USING MULTI-AGENT NEGOTIATION TECHNIQUES FOR THE AUTONOMOUS RESOLUTION OF AIR TRAFFIC CONFLICTS

A Thesis

by

STEVEN WOLLKIND

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2004

Major Subject: Computer Science

USING MULTI-AGENT NEGOTIATION TECHNIQUES FOR THE AUTONOMOUS RESOLUTION OF AIR TRAFFIC CONFLICTS

A Thesis

by

STEVEN WOLLKIND

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Thomas R. Ioerger (Chair of Committee)	Jianer Chen (Member)
John Valasek (Member)	Valerie Taylor (Head of Department)

December 2004

Major Subject: Computer Science

ABSTRACT

Using Multi-agent Negotiation Techniques
for the Autonomous Resolution of Air Traffic Conflicts. (December 2004)
Steven Wollkind, B.A., Williams College
Chair of Advisory Committee: Thomas R. Ioerger

The National Airspace System in its current incarnation is nearing its maximum capacity. The Free Flight initiative, which would alter the current system by allowing pilots to select more direct routes to their destinations, has been proposed as a solution to this problem. However, allowing pilots to fly anywhere, as opposed to being restricted to planned jetways, greatly complicates the problem of ensuring separation between aircraft.

In this thesis I propose using cooperative, multi-agent negotiation techniques in order to efficiently and pseudo-optimally resolve air traffic conflicts. The system makes use of software agents running in each aircraft that negotiate with one another to determine a safe and acceptable solution when a potential air traffic conflict is detected. The agents negotiate using the Monotonic Concession Protocol and communicate using aircraft to aircraft data links, or possibly the ADS-B signal.

There are many benefits to using such a system to handle the resolution of air traffic conflicts. Automating CD&R will improve safety by reducing the workloads of air traffic controllers. Additionally, the robustness of the system is improved as the decentralization provided by software agents running in each aircraft reduces the dependence on a single ground based system to coordinate all aircraft movements. The pilots, passengers, and carriers benefit as well due to the increased efficiency of the solutions reached by negotiation.

To Waldo

ACKNOWLEDGMENTS

I would like to thank Dr. Ioerger for his guidance as my adviser. He has helped me to understand that research happens in small steps, and without this knowledge I would never have finished. I would also like to thank Dr. Valasek and his students for giving me the answers to my many questions about aviation. I also thank Dr. Chen for his membership on my thesis committee.

Finally, thanks to my wife for not killing me when I returned to grad school, and for helping me get through it in one piece.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Motivation

Maintaining separation between aircraft is an important task of air traffic controllers. Controllers are assisted in this task by a national network of jetways to which air traffic is currently confined. These jetways can be thought of as predefined routes, or highways, through the national airspace system. Unfortunately, the jetway system is an inefficient use of airspace, and current research indicates that with continuing increases of air traffic it will cease to be viable. In order to more efficiently use the national airspace the system will need to become more flexible and open. A NASA initiative currently being researched would allow pilots to chose their own direct routes rather than relying on air traffic control and the jetway network.

The NASA initiative is known as "free flight" and it allows pilots to plot their own preferred courses, free from external interference by ground control. This will allow a more efficient use of airspace. Additionally, when aircraft are allowed to take advantage of local weather and traffic conditions, the efficiency of their own flight plans will be greatly increased. [8]

These benefits come at the cost of a significant increase in the difficulty of ensuring safe separation between the aircraft in the airspace. The jetway network, while inefficient, greatly reduces the complexity of the task of maintaining aircraft separation. As the national airspace is shifted from a constrained network of highways in the sky to a system where pilots select their own routes, human air traffic controllers may no longer be able to manually provide this separation.

For this and other reasons there has been a great deal of research into automating

---

The journal model is *IEEE Transactions on Automatic Control.*

the detection and resolution of air traffic conflicts. The majority of this research has approached the problem from a non-cooperative, game theoretic angle. These approaches often solve for solutions that will work in the worst case scenario. While these methods produce viable solutions, they are far from optimal.

Goals

I propose applying multiagent negotiation techniques to the problem of air traffic conflict resolution. By treating the aircraft as agents that are competing for the limited resource of airspace, well known negotiation techniques can be applied to achieve solutions that are of greater efficiency than those obtained by other methods. By allowing aircraft to cooperate and negotiate to resolve their conflicts, two goals will be accomplished. First, the aircraft will be able to resolve impending conflicts without the assistance of a central ground control system. Secondly, inefficient worst case scenario planning which is common in currently proposed systems will be avoided.

This thesis will demonstrate that multi-agent negotiation techniques can be effectively applied to resolve air traffic conflicts and that the solutions generated by these methods are more efficient than those created by traditional methods or alternative, non-cooperative solutions.

Contributions

In spite of the fact that air traffic can be very naturally modeled with multi-agent methods, this approach has not yet realized its full potential in air traffic research. While there have been attempts to model aircraft as self interested agents, they have been primarily focused on non-cooperative methods. One of the primary contributions of this work is a demonstration that cooperative agent methods can be used to create

solutions to problems within the air traffic control domain. This proof of concept will help to establish multi-agent methodologies as a viable tool in the modeling and solving guidance and navigation problems.

CHAPTER II

RELATED WORK

Conflict Detection and Resolution Systems

There has been a great deal of research in the last few years focused on automating the conflict detection and resolution (CD&R) process. While many conflict detection and resolution methods have been proposed, few have applied the techniques of distributed artificial intelligence and multi-agent systems to the problem. Fewer still have applied cooperative methods. Kuchar and Yang[4] provide an overview of 68 of these systems and develop a taxonomy of conflict detection and resolution methods. This taxonomy is a useful guide to the major issues that must be solved by such a system and some of the standard ways in which those issues are approached.

The Kuchar and Yang taxonomy identifies six attributes of CD&R systems. They are: dimensions of state information, method of dynamic state propagation, conflict detection threshold, conflict resolution method, maneuvering dimensions, and management of multiple aircraft conflicts. Within each of these attributes there are several basic classes into which CD&R systems can be classified. [4]

The dimensions of state information of a model are the spatial dimensions which are considered for determining the positions of the aircraft involved. All CD&R systems can be classified into those which consider purely horizontal state, purely vertical state, or the full three-dimensional space. The majority of extant systems fall into the latter category. [4]

Method of state propagation refers to how a model uses current state information to predict future states. Conflict detection and resolution systems must be able to detect future conflicts, predicting when and where they will occur. Clearly, in order to perform these predictions the systems must extrapolate future positions of aircraft

given current state information. There are several possible ways to do this, and Kuchar and Yang outline three fundamental methods. These are nominal, worst-case, and probabilistic. Nominal state propagation simply projects into the future using the current course, speed, and position of the aircraft and does not account for planned or unexpected course changes. Worst-case projection is nearly an opposite of nominal projection. In a worst case projection it is assumed that any possible maneuver might be made by the aircraft. The result is a set of possible trajectories, rather than one. In this method, if any of the possible trajectories would cause a conflict, then a conflict is predicted. Finally, probabilistic projection produces a set of possible trajectories, each with a probability. These trajectories are projected into the future and the probability of a future conflict is determined using the projections and the likelihood associated with each one. It is worth noting that both nominal and worst case projection are subclasses of probabilistic projection. In nominal projection the aircraft follows its current course with probability 1, and in worst-case the aircraft has an equal probability of making any possible course change. [4]

