AIAA Guidance, Navigation, and Control Conference and Exhibit
21 - 24 August 2006, Keystone, Colorado
AIAA 2006-6060

Pattern-Based Genetic Algorithm for Airborne Conflict Resolution

Robert A. Vivona,* David A. Karr,† David A. Roscoe‡
L-3 Communications Titan Group, Billerica, MA 01821

NASA has developed the Autonomous Operations Planner (AOP) airborne decision support tool to explore advanced air traffic control concepts that include delegating separation authority to aircraft. A key element of the AOP is its strategic conflict resolution (CR) algorithm, which must resolve conflicts while maintaining conformance with traffic flow management constraints. While a previous CR algorithm, which focused on broader flight plan optimization objectives as a part of conflict resolution, had successfully been developed, new research has identified the need for resolution routes the users find more acceptable (i.e., simpler and more intuitive). A new CR algorithm is presented that uses a combination of pattern-based maneuvers and a genetic algorithm to achieve these new objectives. Several lateral and vertical maneuver patterns are defined and the application of the genetic algorithm explained. A new approach to defining a conflicted fitness function using estimates of the local conflict region around a conflicted trajectory is also presented. Preliminary performance characteristics of the implemented algorithm are provided.

Introduction

Since the release of the RTCA Free Flight concept[1] in the mid 1990's, the air traffic management (ATM) research community has seen a renewed increase in the study of advanced ATM operational concepts that reduce the airspace system's reliance on centralized, ground-based air traffic management. The crux of these concepts is the distribution, at some level, of separation authority to the cockpit crew under instrument flight rules (IFR). Though distributed separation authority under IFR has been studied since the mid 1960's,[2] even recent advances in navigation and surveillance technology has only enabled very limited cockpit-based concepts to progress all the way to field demonstrations.[3]

As part of the National Aeronautics and Space Administration's (NASA) Distributed Air/Ground Traffic Management (DAG-TM) concept,[4] NASA's Langley Research Center has been researching the extent to which separation authority can be delegated to the cockpit. In the NASA concept, properly equipped aircraft can achieve full autonomous flight, self-optimizing their full 4D path with a minimum of external constraints. Unlike concepts which assume autonomous flight is only allowed in specifically segregated airspace, the NASA concept assumes that autonomous aircraft fly in a mixed-equipage environment, where equipped aircraft regularly interact with unequipped aircraft managed by controllers on the ground. Though ground-based traffic flow management (TFM) constraints are possible in this environment and conformance to these constraints by autonomous aircraft is required, the NASA concept includes cockpit-based airspace complexity monitoring to support autonomous aircraft in selecting paths that avoid potentially complex airspace.

The cornerstone of NASA Langley's research into autonomous flight is the development of a prototype cockpit-based decision support tool (DST) called the Autonomous Operations Planner (AOP).[5] The AOP's role in the cockpit is twofold: (1) to support the flight crew in performing the new duties required for assumption of separation authority in the cockpit; and (2) to support enhanced optimization of the aircraft's flight path given the flexibility

---

* Research Engineer, Associate Fellow AIAA
† Senior ATC Software Developer, Member AIAA
‡ ATC Software Developer

enabled by autonomous flight. To successfully fulfill both of these roles, the AOP seamlessly integrates with other airborne avionics in charge of navigation, surveillance, and guidance, including the Flight Management System (FMS), the Mode Control Panel (MCP), the Aircraft Dependent Surveillance – Broadcast (ADS-B) transceiver, and Flight Information System (FIS) and Traffic Information System (TIS) receivers. AOP outputs are integrated into existing cockpit displays, including the Primary Flight Display (PFD), the Navigation Display (ND), and specialized AOP pages on the Multi-purpose Control and Display Unit (MCDU).

In its role as the enabler of delegated separation authority, the AOP is responsible for providing all information in support of airborne conflict management (ACM). Specifically, the AOP provides a variety of integrated airborne conflict detection and resolution (CD&R) capabilities. For support of strategic planning (5-20+ minute look-ahead decision making), the AOP provides strategic, intent-based CD&R advisories for evaluation and ultimate acceptance/rejection by the flight crew. Based on MCP and FMS data, the AOP automatically displays predicted separation loss locations on both the ND and the MCDU AOP pages. If the aircraft is in an MCP-based guidance mode (e.g., altitude hold and heading select), the AOP automatically displays MCP target resolution advisories (e.g., altitude, vertical speed, and/or heading "bugs") on the PFD and ND when separation loss is predicted. If the aircraft is in fully engaged, FMS-based Lateral Navigation (LNAV) and Vertical Navigation (VNAV) guidance mode, the AOP (upon request) displays a full 4D resolution path on the ND that can be uploaded directly into the FMS for execution. The AOP also supports tactical maneuvering (3-5 minute look-ahead decision making) by providing state-based CD&R information and ultimately suppresses its CD&R output if Traffic Alert and Collision Avoidance System (TCAS) resolution advisories become available.

The focus of this paper is on the AOP's strategic, intent-based conflict resolution (CR) algorithm which provides full 4D resolution paths for upload into the FMS. The discussion starts with a description of the AOP's requirements for strategic CR. Then, a discussion of the initial CR approach, a genetic algorithm implemented to provide both separation assurance and flight path optimization, is presented highlighting both the successes and challenges discovered during this development. This is followed by a description of the new CR algorithm, the majority of the discussion, which is based on defined maneuver patterns designed to address the identified challenges of the first algorithm while capitalizing on its successes. Details of applying an enhanced genetic algorithm as part of the CR algorithm are then presented. Finally, preliminary performance results of the implemented algorithm are provided.

