1085	1028	56
a1	864	864	0
a2	866	866	0
a3	1923	994	928
a4	1001	1001	0
a5	1378	921	456
a6	1043	1038	4
a7	1302	934	367
a8	1346	897	448

Figure 9. Table summarizing statistical results for the random scenario with 12 aircraft.

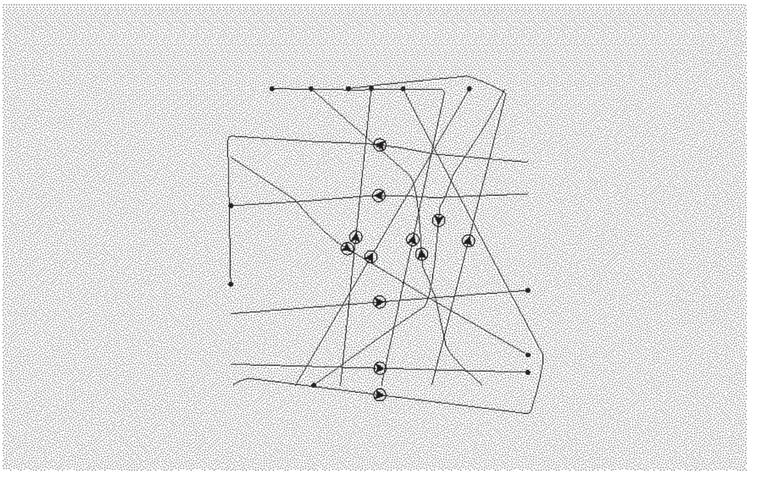


Figure 10. Snapshot of random square scenario with 12 aircraft.

Then we will try to manage with our algorithm all the traffic during the whole day, comparing our simulation with the ENAV Fast Time Simulation.

In order to test and improve the simulation platform, we plan to get further data on potential conflict situations that have been dealt with by human controllers in a Real Time Simulation (i.e. the Real Time Simulation held in Rome for the 'European ATM Validation Platform AMAN Project' in 2004) and compare the results with the solution proposed by our algorithm (with respect to separation assurance, delays, and the emergence of different traffic patterns in an Airborne Self-Separation environment).

OPEN ISSUES

The introduction of automated conflict resolution tools will lead to new challenges in ATM. Among the open issues that must be addressed in further work are the overall safety achieved, with particular attention to any issues that might arise in data exchange using available imperfect communication channels. The robustness of the SGT approach must be examined in the presence of equipment failures, communication failures, or malicious behavior of an aircraft.

Problems arising from incomplete or missing information or uncooperative behavior may be addressed using randomized agreement protocols [23]. Integrating the SGT tool with consensus algorithms may permit each aircraft to make an acceptable choice in the presence of partial or incorrect information by agreeing on the 'right' piece of information with the other compliant participants. Human factor and organizational issues must also be considered, with careful consideration of the transition pathway between current and future operational concepts.

CONCLUSION

As traffic densities increase, the need for new algorithms that automate decision making will continue to grow. The Satisficing approach is mathematically well grounded and provides a natural mechanism for cooperation that optimises the overall traffic trajectories, while ensuring the minimization of risks. Our preliminary simulation experiments confirm the flexibility and the good performance offered by Satisficing Game Theory in an Airborne Self-Separation environment. We are exploring new extensions of the algorithm that will take into account more realistic conditions, and we will compare our simulation results with real cases handled by controllers.

APPENDIX

For the sake of simplicity, we assume that all aircraft fly at the same altitude and at the same constant speed. At each time step, each aircraft exchanges information with all other aircraft within a 50 nmi radius. This information includes: current position, destination, actual heading, flight time, delay (relative to an unobstructed straight line flight). At each second, each aircraft chooses one of five directional options: flying straight, moderate turn to the left, sharp turn to the left, moderate turn to the right, sharp turn to the right (see Figure 11).