Models are also classified by their methods of determining when a conflict has actually occurred. Some models use a simple distance threshold while others use more complicated rule based systems to determine if two aircraft are actually in danger of having a collision. Additionally, some systems do not define conflict, but simply attempt to provide as much information as possible and assume that some other system will condense that information into a decision about whether or not a conflict is likely to arise. [4]

The methods by which models resolve recognized conflicts vary widely and are categorized into five groups by Kuchar and Yang. The five classes are prescribed, optimized, force field, manual, and no resolution method. Prescribed resolution methods are determined and fixed at the time the system is created. They have the possible ad-

vantage of being able to reduce response time via pilot training, but are generally less efficient due to their "one solution fits all" approach. Optimized resolutions take various cost functions into account and determine a least cost maneuver which provides the required separation between the aircraft. Force field solutions treat the aircraft as a pair of charged particles and use electrostatic equations to determine possible course trajectories. Manual resolution methods allow the pilot to create possible solutions and receive feedback on their viability from the system. These solutions, while being less automated and slower, allow for resolutions that are more appropriately tailored to the particular conditions (terrain, weather) than the previous methods. Finally, some models do not provide a resolution method. These models are primarily concerned with the previously stated problem of conflict detection and assume that a successful resolution will be calculated by some other system. [4]

Models which do provide some form of automated conflict resolution vary in the types of maneuvers that are employed to resolve the conflicts. Classes of maneuvers include turns, vertical moves, speed changes, and so forth. Models can allow for one or several types of resolution maneuvers. Some models allow for combinations within the same resolution. For instance, an aircraft could perform a climbing turn away from a conflict. [4]

Finally, there are two ways in which a conflict resolution system can handle conflicts between multiple (more than 2) aircraft. The model can explicitly handle multiple conflicts, or it can repeatedly apply its standard pairwise resolution. Kuchar and Yang point out that there are some cases in which a pairwise solution alone is not sufficient to avoid conflict and that any model that is to be used in a real traffic situation must have the explicit capacity to resolve conflicts between multiple aircraft. [4]

Tomlin, Pappas and Sastry have published a number of papers on automated air

traffic conflict resolution. They have developed a non-cooperative, distributed system based on game theory. When a potential conflict is detected, the aircraft use a worst case scenario analysis based on the pursuer/evader problem to plot a trajectory that preserves separation regardless of the actions of the other plane. [13][14][15]

Kosecka, Tomlin, Pappas and Sastry have also done work to apply methods from the field of robotics to the problem of air traffic management. There have been a large number of studies into robot motion planning by the robotics community and some of the theoretical methods that have been employed in those studies are applicable for air traffic management. The authors draw the ideas of potential and vortex fields from these classical robotics studies and apply them to the air traffic problem. They also point out that algorithms embedded in time extended configuration space are computationally difficult, and have been shown to be NP-complete under certain circumstances. Additinally, a 'roundabout' method is proposed for handing conflicts between more than two aircraft. The solution employs flight paths similar to a rotary style intersection. [2]

While the game theoretic worst case scenario approach does ensure separation between the planes it does so at the cost of efficiency. Any system which is adversarial in nature cannot produce optimal resolutions, as the solutions are based on worst case analysis. In order to maintain the flexibility to respond to any possible course change by the pursuer, the evader must trade away efficiency. Viable conflict resolutions that are reached through cooperative methods will often be of higher utility to the aircraft involved.

Wangerman and Stengel[16] presented a solution that does treat the aircraft as negotiating agents, but this system also does not take advantage of the possibilities for agent to agent communication. Their principled negotiation system involves agents negotiating indirectly via a centralized controller which is responsible for ensuring the

suggested trajectories do not conflict.

Menon, Sweriduk and Sridhar[5] developed a method which uses cost functions which are similar to those that are found in agent based systems. However, their system is based on quasilinearization optimization methods rather than agent to agent negotiation. In the system aircraft trajectories are defined as sequences of four dimensional waypoints (three spatial dimensions and a temporal dimension) and various parameterization methods. Using the parameterized representation of initial aircraft trajectories, appropriate cost functions, and the sequential quadratic programming method, optimal trajectories are computed. These new trajectories minimize the overall cost of the system subject to the constraint that the aircraft not violate protected zones of other aircraft. The method developed can handle any number of conflicting aircraft.

Prior work at Texas A&M has begun to focus on the idea of using agent systems in air traffic management to resolve conflicts and perform other pilot advisory tasks. [9][10][12] These methods have primarily focused on agents as advisers which monitor the situation and provide the pilot with warnings or recommendations. In general they do not communicate with other external agents or entities aside from the pilot.

Multi-agent Negotiation

Multi-agent negotiation is not a new field. The basic concepts on which the protocols are based originate in economics and game theory and have existed for 50 years. Harsanyi [1] and Nash [6][7] did early work in this field. The multi-agent system field is somewhat younger, but the basic papers in multi-agent negotiation date from the mid 1980's. In spite of the age of the field new methods of applying these principles are constantly arising, and multi-agent techniques are only now gaining

acceptance in some fields for which they are well suited. Recent contributors to this field include Sarit Kraus [3] and Gilad Zlotkin and Jeffrey S. Rosenschein [18]. Zlotkin and Rosenschein in particular are responsible for the negotiation protocol that is used in the method proposed here: the Monotonic Concession Protocol.

Monotonic Concession Protocol

The monotonic concession protocol (MCP) is a simple protocol developed by Zlotkin and Rosenschein[11][17] for automated agent to agent negotiations. The MCP captures the incremental bargaining process that takes place between negotiating parties. The agents iteratively make proposals and counter proposals of progressively less value to themselves until a middle ground is reached that can be agreed upon by both agents.

To begin an explanation of the MCP we must first introduce some notation. We will denote the negotiating agents as A and B. At any moment each agent has some plan that it is following. We will denote the agents' plans as P_A and P_B. A *Deal* will be defined as a pair of plans (P_A, P_B).

Each agent i has a utility function $Utility_i$ which relates a plan of action to a value representing the desirability of that plan for the agent. As it is useful to discuss the utility of deals for the various agents we also define a utility function that operates on deals. Given a Deal $D = (P_A, P_B)$ then $DealUtility_i(D) = Utility_i(P_i)$. For example, $DealUtility_A(D)$ is the value of $U_A(P_A)$.