AOP Conflict Resolution Requirements

To ensure effective support of ACM, a series of functional and performance requirements were developed for the AOP strategic conflict resolution capability.[6] The reference gives an extended list of all requirements related to strategic CR, but the discussion below focuses on just those (six) requirements that directly impact the design of the CR algorithm. The other requirements, such as resolution degree-of-freedom selection or the use of modifiable flight rules for selecting which aircraft within a conflict should maneuver, are not impacted by changes in the CR algorithm.

The first CR requirement is that an acceptable resolution path must meet all required flight plan constraints. The AOP currently identifies these constraints from the active route information it receives from the FMS. The assumption is that any constraint that exists in the FMS's current active route has been externally imposed on the aircraft and must be maintained. For example, altitude and speed constraints typically assigned at waypoints on the boundary between en route and terminal airspace, if applied, must be met even if the aircraft's path up to this transition is completely unconstrained. In addition, required time of arrival (RTA) constraints applied to waypoints must also be met. This is the currently assumed form of externally defined en route TFM constraints imposed on the aircraft. This overall requirement is not assumed to be limited to just the types of constraints available in current day FMS. Since the AOP integrates with a Langley developed research FMS that can be modified beyond present day capabilities, new types of FMS constraints can be added. Also, AOP receives information from many other systems onboard the aircraft, so additional constraints from other sources could also be added.

The second CR requirement is that an acceptable resolution path must be "flyable" by the FMS. In other words, an uploaded resolution flight path must be acceptable to, and executable by, the FMS. Additionally, any route discontinuities or errors (e.g., a bad turn radius) that would result within the FMS after upload should be identified during the CR process and those candidate routes eliminated from consideration. Since these strategic resolutions will all be implemented via LNAV/VNAV flight guidance, predictable FMS execution of the resolution path is required to ensure minimum separation requirements are maintained.

The third CR requirement is that an acceptable resolution path must be conflict free with respect to all hazards within the airspace over a given look ahead time. This means each aircraft performing a conflict resolution

maneuver is responsible for completely de-conflicting itself with respect to all other hazards. For the DAG-TM concept, hazards include other traffic aircraft as well as area hazards representing active special use airspace (SUA), convective weather cells, and terrain.

The fourth CR requirement is that conflicts within the next 20 minutes must be resolved. Because this time horizon is beyond the decision making horizon for typical tactical maneuvering, the CR algorithm must integrate with the strategic planning decisions of the flight crew. For example, the CR algorithm must support descent management during the transition from en route to terminal airspace, including the handling of any RTA constraints for TFM.

The fifth CR requirement is that the resolution must work in all phases of flight. This requires the CR algorithm to provide effective resolution paths during significant climb outs (transition from terminal to en route airspace) and descents (transition from en route to terminal airspace). The CR algorithm must also provide resolution paths consistent with the types of maneuvers performed in the different phases of flight. For example, step altitude changes are typical vertical maneuvers in the cruise phase of flight, where intermediate altitude level offs are more typical during departure/arrival phases of flight.

The sixth CR requirement is that the CR capability must be able to return a viable resolution path "quickly," even if it continues to search for an improved (optimal) solution. Providing a conflict free path is significantly more important during ACM than providing an optimal path with respect to path distance, fuel burn etc. This requirement implies that, if necessary, optimization should be sacrificed to ensure convergence to a conflict free solution.

Previous AOP Strategic Conflict Resolution Development

Reference 6 describes the initial algorithm developed for the AOP's strategic CR capability as well as the logic behind its design. One focus of this algorithm's design was to simultaneously support both AOP cockpit roles; specifically delegated airborne separation authority and enhanced optimization of the aircraft's flight plan. Upon identification of a predicted conflict, the CR algorithm used a very flexible approach for perturbing the FMS's current active route to achieve both objectives. Perturbations in several dimensions were sequentially performed with each possible perturbation checked against a predefined probability threshold to determine if the perturbation should occur. For example, in the cruise phase of flight, the sequence started with perturbations of global route parameters (e.g., cruise speed or cruise altitude); then lateral waypoints were either inserted or removed at random. This new set of waypoints was then traversed and random "short cuts" (i.e., a series of waypoints removed in sequence) added. Finally, random displacements (along-track and/or off-track) of the final set of lateral waypoints were performed. These perturbations were only constrained by aircraft performance, procedural flight rules or user input constraints. Final perturbations were not constrained by the original active route path except that altitude, speed and time constraints within the original active route had to be maintained.

A genetic algorithm was used to control the perturbations to find an "optimal" flight plan that met the constraints and was conflict free. A genetic algorithm was selected for its ability to solve complex conflict geometries while meeting spatial and temporal constraints, as well as for its ability to simultaneously optimize the flight plan based on broader flight planning objectives. The genetic algorithm's fitness function was responsible for both the convergence to a conflict free solution and optimization of the flight path. Fitness function flight plan optimization options included minimum fuel, minimum time, or minimum cost based upon a specified cost index. Initial testing of the implemented strategic CR algorithm showed positive results with respect to both resolution and optimization.