Each aircraft ranks the set of viewable aircraft (within 50 nmi)

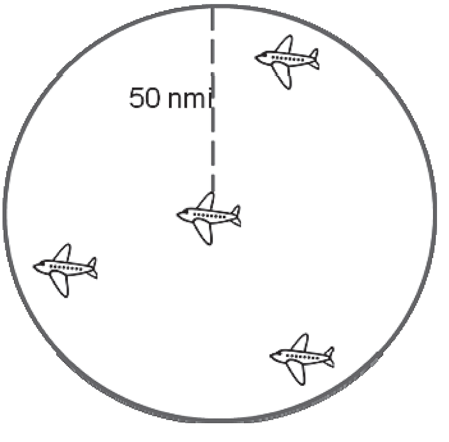


Figure 11. Set of neighbors of the considered aircraft.

according to the information received. For each aircraft X_i, we define the priority set P_i, i.e., the set of all viewable aircraft with higher ranking than the considered aircraft that could conflict with the aircraft itself (for some heading choice). This set is also known as the set of X_i's parents.

The rejectability function reflects concern about the safety of X_i. Each aircraft compares the linear extension of each of its directional options with linear projections of current headings of all aircraft in its priority set P_i. Each projected conflict adds a weight to that option, depending on distance in time and severity of the conflict. After all the parents are considered, the weight of each option is normalized over the option space.

More in details: let r_c and r_{nm} denote collision radius and near miss radius and let u_c represent current heading.

Let $d(i,k)$ represent the projected distance from X_i current position to the point of closest approach to X_K.

Let $d_{min}(i,k)$ be the shortest distance between X_i and X_K on their projected paths.

Then, we compute:

$$p_{R_i}(u_l) \propto \sum_{X_k \in P_i} W_R(X_k(u_c), X_i(u_l)) \tag{A1}$$

Where the weight W_R is defined by:

$$W_R(X_K(u_c), X_i(u_l)) = \begin{cases} 2\alpha \leftarrow d_{\min}(i,k) \leq r_c \\ \alpha \leftarrow r_c < d_{\min}(i,k) \leq r_{nm} \\ 0 \leftarrow otherwise \end{cases} \tag{A2}$$

and the α parameter is given by:

$$\alpha = \begin{cases} \left(1 + \dfrac{r_{nm} - d_{\min}(i,k)}{r_{nm}}\right)\left(\dfrac{1}{d(i,k)}\right)^{\beta} \leftarrow d(i,k) < 3r_{nm} \\ \left(\dfrac{1}{d(i,k)}\right)^{\beta} \leftarrow d(i,k) \geq 3r_{nm} \end{cases} \tag{A3}$$

The above construction increases the rejectability of options that lead to conflicts or small separations, with more weight for incidents closest in time. The selectability function reflects goal achievement. In contrast with rejectability, selectability is influenced by the preferences of other agents. Thus, we have two distinct components for the selectability: the 'base selectability' which accounts for the considered aircraft heading preferences and the 'parent selectability' which accounts for other viewable agents' preferences. Thus, the selectability mass function is formed by the convex combination:

$$p_{S_i}(u_l) = \lambda\sigma_{S_i} + (1-\lambda)\rho_{S_i} \tag{A4}$$

with $\lambda = 1$ if $|\mathrm{Pi}| = 0$, otherwise $= 0.001$.

Let us compute now the 'base selectability' σ_{S_i}. The first step is assigning a rank $r(u_l) = 1, \ldots, 5$ to each heading option according to distance between the direct heading to destination and the considered heading.

A weight $w_s(u_l)$ is then assigned as a function of the ranking and of the distance:

$$w_s(u) = \begin{cases} 4 \Leftarrow r(u_l) = 1 \\ 2 \Leftarrow r(u_l) = 2 \vee (3 \wedge 2.5° < |u_{dir} - u_l| < 5°) \\ 1.5 \Leftarrow r(u_l) = (3 \wedge 5° < |u_{dir} - u_l|) \vee (4 \wedge |u_{dir} - u_l| < 5°) \vee (5 \wedge |u_{dir} - u_l| < 5°) \\ 1 \Leftarrow r(u_l) = (4 \wedge 5° < |u_{dir} - u_l|) \vee (5 \wedge 5° < |u_{dir} - u_l|) \end{cases} \tag{A5}$$

Weights are then normalized to form the 'base selectability'. Let us now compute the 'parent selectability' ρ_{S_i}, in a simplified model, which scales better in complexity and performs nearly as well as the full model in [21, 22].

We partition the priority set into $|\mathrm{U}| = 5$ subsets $S_1^g, \ldots, S_5^g$, according to each aircraft's preferred heading option, as determined by 'base selectability'.