We will also define the negotiation set NS to be the set of all Deals which are under consideration during the negotiation. The deals in the negotiation set have an additional property that they are *pareto optimal*. A deal $D = (P_A, P_B)$ is pareto

optimal if there is no other deal $D' = (P'_A, P'_B)$ for which

$$DealUtility_A(D') > DealUtility_A(D)$$

and

$$DealUtility_B(D') > DealUtility_B(D)$$

Note that D' is preferred over deal D by *both* agents. Since both agents prefer D' to D the deal D is not pareto optimal and can be safely left out of the negotiation process. Restricting the NS to pareto optimal deals ensures that we need not be concerned at the end of the negotiation that there was some useful deal that was overlooked. NS also contains the *conflict deal*. This is the deal that will be implemented if no agreement is reached. This is usually the set of plans that the agents were operating under before the negotiation began but this need not be the case (indeed, in the realm of air traffic conflicts we will see that this is *not* an acceptable option).

The protocol begins (at $t = 0$) with each agent i proposing the deal from the NS that maximizes the deal utility function for that agent. The proposals made by the agents are simultaneous on each step. Once the proposals for a step have been made, the agents have several options.

First, an agent may accept the deal proposed by the other agent. Let D_A be the deal proposed by agent A and let D_B be the deal proposed by agent B. Agent B would choose to accept the deal offered by A if $DealUtility_B(D_A) \geq DealUtility_B(D_B)$. That is, if agent A offers a deal that is better for B than the deal that B suggested. If both agents would be willing to accept the deal proposed by the other, one of the two deals is chosen at random. The negotiation process ends when one of the agents accepts a deal.

If neither deal is acceptable to both parties the negotiation continues. Each agent

then decides whether to concede or stick to its last offer. This decision is made using a calculation of risk based on the utility values of the various deals that are under consideration. Intuitively, risk is defined as the ratio of how much utility is lost by accepting the offer of the other agent to the amount of utility that is lost by causing a conflict.

Formally, this is defined for agent A as

$$Risk_A = \frac{DealUtility_A(D_A) - DealUtility_A(D_B)}{DealUtility_A(D_A) - DealUtility_A(D_{Conflict})}$$

and similarly for agent B. This relationship is derived in[17].

If $Risk_A > Risk_B$ then agent B should concede on this step. Both agents calculate both risk values and decide accordingly whether to concede or hold. It is important to note here that this definition of risk requires that the agents are aware of each other's utility values for the various deals that are under consideration. This may lead to some concern that an agent could manipulate the negotiation system by misrepresenting its utility values. It has been shown by Rosenschein and Zlotkin that no advantage can be gained in this manner[11].

If an agent is to concede, it must determine which new deal from NS it should propose. The appropriate strategy is to propose the minimum concession such that the other agent will have to concede (if it is not also conceding on this step).

The strategy of beginning the negotiation by proposing the deal of minimal utility to the other agent and proceeding with minimum sufficient concessions when $Risk_{self} < Risk_{other}$ is known as the Zeuthen Strategy. The Extended Zeuthen Strategy is similar but contains some additional logic to take care of a special case that can arise at the end of the negotiation[17].

It has been shown that the Extended Zeuthen Strategy will result in the agents selecting the deal of highest product of utilities from the negotiation set. It has also

been shown that this strategy is in equilibrium, meaning that if agents A and B are negotiating, if agent A uses this strategy then agent B also prefers this strategy to any other.[17]

The fact that this protocol has an equilibrium strategy is of key importance. When a protocol has such a strategy it is not possible for the agents to gain an unfair advantage by seeking alternate strategies. If any agent uses the equilibrium strategy then the maximal payoff is obtained by other agents by using the same strategy. All agents can safely use the equilibrium strategy without concern that another agent can abuse the system to obtain a better result for itself at the expense of the others.

CHAPTER III

METHODS

In order to demonstrate that a multi-agent system can be used to resolve air traffic conflicts autonomously a simple air traffic simulation was created. The simulation architecture used was the Texas A&M Multi-agent Intelligent Distributed Air Traffic Simulation (MIDAS). The MIDAS architecture is reviewed more thoroughly below. The various aircraft in the simulation were piloted by TRL (Task Representation Language) agents which communicated with MIDAS via TCP socket connections.

Task plans were written for the TRL agents which enabled them to monitor the airspace around them for potential conflicts. When an impending conflict was discovered, the agents involved initiated a procedure for generating a set of possible alternate courses. These course alternatives were then exchanged and each pair of possible courses was checked for airspace conflicts. Any pair of courses that would cause a future conflict was removed from consideration. Once this final negotiation set was created the agents executed the monotonic concession protocol to select the preferred resolution. Finally, the agreed upon resolution deal was implemented and the aircraft flew the modified flight paths. The situation was then monitored to verify that conflict did not, in fact, arise.

Several problems had to be solved to allow multi-agent negotiation techniques to be applied to the air traffic conflict problem. The Monotonic Concession Protocol (described above) requires a set of options, or deals, that the agents involved will negotiate over. While some domains have very natural negotiation sets, the air traffic conflict domain does not. A method was developed by which the agents created a set of alternatives and determined which ones are viable after detecting an impending conflict.

Additionally, negotiation protocols depend on utility functions. The agents must

be able to determine which deals in the negotiation set are most desirable. Several different utility functions were created and employed by the agents in the testing process. Several safeguards were also developed and added to the protocol to ensure that the process was reasonably robust with respect to plane to plane communications failures and similar issues.

Finally, a number of simplifications were made to allow the problem to be solved. They are enumerated at the end of this chapter and considered at more length as possible areas of future research in the conclusion.

The MIDAS Architecture

For this and other aircraft simulation projects in conjunction with the Department of Aerospace Engineering we have developed a distributed, low fidelity, air traffic simulation system. When paired with autonomous agents which pilot some or all of the aircraft in the system we refer to it as the Multi-agent Intelligent Distributed Air Traffic System, or MIDAS for short.

MIDAS is an extension of a simple, generic simulation system written in Java. The base simulation system consists of a class which keeps track of elapsed time and a list of objects which have been registered for simulation. These objects within the simulation are based on a generic class with a method for performing time based updates on the simulated object. On each simulation cycle the central simulator calls the update method on each of its registered objects and informs them how much time has elapsed. The objects are then responsible for computing whatever dynamics are required to update their states appropriately.

The other primary facility provided by the central simulation class is a simple socket server. The simulator class opens a TCP socket for listening at startup time.

On each simulation cycle the server checks for incoming messages and processes them appropriately. This allows for a wide variety of client programs to connect and interact with the simulation. Two examples relevant to this work are a simple visualization program which displays the locations of the aircraft within the simulation and the agent pilots themselves, which 'drive' the aircraft via commands sent to the central server.

The MIDAS specific extensions to the simulation enable it to be used as a simple, low fidelity, air traffic simulator. A number of subclasses of the simulator object were written to handle aircraft specific data and dynamics. Additionally, these classes provide "driver" methods that allow agent processes to connect and "fly" the plane, allowing the decision making to be encoded in an intelligent agent plan while still making use of the existing simulation architecture.