In 2004, NASA's Langley and Ames Research Centers performed a joint simulation experiment[7-10] to investigate the feasibility of elements of the DAG-TM concept. In the time leading up to this experiment, research led to the emergence of new requirements for the AOP strategic conflict resolution in support of the experiment. These requirements increasingly focused on improving the "user acceptability" of the resolution route maneuvers and focused less on flight plan optimization. A shift in the general perception of the resolution routes occurred. Instead of these routes being perceived as new, re-optimized paths where optimization was initiated because of a detected conflict, they were viewed as resolution maneuvers added to the previous active route. In this new context, some of the CR algorithm's flexibility in route perturbation became undesired. For example, an optimal resolution route that crisscrossed the old active route was deemed an undesirable maneuver. Since the CR algorithm was not designed to constrain its solutions with respect to the original active route, complex perturbation restrictions had to be added to make the resolution routes conform to these new requirements. What was worse, these new maneuver-based constraints were often not easy to define within the context of the highly flexible perturbations. For example, to stop the crisscrossing of the active route, coupling between early and late route perturbations had to be added. In other

words, the effect of deleting waypoints, displacing waypoints (along- or cross-track), and adding short cuts all had to be synchronized to ensure a crisscross did not occur.

The addition of these new maneuver constraints ultimately led to several negative impacts. The first was a decrease in the algorithm's ability to find a conflict free path. This was believed to result from the difficulty in casting the maneuver constraints into perturbation restrictions, which led to the various restrictions interacting in unexpected and undesired ways. Second, the actual resolution maneuver (i.e., the change to the active route, not the entire active route) was difficult to optimize. Typically, the objective was to minimize the maneuver in some way, but additional objectives included completing maneuvers as early in the route as possible. Since the geometry of the resolution maneuver had to be extracted from the final route (the perturbations themselves were general and did not define the geometry of any specific resolution maneuver), the ability to identify the relevant parameters of the maneuver geometry (e.g., offset distance from original path) for minimization was computationally costly. The final negative impact was that the software code logic became very complex and difficult to maintain/modify.

Though the development of the original CR algorithm was successful in supporting the joint simulation, it became apparent that a new CR algorithmic approach needed to be explored to support the new direction in requirements. This new approach needed to be built upon the successes of the original algorithm while addressing the increased desire for maneuver-based resolution routes and avoiding the negative impacts experienced by constraining the original algorithm.

AOP's New Strategic Conflict Resolution Approach

A. Overview

The new strategic CR approach uses a re-designed genetic algorithm to select and optimize a resolution maneuver from a set of pre-defined maneuver patterns. Each maneuver pattern is designed to execute a different type of user-accepted path modification. For example, one pattern may add a lateral offset to the original active route where another may create a step altitude change in the current cruise altitude. The creation of a resolution route using a pattern is then just a matter of "positioning" the pattern (i.e., where along the active route to start the maneuver) and "sizing" the pattern (i.e., defining appropriate values for the pattern's geometric parameters) to avoid the conflict. Because only one pattern is applied at a time,[§] each pattern's geometry can be designed without regard to the geometric requirements of the other patterns. As opposed to the application of constraints to control resolution path maneuvers (as was required in the original CR algorithm), the pre-defined pattern approach eliminates the possibility of geometric requirements for different patterns "interacting in unexpected and undesired ways."

The current set of AOP lateral and vertical maneuver patterns (described in more detail later) are:

- Lateral Offset Pattern (Lateral) – the aircraft laterally parallels the original active route along one leg
- Direct Intercept Path Pattern (Lateral) – the aircraft leaves the original active route on a direct lateral path to reconnect to a downstream leg
- Path Stretch Pattern (Lateral) – the aircraft leaves the original active route on a "turn out" course, follows this course for some distance before turning back to a direct capture of a downstream waypoint
- Waypoint Migration Pattern (Lateral) – one or more active route waypoints are displaced to create a new lateral path with the same number of active route legs
- Cruise Step Climb/Descent Pattern (Vertical) – the aircraft's cruise altitude is temporarily increased/decreased at some location

To select the appropriate maneuver pattern for a given situation, the CR algorithm must determine which patterns are viable (and desired) for different conflict scenarios. For example, a pattern that requires an intercept of a later active route leg would not work in resolving a conflict on an active route comprised of a single, great circle path. To support maneuver pattern selection, an independent set of viability constraints is defined for each pattern. These constraints are checked against the current active route, the conflict's first/last loss locations, and any other relevant information to determine if a maneuver pattern can be used by the CR algorithm for the particular conflict scenario. Viability constraints are typically based on either geometric considerations (e.g., whether an active route

[§] The simultaneous application of multiple patterns is accomplished by designing a new "combined" pattern.

leg is long enough to perform at least a minimum offset maneuver) or procedural limitations (e.g., the aircraft is not allowed to "shortcut" the active route while bypassing an altitude or speed constrained waypoint), but in practice can be based on any relevant criteria. As necessary, new viability constraints can be added to control the applications of a given resolution maneuver.

Like the definition of a maneuver pattern's geometry, the definition of the pattern's viability constraints is independent of the definitions for other patterns. For example, the Cruise Step Climb/Descent Pattern temporarily changes the aircraft's cruise altitude to avoid a conflict. For this pattern to be a viable resolution maneuver, the following viability constraints must be met by the conflict scenario:

- the aircraft's active route must be have a cruise segment (i.e., the aircraft must be in or transitioning to a cruise phase of flight)
- the conflict's first and last loss locations must be on this cruise segment
- the cruise segment must be of sufficient length to allow the aircraft to return to the original cruise altitude[¶]

For a vertical resolution maneuver that impacts the aircraft's descent phase of flight (e.g., adding an at-or-above constraint during the descent[11]), the viability constraints would obviously be very different.