Let $W_g(k)$ denote the cardinality of S_k^g.

For each subset S_k^g, calculates a matrix of weights (so we have 5 different matrices):

$$W_{ik}(u_l^i, u_m^k) = \sum_{X_j \in S_k^g} W_S(X_i(u_l^i), X_j(u_m^k)) \tag{A6}$$

where $l = 1, \ldots 5$ and $m = 1, \ldots, 5$.

We define W_S as:

$$W_S = \begin{cases} 1 \leftarrow d_{\min}(i,k) > r_{nm} \\ 0 \leftarrow d_{\min}(i,k) \le r_{nm} \end{cases} \tag{A7}$$

Thus, all pairs of heading options which do not conflict are assigned a weight of one. The column of the matrix are then normalized such that:

$$\sum_{u^i \in U} W_{ik}(u^i, u_m^k) = 1 \tag{A8}$$

The simplified conditional selectability is calculated as:

$$\rho_{S_i | S_1^g, \ldots, S_5^g}(u_l^i | u^1, \ldots, u^5) \propto \sum_{k=1,\ldots,5} W_g(k) W_{ik}(u_l^i, u^k) \tag{A9}$$

Thus, the marginal selectability is calculated as:

$$\rho_{S_i}(u_l) = \sum_{u^1 \in U} \sum_{u^2 \in U} \ldots \sum_{u^5 \in U} \rho_{S_i | S_1^g \ldots S_2^g}(u_l^i | u^1, \ldots, u^5) \sigma_{S_1^g}(u^1) \ldots \sigma_{S_5^g}(u^5) \tag{A10}$$

In this case, each partition is treated as an individual aircraft l (that influences the considered aircraft), so there are only values of the conditioning direction vector. Moreover we use simply, 'base selectabilities', as marginals for the 5 partitions of the parents' set. So the computation of is much easier than in the full model. Finally, we compute the selectability mass function as the convex combination of the two components (as said above).

After computing rejectability and selectability, each agent (aircraft) has to choose the heading change to perform. In order to do that, each aircraft selects the heading option that maximizes the difference between the selectability and rejectability utilities:

$$u_l^i = \arg\max_{u_l^i \in U}(p_{S_i}(u_l^i) - p_{R_i}(u_l^i)) \tag{A11}$$

Thus, each 'satisficing agent' is looking for the highest gain, with lowest risk, taking also into account preferences of other agents, thus obtaining a solution that could be effective for the whole system.

ACRONYMS

ATC	Air Traffic Control
ATM	Air Traffic Management
COIN	Collective Intelligence
ENAV	Ente Nazionale per l'Assistenza al Volo (Italian Service Provider)
DAG-TM	Distributed Air-ground Traffic Management
SE	System Efficiency
SGT	Satisficing Game Theory
SS	System Stability

REFERENCES

[1] Hill, J., Johnson, F., Archibald, J., Frost, R., Stirling, W. (2005), "A Cooperative Multi-Agent Approach to Free Flight", *AAMAS 2005—4th Joint Conference on Autonomous Agents and Multi-Agents Systems*, Utrecht, Netherlands, July.

[2] http://www.eurocontrol.int/care-asas/gallery/content/public/docs/poasas71.pdf

[3] Nordwall, B.D. (1995), "Free Fight: ATC model for the next 50 years", *Aviation Week and Space Technology*.

[4] Perr, T.S. (1997), "In Search of the Future of Air Traffic Control", *IEEE Spectrum*.

[5] RTCA Task Force 3 (1995), *Final Report*, RTCA Inc., Washington D.C.

[6] Eurocontrol (1998–1999), *Annual Report*, Eurocontrol HQ, Brussels, Belgium.

[7] Wolpert, D.H., Tumer, K. (2006), *An Introduction to Collective Intelligence*, NASA-ARC-IC-99–63.

[8] Hoekstra, J.M., Ruigrok R.C.J., Gent R.N. (2000), "Free Flight in a Crowded Airspace", *3rd ATM R&D Seminar*, Napoli, Italy.

[9] Kuchar, J.K., Yang, L.C. (2000), "A review of Conflict Detection and Resolution Modeling Methods", *IEEE Transaction on Intelligent Transportation Systems*, Vol. 1, No. 4.