TRL Agents

The TRL agent architecture was used to create the intelligent agents that carried out the negotiation and piloting processes. These agents evaluate simple task plans which are written with Lisp-like syntax. They are capable of direct communication between one another via Java remote method invocation.

The primary power of the TRL agents is in their underlying inference engine. TRL agents use the Java Automated Reasoning Engine, or JARE, to store facts in first order horn clauses and perform logical inference using back-chaining.

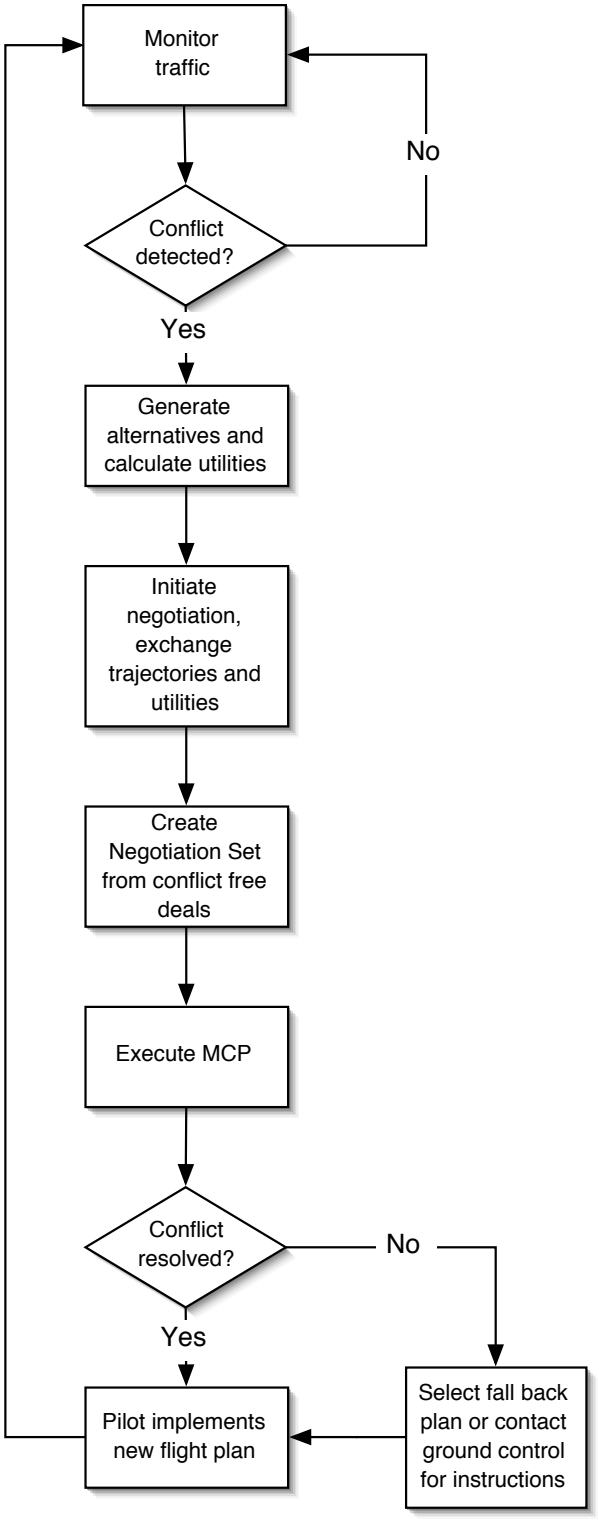

Fig. 1. Overview of the conflict resolution process

Conflict Detection and Resolution System Overview

Figure 1 provides an overview of the CD&R process described in the following sections. The process begins with agents constantly monitoring the traffic situation in their area and uses nominal state propagation to check for impending conflicts. The agents look 20 minutes into the future when predicting conflicts. If the agent determines that a conflict is going to occur, the agent initiates negotiation with the aircraft in question.

The agent initiates negotiation by sending a message to the other aircraft that an impending conflict has been detected. The agent then uses a prescribed procedure to generate a number of possible alternate trajectories. Each of these trajectories is then evaluated using the flight plan cost function of the agent to determine a cost, which is then compared to the cost of the fall back trajectory to determine a utility score.

Each alternate trajectory and its associated utility score is then sent to the aircraft with which conflict is impending. At the same time, the agent receives the other aircraft's possible trajectory alternatives and utility functions. Each of these trajectories is paired with each of the agent's own trajectories to produce 36 potential deals.

It is possible that some or all of these deals will not be conflict free. Therefore, each deal is checked, using a nominal state propagation method, to ensure that no conflict will arise if that deal is implemented. Any deal that is not conflict free is then rejected. Once this process is complete the negotiation set has been generated.

After the negotiation set is finalized the agents execute the monotonic concession protocol to select one of the deals from the negotiation set, or the conflict deal if no deal can be agreed upon.

Finally, the agents present the selected flight plan to the pilot for approval and implementation. In the simulations used for this paper the agent was also playing the role of pilot, so the selected flight plan was simply executed. In real life the pilots would have some sort of veto power and it is possible that the resulting plan would also need to be submitted to an overseeing air traffic controller for approval before implementation.

Conflict Detection

The system implemented uses a simple conflict prediction tool. Nominal state propagation is used to determine the future location of aircraft using current state information. On each agent update cycle the current position, speed, and heading of each other aircraft in the area are used to look 20 minutes into the future. If a conflict will occur during that time period a conflict resolution process will be initiated. The model defines a conflict as any overlap of the protected zones of two aircraft. The protected zone is a cylinder with a 1.5 nautical mile radius and a height of 1000 feet, centered on the aircraft. The model uses all three dimensions of state information.

During the process of state propagation and conflict detection, the times at which the impending conflict would begin and end are stored. The first time and last time of protected zone overlap are used in the process of generating the alternate trajectories that comprise the negotiation set.

Negotiation Set Generation

When the agent detects that a conflict is going to occur with another aircraft it uses a predefined process to generate six alternate trajectories. The six prescribed deviations are left, right, up, down, speed up and slow down. Each pair of opposite

trajectories is generated with a given process. In terms of the taxonomy outlined by Kuchar and Yang, this resolution process allows for climbs, turns, and speed changes, but not for combinations thereof.

The generation of the left turn, right turn pair (see Figure 2) is based upon the predicted times of the beginning and end of the conflict. The agent determines the location it would have occupied at the time the protected zones first would have overlapped. This point is used to generate two temporary waypoints. At right angles to the current heading, the aircraft projects three nautical miles to the left and to the right of the point it would have occupied at the start of the conflict. A similar procedure is used to generate two waypoints using the position the agent aircraft would have occupied at the end of the conflict. Finally, a fifth temporary point is created on the original path several minutes after the conflict would have ended. This point is the rejoin point and is used by the agent to return to the original path after the conflict evasion maneuver has been completed. These five waypoints, two to the left of the original path, the two to the right and the rejoin point, allow two new trajectories to be defined. The left alternative involves an immediate turn to fly towards the first left waypoint, to the second left waypoint, and then rejoining the original path at the rejoin point. Similarly, the right turn alternative is the path from the current position to the first right waypoint, then to the second right waypoint, and finally to the rejoin point.