To support the genetic algorithm's convergence to an optimum resolution maneuver, each maneuver pattern provides its own unique definition of an optimal maneuver. A maneuver pattern could select a global characteristic to optimize (e.g., minimum path, minimum fuel, etc.), but criteria are typically chosen to optimize some aspect of the maneuver's geometry. For example, lateral offsets can minimize the lateral offset distance from the original route, the distance traveled while on the offset path, or some combination of the two. Since the geometry of the maneuver is explicitly defined by the pattern and this definition is in dimensions directly understood by the end users (e.g., distance to laterally offset the path), optimization criteria based on user preferred maneuvers is easily achieved. This approach directly addresses researcher requirements for more "tailored" maneuvers based on user feedback. It is anticipated that the definition of each pattern's optimal maneuver will be a key focus area in future research.

Since the definition of each maneuver pattern's geometry, viability constraints, and optimization approach are all unique to the pattern, new patterns can easily be added, as desired, without fear of "breaking" existing patterns.

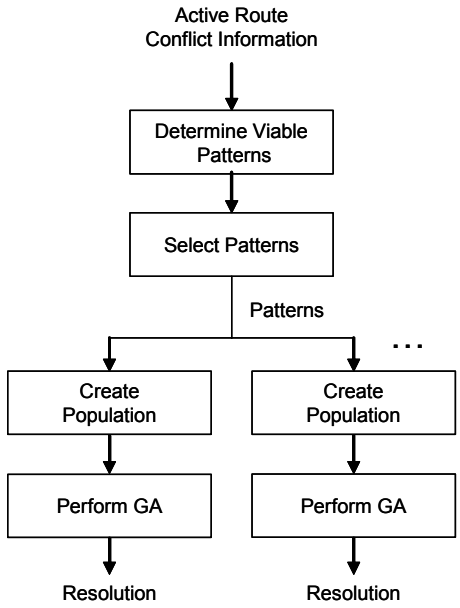

Figure 1. Top-level CR algorithm approach.

B. Top-Level Approach

The CR algorithm begins by determining which pre-defined maneuver patterns are viable resolution maneuvers (Figure 1). Each pattern's viability constraints are checked against the active route, its conflict information, and the user-selected maneuver degrees of freedom (i.e., whether lateral maneuvers, vertical maneuvers, or both are allowed). All patterns that pass the viability check are available to the algorithm for resolution. A benefit of this approach is the ability to control when/how patterns are used.[#] For example, if only a Path Stretch pattern is allowed

[¶] If the pattern did not require a return to the original cruise altitude, then this final constraint would be removed.

[#] An additional benefit is that by removing ineffective or undesired patterns, computation cycles can be applied more efficiently to improve finding successful and/or optimal resolutions.

when an RTA constraint is present (an overly restrictive scenario for illustrative purposes) and this pattern is not allowed otherwise, then this requirement could be built into the viability constraints for all of the maneuver patterns. In this way, if an RTA constraint is present, then the final resolution is guaranteed to be a path stretch maneuver (since all of the other patterns are not viable and hence, not available to the algorithm for resolution). Conversely, if the RTA constraint is not present, then the final resolution is guaranteed to be one of the other patterns (since the Path Stretch pattern is not viable). This use of mutually exclusive viability constraints is a powerful method for enabling "mode switching" between patterns within the CR algorithm. Another (more realistic) application of this technique creates automatic switching between the application of cruise altitude changes (as in the Cruise Step Climb/Descent pattern) and descent altitude restrictions (e.g., adding at-or-above constraints) alluded to earlier. At a specified distance from the aircraft's FMS top-of-descent (TOD), the CR algorithm could automatically "turn off" the cruise altitude change pattern (through its viability constraints) and "turn on" a descent altitude restriction pattern without any input from the flight crew. This effect happens automatically since the CR algorithm performs new viability checks every time the CR algorithm is run. This approach can be extended further to incorporate dynamic user constraints as viability constraints as well. For example, the user could decide that lateral offset maneuvers are no longer desired. By adding a "user not allowed" constraint to the viability constraints, this specific pattern could be removed from consideration through an input from the MCDU. This has the potential to dramatically increase the user acceptability of the final resolution maneuver.

From the set of viable maneuver patterns, the CR algorithm then creates one or more parallel populations for running the genetic algorithm. The use of populations will be described in more detail during the genetic algorithm discussion, but at this point each population can be thought of as a collection of maneuver patterns that will compete against each other to produce a single resolution maneuver option. The current AOP concept for strategic CR is to provide a single lateral and a single vertical resolution maneuver for display to the flight crew. The CR algorithm supports this requirement by creating two parallel populations: one for the viable lateral patterns and one for the viable vertical patterns. By placing all of the viable patterns of the same type (lateral or vertical) in the same population, these patterns are forced to compete with each other with only the most optimum resolution returned. For example, if both the Lateral Offset and Direct Intercept Path patterns were viable, then they would compete (in the Lateral population) until only the most optimum maneuver (either a lateral offset or a direct intercept path) was returned. This competition is performed by applying the genetic algorithm to the population.

The use of a genetic algorithm (GA) was maintained from the original CR algorithm approach (though the new GA has been redesigned for better application to the pattern-based approach, as described later) because of GA's strength in finding solutions to problems with characteristics similar to those for conflict resolution. Specifically, three aspects of genetic algorithms make them very useful for this application: [12]

- they can optimize with continuous or discrete variables
- they don't require derivative information
- they can optimize variables with extremely complex cost surfaces

As will be seen in the detailed descriptions below, the new patterns are often a mixture of discrete parameters[**] (e.g., from which leg to leave the active route) and continuous parameters (e.g., lateral offset distance), which does not cause a problem for GA. Also, when determining a "derivative" for use in de-confliction, many heuristic approaches fall back on simplifications (e.g., a constant groundspeed assumption over the entire aircraft path or considering only one traffic hazard at a time) to deal with the temporally and spatially varying nature of conflicts. The fact that GA does not require derivatives avoids making these simplifications. As will be seen in the discussion about the new conflicted fitness function below, the fitness function uses an estimation of the conflict variation around the aircraft's trajectory. This makes the "cost surface" variable in space and time (i.e., the same spatial location can have a different cost for a different arrival time), which causes the same issues as in trying to determine derivative information. Genetic algorithms are less sensitive (i.e., rate of convergence is less negatively impacted) to this situation than other algorithmic approaches.