[10] Dimarogonas, D.V., Kyriakopoulos, K.J. (2003), *Inventory of Decentralized Conflict Detection and Resolution System in Air Traffic*, Deliverable D6.1, Hybridge Project.

[11] Krozel, J., Peters, M., Billimoria, K.D., Lee, C., Mitchell, J.S.B. (2001), "System performance characteristics of centralized and decentralized air traffic separation strategies", *4rd ATM R&D Seminar*, Santa Fe, New Mexico.

[12] Dugail, D., Feron, E., Bilimoria, K. (2002), "Stability of intersecting aircraft flows using heading change maneuvers for conflict avoidance", *Proc. of the American Control Conference*, May.

[13] Resmerita, S. and Heymann, M. (2003), "Conflict Resolution in Multi-Agent Systems", *IEEE Conference on Decision and Control*, pp. 253742.

[14] Pallottino, L., Feron, E. M., and Bicchi, A. (2002), "Conflict Resolution Problems for Air Traffic Management Systems Solved with Mixed Integer Programming", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 3, No. 1.

[15] Hwang, I., Kim, J., Tomlin, C. (2007), "Protocol-based conflict resolution for Air Traffic Control", *ATC Quarterly*, vol. 15–1.

[16] Blom, H.A.P. , Obbink, B.K., Bakker G.J. (2007), "Safety risk simulation of an airborne self separation concept of operation", *Proceedings 7th AIAAATIO Conference*.

[17] Wing, D. J. (2005), "A Potentially Useful Role for Airborne Separation in 4D-Trajectory ATM Operations", *AIAA 5th Aviation, Technology, Integration, and Operations Conference (ATIO)*, Arlington, VA, USA, Sept.

[18] Lee, P.U., Mercer, J.S., Martin, L., Prevot, T., Shelden, S., Verma, S., Smith, N., Battiste, V., Johnson, W., Mogford, R., Palmer, E. (2003), "Free Maneuvering, Trajectory Negotiation, and Self-Spacing Concepts in Distributed Air-Ground Traffic Management", *AA/Eurocontrol, ATM Seminar,* Budapest, Hungary.

[19] Barhydt, R. and Kopardekar, P. (2005), "Joint NASA Ames/Langley Experimental Evaluation of Integrated Air/Ground Operations for En Route Maneuvering", *The Sixth International ATM R&D Seminar ATM*, Baltimore, MD, USA, July.

[20] Denery, D.G., Erzberger, H., Davis, T.J., Green, S.N., and McNally B.D. (1997)," Challenges of air traffic management research—Analysis, simulation, and field test", *AIAA Guidance, Navigation, and Control Conference*, New Orleans, LA, August.

[21] McNally, D., Erzberger, H., Bach, R., Chan, W. (1999), ′A Controller Tool for

Transition Airspace,′ *AIAA-99–4298, Guidance Navigation and Control Conference*, Portland, OR, Aug. 9–11.
[22] Stirling W.C. (2003), *Satisficing Games and Decision Making: With Applications to Engineering and Computer Science*, Cambridge University Press, UK.
[23] Moniz, H., Ferreira Neves, N., Correia M., and Verissimo, P. (2006),” Randomized Intrusion-Tolerant Asynchronous Services”, *University of Lisbon Technical Paper*, Lisbon, Portugal.

BIOGRAPHIES

Francesco Bellomi is a Computer Scientist, with a PhD from the University of Verona. He was International Fellow at SRI International (Menlo Park, CA). He is a senior researcher in the field of knowledge management.

Roberto Bonato is a Computer Scientist with specialization in knowledge representation and machine learning, and junior expert in ATM safety. He has a PhD in Computer Science from the University of Verona and University of Bordeaux (joint programme), with Computational Linguistics as the main area of expertise, and a full University Degree in Computer Science from the University of Verona.

Vincenzo Nanni is a senior research scientist at ENEA. He has a master's degree in Nuclear Engineering and more than 20 years of experience in the field of artificial intelligence, man-machine interface system design for complex systems supervision and control.

Alessandra Tedeschi has a PhD in Applied Mathematics from the University of L'Aquila and Rome 'La Sapienza'. Her research interests are complex systems and agent based models.