The climb and descend alternatives (Figure 3) also make use of the conflict times recorded during the conflict prediction step. The climb alternative is comprised of an immediate climb to an altitude 500 feet above the current altitude. The agent continues to fly its original course at the new altitude until it passes the point at which the conflict is over and then descends back to the original altitude. The descend option is the opposite, in which the agent descends 500 feet and maintains that altitude until

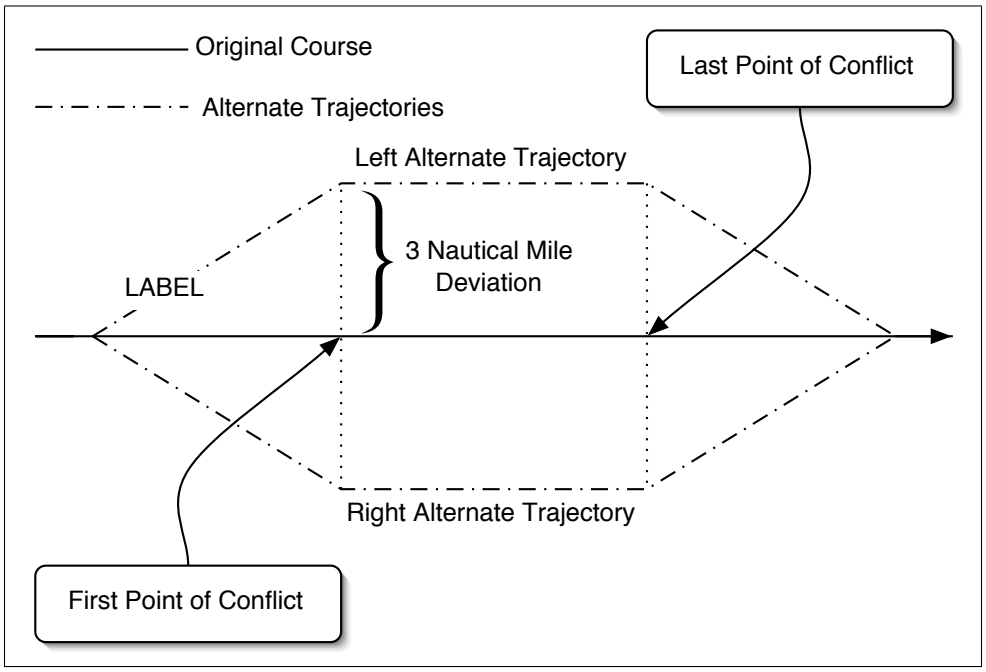

Fig. 2. Generation of left and right alternate trajectories

crossing the end of conflict point and climbing back to the original altitude.

Finally, the speed up and slow down options are generated. They are less complicated and simply require the agent to fly the previously intended path at a slightly faster or slower speeds, respectively. No other course or altitude changes are required. A value of 50 knots was arbitrarily chosen for the amount of speed up and slow down that were used in this model, but this value could easily be altered. It is also likely that in a real world application of this process this value would be dependent on the class of aircraft in question.

Once all six options have been generated, the agent processes each one using its utility function, as described below. This produces a utility score for each potential conflict resolution. The agent then signals to the other aircraft involved in the potential conflict. The message includes a request to initiate the negotiation process and then all of the flight plan data required to completely describe the six alternate tra-

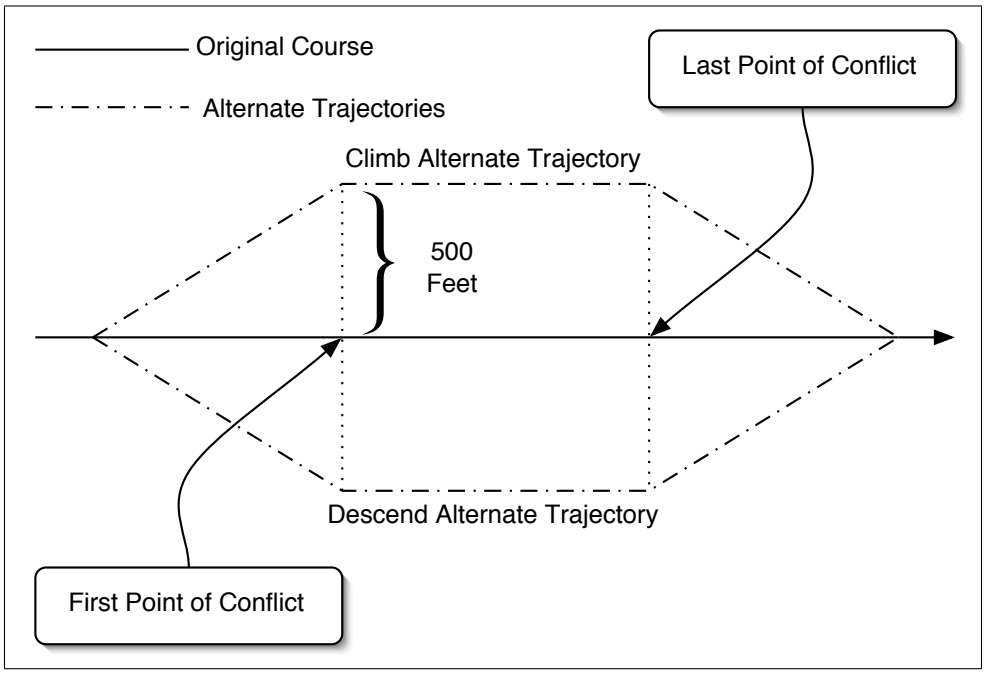

Fig. 3. Generation of climb and descend alternate trajectories

jectories and their associated utility scores. The message concludes with a statement that all the information has been sent.

Because all of the agents in the simulation are using the same process as described above, the second agent involved need not wait for the initiation of negotiation message to begin generating its course alternatives. The process described above can be thought of as occurring simultaneously in both affected agents. Similarly, the following steps that are taken by an agent after receiving the trajectory alternatives and their utilities can also be considered to be occurring simultaneously in all parties.

After receiving the trajectory alternatives, the agent pairs them up with its own generated alternatives. Each received trajectory is paired with each generated trajectory for a total of 36 potential solutions. In the terminology of the MCP, each of these 36 trajectory pairs is a deal, and together they form a preliminary negotiation set.

There is an additional constraint that must be met, however, before the process can go forward. In the deal validation step, each of the 36 potential deals must be checked to determine if it is a viable solution. In order to do this, each deal is analyzed using a process similar to that used for conflict detection. Nominal state propagation is applied to each leg of the intended trajectories to determine if the deal is truly a conflict free solution. Any deal which contains a future conflict is rejected and discarded from the negotiation set. It should be noted that while this validation occurs twice, once in each agent that is involved in the conflict, the results obtained should be symmetric.