[**] It should be noted that many discrete parameters could potentially be re-cast as "equivalent" continuous parameters. Since one of the goals of the new approach was to avoid the previously experienced pitfalls of re-casting constraints (e.g., added expense in calculating fitness values), this practice was generally avoided.

C. The Current Patterns

Each maneuver pattern represents a different "user desired" deviation from the original active route path and is defined by the geometry of that desired deviation. Figures 2a-d show the designs for the four current lateral maneuver patterns and Figure 3 shows the design for the current vertical maneuver pattern. Most of these patterns were developed to represent a variety of current controller (and ultimately pilot) procedures in dealing with conflicts (e.g., route intercepts, waypoint captures, lateral offsets, path stretches, cruise altitude changes), with and without the presence of time-based TFM constraints. One (the Waypoint Migration pattern) was designed to represent a more flexible route planning approach to conflict resolution that could be enabled (for either airborne or ground-based with datalink applications) in the near future with additional technology. The goal was to illustrate a variety of maneuvers that are possible with the pattern-based approach. Additional lateral, vertical, speed, etc. maneuver patterns can easily be generated, based on researcher or user input.

Each maneuver is defined by a series of parameters. Each parameter represents a degree-of-freedom that the genetic algorithm can vary to attempt to resolve all conflicts and meet all constraints. During the design of a maneuver, some of the pattern parameters are given predefined minimum or maximum limits, where others are limited by the active route geometry. Predefined parameter limits are typically chosen based on either procedural restrictions (e.g., maximum offset path length acceptable to a pilot) or to eliminate ineffective solutions during the genetic algorithm (e.g., minimum direct intercept path turnout angle to ensure a significant path deviation).

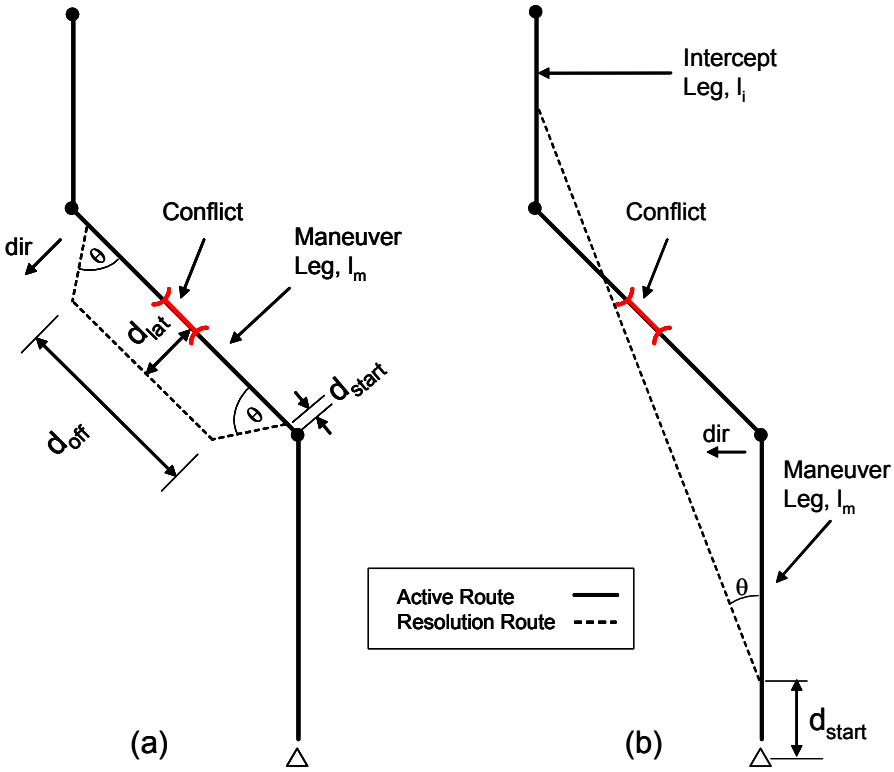

Figure 2a-b. Lateral Offset (a) and Direct Intercept Path (b) lateral maneuver patterns.

1. The Lateral Offset Pattern

The use of an offset maneuver (Figure 2a) to parallel the current active route is typical in today's environment to avoid a local problem (e.g., bad weather or a potential conflict) and is a navigation feature available in many FMS.

This pattern has been simplified to allow a parallel path to only one active route leg,[††] but allowing the parallel path to continue through one or more turns is a simple extension.

The maneuver is defined by five parameters:[‡‡]

- maneuver leg (l_m) – a discrete parameter (from the set of all active route legs) which defines the active route leg to parallel
- offset direction (dir) – a discrete parameter defining on which side to parallel the active route
- start distance (d_{start}) – a continuous parameter defining the distance along the maneuver leg where the aircraft leaves the original active route path
- offset distance (d_{lat}) – a continuous parameter defining how far from the original active route to parallel the path
- offset length (d_{off}) – a continuous parameter defining the distance to remain on the parallel path

By using the maneuver leg as a parameter, the CR algorithm is able to identify offset maneuvers on future legs as well as on the current leg. Where to begin the maneuver, how far to offset and how long to remain on the offset are all critical parameters for both resolution and path optimization. A minimum lateral offset maneuver is currently defined as one that minimizes a combination of offset length and offset distance.