The Conflict Deal

As outlined in the discussion of the MCP in chapter 2, a conflict deal should exist in the negotiation set. This deal serves two purposes. First, it is a baseline for the calculation of utility scores. The utility of an option for an agent is defined as the cost of the conflict deal minus the cost of the alternative being evaluated. Second, the conflict deal provides a fallback that will be used in the event that the negotiation ends in conflict or does not end at all.

In most multi-agent applications of the MCP the conflict deal is simply taken to be whatever plans the agents had been operating under prior to the initiation of the negotiation process. The air traffic domain does not allow for this possibility, however, as both agents following their initial plans will lead to a violation of the safety constraints of the system. Special considerations must be made for the conflict deal in this domain.

In this system, the conflict deal must consist of a conflict free pair of trajectories that is available to both agents at the start of the negotiation process. There are

several possibilities for the generation of this deal.

First, prior to the negotiation process the aircraft could contact air traffic control and obtain the pair of trajectories that air traffic control would have given to the aircraft in the event that they could not negotiate. This is a poor choice as it does not allow for the full automation of the process.

Second, the pilots of the aircraft in question could manually communicate at the start of the negotiation process and determine a pair of conflict free trajectories that they would be willing to fly and input this into their flight computers. The agreed upon pair could then be used as a conflict deal for the MCP process. This possibility significantly adds to the workload of the pilot, however, and is therefore not preferred.

Finally, one of the previously described conflict resolution methods that are based on worst case analysis could be used by the agents prior to the negotiation process to determine the fall back conflict deal. Each agent would first run this method to determine what course it would fly to avoid the other aircraft. The agents would then exchange these courses and they would form the conflict deal. As these courses are created using a worst case analysis they can safely be used in the event that this initial communication cannot be made and the negotiation cannot take place. This method also preserves the full automation of the process and does not require pilot intervention until the final agreed upon resolution is presented to the pilot for implementation.

This final approach is the one used in the system presented here. I have implemented the method discussed by Tomlin, Papas, and Sastry in [14] to use as the baseline worst case analysis system. The trajectory that is returned from their method is inserted into the negotiation set as the fall back conflict deal.

Utility Functions

In order to make use of the MCP, each agent must have a utility function. This function is used to encode the agent's preferences for certain flight plans. The power of the cooperative negotiation approach comes from these functions. By encoding preferences for certain flight plan attributes in a utility function, the agents are able to inject their desires into the conflict resolution process in a way that is not possible with most standard systems.

Utility functions in the MCP are built on top of cost functions. The utility of a deal is defined as the reduction in cost of that deal as compared to the conflict deal. The cost functions themselves are where the preferences are actually encoded. There has not been extensive work in the air traffic domain dealing with utility functions; some simple examples were formulated for this work. The important feature of these functions is not their precise formulation, but the fact that any preference can be encoded into them. This allows maximal flexibility for the agents to express their desires for some flight plans over others. Any conceivable function that maps flight plan data to a number can be used.

Encoding flight plan preferences into the MCP utility functions allows the MCP to select the deal that is of greatest value to the two agents. The fact that the MCP takes these agent cost functions into account is what gives it an advantage over non-cooperative methods. A system which does not incorporate agent preferences can never satisfy those preferences. The MCP is able to provide solutions which are more acceptable to all parties due to this feature.

Negotiation

Once the negotiation set has been determined, the process of selecting a deal to resolve the conflict can take place. As noted in Chapter 2, there is a short form of the MCP which requires less communication and can be completed more quickly. This is accomplished by computing the product of the utilities for each deal in the negotiation set. The deal with the highest utility product is the deal that would have been selected by the longer MCP process. In the event that there is a tie between several deals, the deal with the lowest sum of utilities is selected in order to distribute the cost as fairly as possible among the agents.

After the agent has selected the deal that would result from the negotiation process, it exchanges its selection with the other agent. If the selections are identical then the agents proceed to put the deal into action by altering their flight plans to the alternate plans agreed upon. If there is a disagreement in the final selection, which can only happen if there are multiple deals with identical utility products in the negotiation set, then the agents are deemed to be in conflict and must fall back on the conflict deal that was previously determined by the non-cooperative analysis, or the agents (pilots) could elect to appeal to ground controllers for a final deconfliction.

Experiments

The conflict detection and resolution system was evaluated using randomly generated scenarios. Each scenario consisted of two aircraft whose initial headings and speeds would cause a conflict. Figure 4 provides a visual overview for the scenario generation process.

Random scenarios were created with the following process. First, a conflict point was selected. This point is the location at which the aircraft would sustain a collision

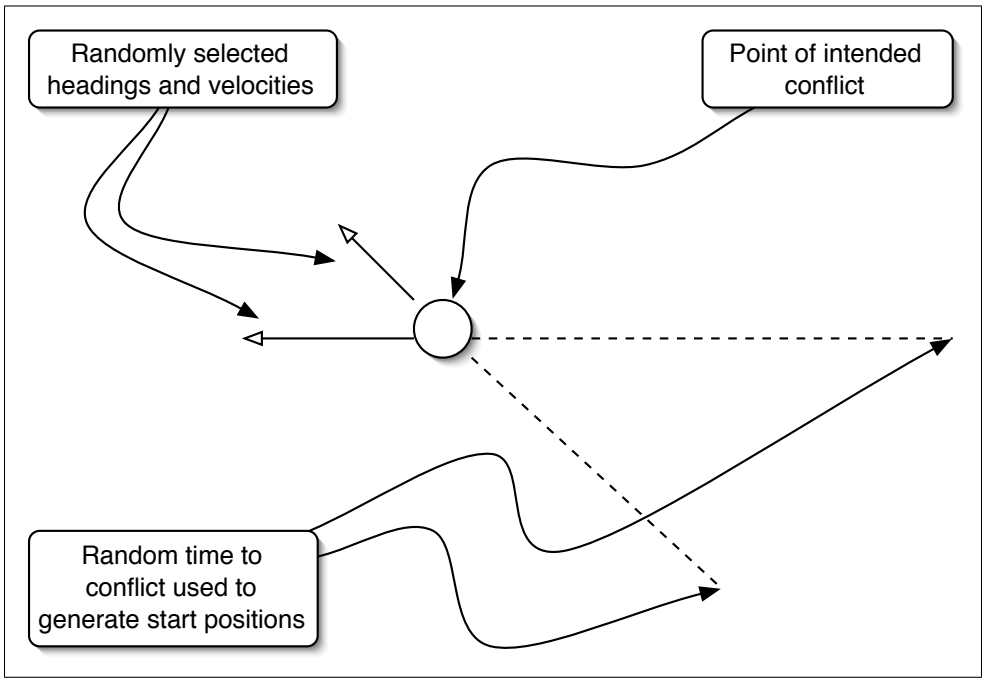

Fig. 4. Random scenario overview

without intervention. Random velocities were selected for the aircraft from a nominal range. Headings were also randomly generated for the aircraft with the restriction that they not be within 30° of one another. Finally a conflict time was selected. This number is how many minutes after the start of the scenario the conflict will occur, and was drawn from the range 10 to 20.

Using the selected headings, velocities, and time, aircraft starting positions were calculated. The flight plans were completed by creating waypoints on the aircrafts' headings one hour forward from the starting position.

CHAPTER IV

RESULTS, DISCUSSION AND CONCLUSION

Results

Several tests were run to determine the quality and viability of the system presented. As described in the previous section, a procedure was developed for generating test scenarios consisting of pairs of aircraft flying towards a conflict. This system was used to generate hundreds of test cases which were then used as initial conditions for the simulation.

The most important function performed by any conflict detection and resolution system is the maintenance of safe separation between the aircraft involved. This was qualitatively verified by running several hundred trials of randomly generated scenarios. In these trials the agents were presented with the scenario, negotiated a resolution, and the resolution was analyzed to verify that no conflict occurred. In all trials conflict was avoided. The protocol meets the minimum safety standard.

The second goal of the system was increased efficiency through cooperative conflict resolution. The efficiency of the system was evaluated using cost functions as described previously. A number of basic cost parameters were defined and combined to create cost functions for use in these tests.

$$
\begin{aligned}
D &= \text{total distance traveled} \\
\Delta A &= \text{total altitude changes during flight plan} \\
\Delta H &= \text{total heading changes during flight plan} \\
Cost &= D \qquad (4.1) \\
Cost &= D + \Delta A \qquad (4.2)
\end{aligned}
$$

$$Cost = D + \Delta H \tag{4.3}$$

$$Cost = D + \Delta A + \Delta H \tag{4.4}$$

Equation 4.1 is the simplest cost function, in which the cost of the flight plan is equal to the distance in nautical miles that is flown. Equations 4.2, 4.3, and 4.4 add penalties for altitude and heading changes. The altitude penalty is equal to the total altitude traveled during the flight plan. A plan which includes a 500 foot climb and 1000 foot descent will incur an altitude penalty of 1500. The heading change penalty is defined similarly. These penalties are intended to capture the desire to minimize pilot workload and maximize passenger comfort.

Test 1 (see Table I) shows the average results from 100 trials in which both agents use cost function 1. As expected, when both agents use the same utility function their results are very similar. Agent 1 averaged a 6.1% improvement and agent 2 averaged 9.3% improvement when comparing the negotiated deals to the conflict deal.

Table I. Summary Results for Evaluation #1

Agent	Conflict Cost	Negotiated Cost	Average Improvement	Std. Deviation
1	156.31	146.00	6.1%	5.4%
2	160.04	144.65	9.3%	5.6%

Test 2 (see Table II) shows the results of 100 trials in which agent 1 continued to use cost function 4.1, but agent 2 used cost function 4.4 which includes all three penalties. In this test agent 1 showed an average of 8.5% improvement, which is not statistically different from the result of test 1. Agent 2, however, received a much greater improvement. This demonstrates that when an agent has strong preferences for certain solutions the system allows those preferences to be expressed. This is the primary difference between this system and any non-cooperative solution. The

negotiation process allows agents to select the plan that suits them the best.

Table II. Summary Results for Evaluation #2

Agent	Conflict Cost	Negotiated Cost	Average Improvement	Std. Deviation
1	145.03	143.76	8.5%	6.4%
2	218.01	158.09	27.8%	20.5%

Overall, the tests have borne out the intuition that when agents are allowed to use cost functions to express preferences they can reach resolutions that are of higher utility than those provided by non-cooperative methods.

Discussion

The results demonstrate the primary strength of the system I have proposed in this paper. The agents are able to express preferences for certain types of flight plans through the use of cost functions. Since the cost function in use by an agent has a direct impact on the end result of the negotiation process the agent is able to achieve higher efficiency with this method. Non-cooperative methods lack this feature.

The system is also efficient with respect to the time required to produce a resolution to the conflict. This is very important due to the real-time nature and safety considerations of the domain. The primary bottleneck in the negotiation process is the communication between the agents. In particular, the initial exchange of trajectory alternatives is the primary communications load. Subsequent communications and proposals of deals need only to refer to an deal identifier. In the model proposed, six initial alternative flight plans must be sent each way. Each flight plan consists of several pieces of flight segment data including altitude, velocity and destination waypoints. Each flight plan is also sent with a utility score. In total, each flight plan requires less than 1 kilobyte of data if it is represented in a compact form. The nego-

tiation set in the system may have a maximum of 36 deals, and because at least one agent must concede on any negotiation step that is not the last step, the entire process cannot have more than 18 proposals. The total information exchanged between the two agents will depend on the representations chosen, but there is no reason it should exceed 50 kilobytes, and would likely be far less.

In practice an air to air communication system is required which can quickly transmit tens of kilobytes between aircraft. The current ADS-B system is capable of this, but there is currently a fair bit of debate as to how the system should be and will be used in the future. Additionally, there are guidelines for the frequency of ADS-B transmissions which may make it difficult to adapt to the negotiation process. Future research may well conclude that direct aircraft to aircraft data links are required or highly desirable for the higher communication load of such protocols.

The complexity of computations performed by this system, while strongly affected by the cost functions that are used, are not typically computationally intensive. Given the trend towards more and more affordable computer power, even a modestly complex utility function will be able to be computed very quickly by inexpensive hardware. This is one area where the cooperative negotiation system has a large advantage over the more complex optimization systems. This model requires a cost function to be computed fewer than ten times and a sort of the results. The more complex systems, such as that described in [5], require significantly more computing power. The system described in this paper can be run from start to finish, even assuming modest processing and communication power, in several seconds. Given that the system specifies a 20 minute look ahead for conflict detection, this is well within what would be required for safety. If speed does become a concern due to sub-optimal communication or processing capabilities then the 20 minute look ahead time can be extended, but there is a trade off between forecast time and uncertainty, and a longer

look ahead time will increase the rate of false alarms.

A number of simplifications were made to the problem of conflict detection and resolution to enable this prototype cooperative system to be developed. In order for the system to be applied in a real air traffic conflict situation these issues must be addressed.

The nominal state propagation used for conflict detection in this model is the simplest method available. The current speed, altitude and heading are used for simple linear course projections. The cases analyzed were all of such a form that they were compatible with this model as the aircraft involved were flying straight line trajectories to specific destination waypoints. The next natural extension to this is to use intended flight plan information to predict conflicts that are not on the current flight segment. This is not difficult, and in fact such flight plan state propagation is already in use in this system: after the initial pairing of trajectory alternatives, those with conflicts are rejected using this process. The more interesting issue raised by using flight plan propagation in the conflict detection step is how to handle conflicts that occur very close to an intended course change. A much more sophisticated alternate trajectory generation process would be required to handle this elegantly. The system currently in use adds an out of route deviation and then returns the aircraft to the original flight segment. In the case of a conflict occurring near an intended course change, this could result in highly inefficient flight plans, and perhaps recurring conflicts between the same two aircraft. One possible solution could be to ignore the current waypoint in the trajectory generation process if the aircraft is within some threshold distance of it. The threshold should be chosen based on the distance the aircraft would travel during the conflict resolution maneuver itself. This avoids the problem of the conflict resolution taking the aircraft past an intended waypoint and forcing it to return. Instead, the alternate trajectories would be based

on a course directly from the current position to the next waypoint after the current one in the flight plan sequence.