2. The Direct Intercept Path Pattern

This maneuver (Figure 2b) enables the aircraft to leave the active route path from one leg and to intercept the active route somewhere along the length of a later leg (i.e., a route intercept maneuver). This type of maneuver shortens the path flown, a common technique used by controllers and often desired by pilots when a conflict must be avoided.

The maneuver is defined by five parameters:

- maneuver leg (l_m) – a discrete parameter (from the set of all active route legs) which defines the leg from which to leave the active route path
- intercept leg (l_i) – a discrete parameter (from a subset of the active route legs) which defines the leg to reconnect to the active route
- turnout direction (dir) – a discrete parameter that defines to which side of the active route path the aircraft leaves
- start distance (d_{start}) – a continuous parameter defining the distance along the maneuver leg where the aircraft leaves the original active route path
- turnout angle (θ) – a continuous parameter that defines the angle at which the aircraft leaves the maneuver leg

A minimum intercept maneuver is currently defined as one which skips the fewest number of intermediate legs (intercepting the next leg is preferred; other legs are used if necessary), but also intercepts as close to the "downstream" end of the intercept leg as possible. The concept is that this is a "local" intercept maneuver (hence the preference to intercept "closer in" legs), but that if you are going to perform a short cut, you might as well short cut the route as much as possible. This type of specific, user-preferred "procedural" definition for a single maneuver's optimization criteria shows an additional strength of the pattern-based approach. If it were desired to change this pattern's optimization criteria to a more global one (e.g., minimum overall path length), this would simply require changing this pattern's fitness criteria. This change would have no impact on the optimization criteria of the other patterns.

[††] The actual pattern allows connecting to the next active route leg past the maneuver leg without first returning to the maneuver leg, but this geometry is not illustrated.

[‡‡] Note that the angle at which the aircraft leaves and returns to the active route could be an additional parameter, but is assumed constant in the current definition of this pattern.

3. The Path Stretch Pattern

Path stretch maneuvers (Figure 2c) are a common current-day delay absorption technique used by controllers to meet TFM constraints, especially the time-based TFM constraints generated by ground-based metering systems. A path stretch is performed by vectoring the aircraft off of its flight plan route (in the airborne case, the active route) and maintaining the new course until the required amount of delay has been absorbed. The aircraft is then turned back on a direct course to capture a downstream waypoint (typically the waypoint with the time-based TFM constraint). Path stretch maneuvers are also an effective conflict resolution technique, with or without the presence of a time-based TFM constraint.

The path stretch maneuver is defined by six parameters:

- maneuver leg (l_m) – a discrete parameter (from the set of all active route legs) which defines the leg from which to leave the active route path
- capture waypoint (*cp*) – a discrete parameter (from a subset of the active route waypoints) which defines the waypoint at which to recapture the active route after completing the path stretch
- turnout direction (*dir*) – a discrete parameter that defines to which side of the active route path the aircraft leaves
- start distance (d_{start}) – a continuous parameter defining the distance along the maneuver leg where the aircraft leaves the original active route path
- turnout angle (θ) – a continuous parameter that defines the angle at which the aircraft leaves the maneuver leg
- stretch distance ($d_{stretch}$) – a continuous parameter that defines how long the aircraft stays on the vector before turning to capture the capture waypoint

This pattern is a very flexible version of a path stretch maneuver. It can be initiated on any active route leg in front of the aircraft (including the leg the aircraft is currently on) and can return to any waypoint past (and including) the maneuver leg end waypoint. If an RTA is present, the pattern only changes the active route's lateral

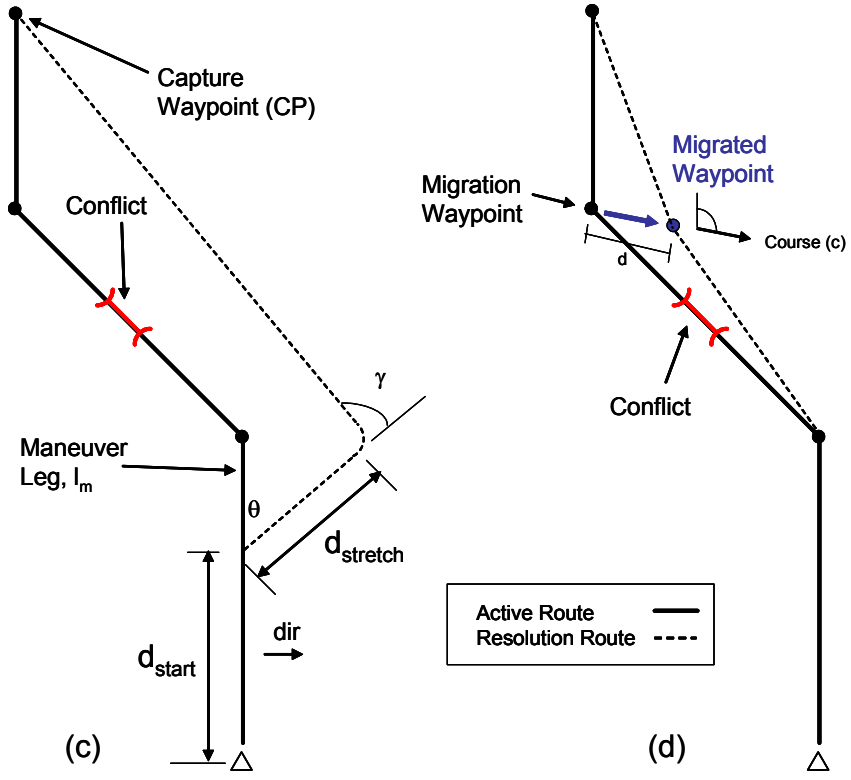

Figure 2c-d. Path Stretch (c) and Waypoint Migration (d) lateral maneuver patterns.

path, leaving the FMS's RTA capability free to vary the speed as necessary to meet the RTA. This enables the pattern to avoid conflicts while still achieving the required time constraint.

The definition of a minimum path stretch maneuver is currently still being researched. One of the ideas being investigated is to minimize the stretch distance while maximizing the turnout angle. This will tend to move the "turn out" location towards the capture point, while pushing the "turn back" location away from the capture point. This addresses a concern voiced by pilots during development for the Ames/Langley Joint Simulation, where late turn back locations were undesired. As with all of the patterns, the ability to define an "optimal" path stretch can be studied and implemented without impacting the optimal definitions for the other patterns.

4. The Waypoint Migration Pattern

This pattern (Figure 2d) uses minor deviations in the position of one or more of the active route's existing waypoints to avoid a conflict. Since the FMS is not limited to only published waypoints and the new migrated waypoints can be directly uploaded into the FMS, minor changes in the active route path can easily be accommodated. This pattern does not have a current day equivalent in the context of conflict resolution, but it is expected to be very effective if only small changes in the active route's lateral path are required for de-confliction.

The waypoint migration pattern is defined by four parameters:[§§]

- number of migration points – an integer parameter defining the number of waypoints to migrate
- migration points – a set of discrete sub-parameters defining the waypoints to migrate
- migration course set (c) – a set of continuous sub-parameters defining the course (in degrees) in which to migrate each waypoint
- migration distance set (d) – a set of continuous sub-parameters defining the distance in which to migrate each waypoint

Since the number of waypoints to migrate (as well as the direction and distance of each of these waypoints) is variable, the pattern is very flexible. A minimum waypoint migration is one which minimizes the total number of waypoints migrated and their total distance migrated.

5. The Cruise Step Climb/Descent Pattern

This pattern (Figure 3) uses a temporary change (increase or decrease) in the aircraft's cruise altitude to avoid the conflict, returning the aircraft to the original cruise altitude after the conflict has been passed. The return to the original cruise altitude is optional (it actually simplifies the pattern if it is removed) and could be eliminated completely or only in the case of a step climb to a higher cruise altitude (to improve fuel efficiency). This type of step change in cruise altitude is a typical current day procedure and can be supported by current-day FMS.

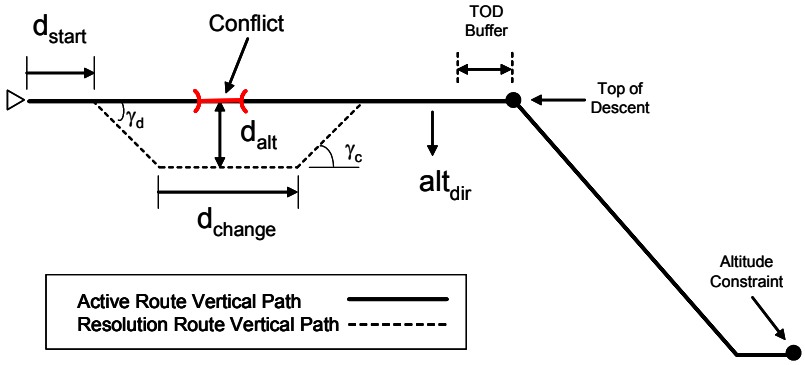

Figure 3. Cruise Step Climb/Descent vertical maneuver pattern.

The maneuver is defined by four parameters:

- altitude direction (alt_{dir}) – a discrete parameter that defines whether the step change increases or decreases the cruise altitude
- start distance (d_{start}) – a continuous parameter defining the distance from the aircraft's current position where the aircraft leaves the current cruise altitude

[§§] This pattern is in the early stages of development, so the exact parameter representation has not been finalized. It is not uncommon for some of the parameters to be changed during development to improve implementation. The parameters given are based on the current design and are sufficient for illustrative purposes.

- altitude change (d_{alt}) – a continuous parameter that defines the change in cruise altitude
- length of step (d_{change}) – a continuous parameter that defines how long the aircraft stays at the temporary cruise altitude

A minimum cruise step climb/descent maneuver is one which minimizes a combination of the altitude change and length of step.

D. Varying Resolution Complexity through Patterns

During the development for the Joint Ames/Langley Simulation, one overarching requirement identified to increase the user acceptability of the AOP's strategic resolutions was to return "simple" intuitive (i.e., user expected) maneuvers when possible, resorting to more complex maneuvers only when necessary. Figure 4 illustrates this idea. In scenario A, a single traffic aircraft (A) causes a conflict with the ownship aircraft and no other traffic in the area impact the potential resolution paths. In scenario B, the same traffic aircraft (A) causes an identical conflict with the ownship aircraft, but now an additional traffic aircraft (B) impacts the potential resolution paths. The original active route and proposed resolution path are the same in both scenarios.[¶] In scenario A, the conflict free resolution path is unnecessarily complex for the given conflict scenario. A simpler, more intuitive maneuver would be to deviate right of the active route using a lateral offset maneuver until avoiding traffic aircraft A and then to reconnect to the original active route. In scenario B, a lateral offset to the right may bring the resolution path into conflict with traffic aircraft B. In this scenario, the zigzag maneuver may be preferred over a very large lateral offset (left or right) which avoids both traffic aircraft. The addition of the second traffic aircraft increases the complexity of the conflict scenario and hence, increases the acceptable complexity of the resolution.