Another simplification made to the process is in the generation of alternate trajectories. In the system proposed, each aircraft creates a set of potential course deviations and sends them to the other party in the negotiation process. Each agent then pairs each of its own trajectories with each of the trajectories received from the other agent. This is a viable solution, but has room for improvement.

First, there is no guarantee that any of the 36 resulting trajectory pairs will resolve the conflict. Given the set of prescribed course deviations, this is unlikely, but not explicitly prevented by the process.

Second, this solution is somewhat wasteful. The trajectories that an agent pairs with those received from its negotiation partner may not be the preferred responses to the suggested course deviations.

Applying more advanced motion planning tactics to this step would resolve both issues. Instead of mindlessly pairing the generated trajectories with those that are received, a specific response to each trajectory should be generated. For each proposed course deviation the location of the other aircraft is well known and can be used to plot a conflict free response. This response could then be optimized with respect to the agent's own cost function. This will guarantee a conflict free deal and increase the utility of the deal. This method does, however, come at the price of increased complexity and increased computational requirements.

Finally, the largest simplification made by this model is that only pairwise conflicts are considered. Many conflict resolution systems currently do not handle multiple conflicts, but as noted by Kuchar and Yang, there are cases that can be constructed in which simple chaining of pairwise conflict resolutions is not sufficient.

For cooperative solutions this presents a particularly difficult problem. Increasing

the number of participants in a negotiation beyond two significantly increases the complexity. It is no longer possible to receive the proposed trajectories from one other agent and construct conflict free responses. One possibility for the deal generation would be to have the agents take turns as follows: for a three party negotiation, there would be three cycles. On the first cycle, agent A would propose trajectory alternatives and broadcast them. Agent B would then construct conflict free preferred responses and broadcast them. Finally, agent C would be able to construct responses that take both A and B's trajectories into account. Then the process would begin again with B proposing the initial set of trajectories. At the end of the process there would be a set of possible deals that have been somewhat fairly generated.

The process of a multi-party negotiation is difficult as well. It is not clear if the basic MCP process can be extended to multiple parties simultaneously proposing and subsequently conceding or holding firm. Intuitively, it seems that the process should work, if on each step one agent is allowed to hold firm and the others must concede. Whether this results in an optimal result remains to be seen.

Conclusion

As air travel continues to increase in popularity and technology continues to advance, Free Flight will become both more necessary and more practical. These trends create the need for more efficient and autonomous conflict detection and resolution processes.

In this thesis I have proposed that cooperative multi-agent negotiation techniques can be brought to bear on the CD&R problem. This is an important deviation from the existing research in the domain, which has largely avoided cooperative solutions.

I have outlined a number of the issues which must be approached in order to apply

these multi-agent negotiation techniques to this problem and presented one possible implementation. The system was shown to be viable, as it successfully prevents conflicts from occurring. Additionally, the system was demonstrated to improve the efficiency of the resolution process by allowing agents to express their preferences through cost functions which directly influence the resulting conflict free trajectory pair.

REFERENCES

[1] J. C. Harsanyi, "Approaches to the bargaining problem before and after the theory of games: a critical discussion of Zeuthen's, Hick's and Nash's theories.," *Econometrica*, vol. 24, pp. 144–157, 1956.

[2] J. Kosecka, C. Tomlin, G. Pappas, and S. Sastry, "Generation of conflict resolution maneuvers for air traffic management," in *Proc Int. Conf. Intelligent Robots and Systems*, (Grenoble, France), pp. 1598–1603, 1997.

[3] S. Kraus, *Strategic Negotiation in Multiagent Environments*, 1st ed. Cambridge, Massachusetts: The MIT Press, 2001.

[4] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, pp. 179–189, December 2000.

[5] P. Menon, G. Sweriduk, and B. Sridhar, "Optimal strategies for free-flight air traffic conflict resolution," *Journal of Guidance, Control, and Dynamics*, vol. 22, pp. 202–211, March-April 1999.

[6] J. F. Nash, "The bargaining problem," *Econometrica*, vol. 18, pp. 155–162, 1950.

[7] J. F. Nash, "Two-person cooperative games," *Econometrica*, vol. 21, pp. 128–140, 1953.

[8] Radio Technical Commission for Aeronautics, "Final Report of RTCA Task Force 3, Free Flight Implementation, RTCA/DO-242," tech. rep., Washington, D.C., October 1995.

[9] J. Rong, S. Bokadia, S. Shandy, and J. Valasek, "Hierarchical agent based system for general aviation CD&R under free flight," in *Proc. AIAA Guidance, Navigation, and Control Conference*, (Monterey, California), 5-8 August 2002.

[10] J. Rong, Y. Ding, J. Valasek, and J. Painter, "Intelligent system design with fixed-base simulation validation for general aviation," in *Proc. IEEE International Symposium on Intelligent Control*, (Houston, Texas), 5-8 October 2003.

[11] J. S. Rosenschein and G. Zlotkin, *Rules of Encounter*, 2nd ed. Cambridge, Massachusetts: The MIT Press, 1994.

[12] S. Shandy and J. Valasek, "Intelligent agent for aircracft collision avoidance," in *Proc. AIAA Guidance, Navigation, and Control Conference*, (Montreal, Canada), 6-9 August 2001.

[13] C. Tomlin, G. Pappas, J. Lygeros, D. Godbole, and S. Sastry, "A next generation architecture for air traffic management systems," Technical Report UCB/ERL M97/7, University of California, Berkeley, 1997.

[14] C. Tomlin, G. Pappas, and S. Sastry, "Noncooperative conflict resolution," in *Proc. IEEE Int. Conf. on Decision and Control*, (San Diego, California), December 1997.

[15] C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution for air traffic management : A study in muti-agent hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 509–521, 1998.

[16] J. P. Wangermann and R. F. Stengel, "Optimization and coordination of multiagent systems using principled negotiation," *Journal of Guidance, Control, and Dynamics*, vol. 22, pp. 43–50, January-February 1999.

[17] G. Zlotkin and J. S. Rosenschein, "Negotiation and task sharing among autonomous agents in cooperative domains," *IJCAI*, pp. 912–917, 1989.

[18] G. Zlotkin and J. S. Rosenschein, "Mechanizms for automated negotiation in state oriented domains," *Journal of Artificial Intelligence Research 5*, pp. 163–238, 1996.

VITA

Steve Wollkind

104 Marlboro St.

Quincy, MA

02170

Steve Wollkind was born in Norwalk, CT on February 20th, 1979. He grew up in Connecticut and attended Williams College in Williamstown, MA, graduating with a B.A. in Computer Science in 2001.