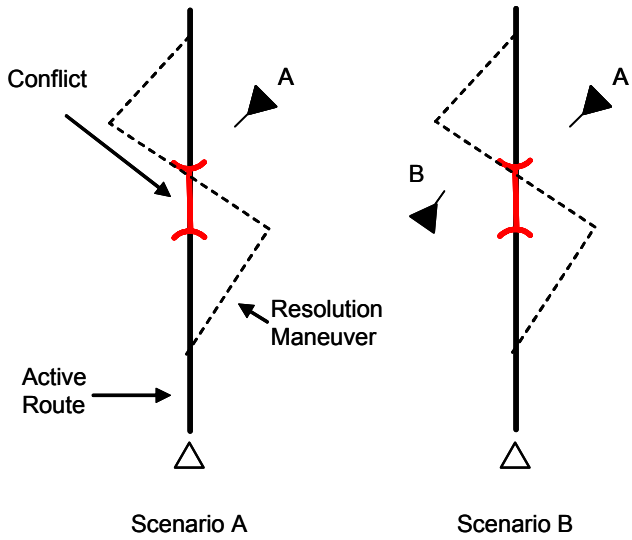

Figure 4. Resolution complexity examples.

The CR algorithm's use of parallel populations was described above as a method for providing resolution alternatives for display to the pilot. The algorithm also uses multiple populations, in this case sequential populations, to enable resolution complexity to grow as the conflict scenario's complexity requires. This approach is illustrated in Figure 5.

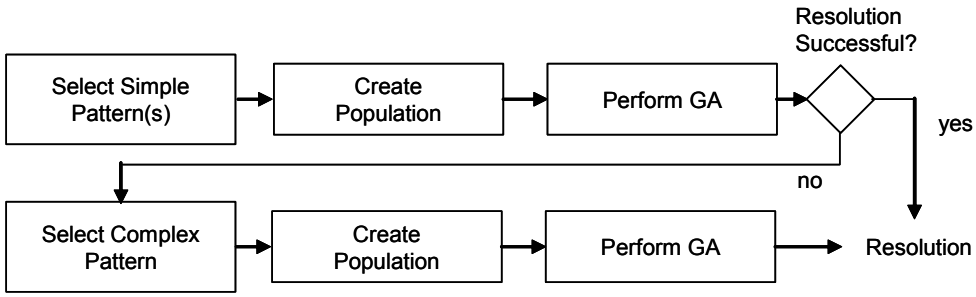

Figure 5. Use of sequential populations for controlling resolution complexity.

[¶] The zigzag resolution path illustrated is not one of the lateral patterns from Figure 2, but is a pattern that is being explored for more complex scenarios. It is actually a solution that was possible with the original CR algorithm, until it was constrained out due to its undesired effect (of crossing over the original path) during simple conflict scenarios.

The CR algorithm always starts by attempting to find a simple resolution maneuver to resolve a conflict. The current lateral and vertical maneuvers described above are all considered simple resolution maneuvers. If a simple maneuver pattern resolves the conflict, the resultant resolution is returned. If after application of the genetic algorithm a conflict free solution meeting all required constraints is not found, the algorithm has the option to attempt resolution with a more complex (flexible) resolution pattern (e.g., the zigzag pattern from Figure 4). A more flexible (less constrained) pattern has the ability to open up more of the resolution solution space and increase the chances of convergence to a conflict free solution. A new population is created with this new pattern and a new GA performed. The use of the sequential populations enables the algorithm to bias its resolutions towards simple maneuvers (the first population), while providing more complex solutions when necessary (the second population), without a dramatic increase in code or algorithmic complexity. Because additional processing time is required to find the more complex solution (since an entirely new GA is performed), the crew should be notified to alert them to expect the more complex solution after a processing delay. Maintaining crew awareness of the complexity (and extra processing time) in identifying a conflict free solution will increase the acceptability of the resultant complex maneuver.

There are many options available for creating the complex/flexible maneuver patterns. One option is to create "combination" patterns. These patterns would systematically attempt to apply two or more of the simple maneuver patterns to resolve the conflict (e.g., a lateral offset that uses a direct intercept path to a later active route leg instead of a return to its maneuver leg). If useful, results from the first population could be used to identify which combination of patterns would likely resolve the conflict. Another approach is to re-cast the original, unconstrained AOP CR algorithm (i.e., using very flexible route perturbations) as a complex maneuver pattern. In this case, all of the constraints applied in attempting to simplify this algorithm's results under simple conflict scenarios could be removed, regaining the power of its flexibility in modifying the path. The removal of these constraints is now acceptable because the pattern would only be applied when the simple maneuver patterns were unsuccessful.

Genetic Algorithm Details

A genetic algorithm attempts to determine a globally optimized solution to a problem through the structured evolution of candidate solutions. At each stage of the algorithm, candidate solutions are ranked based on a fitness function and the highest ranked candidates are allowed to propagate their traits to later stages. One of the significant benefits of a GA approach is that it is not constrained by domain-specific heuristics (for example, resolution heuristics based on conflict geometry), which makes it ideal for complex solution spaces such as those for conflict resolution. This section starts with some general genetic algorithm definitions, explains the general application to the pattern-based CR approach, and finishes with details on the more important design decisions made to implement a successful resolution algorithm.

A. Definitions

The following definitions describe concepts and features that are common to many genetic algorithms[12] and were applied to the pattern-based CR algorithm.

The **chromosome**[##] is the base unit of the genetic algorithm. It consists of a number (N_{var}) of variables or **genes** that define a particular candidate solution for the given problem. The actual candidate solution is constructed by **decoding** the chromosome. In the parameter-based GA approach, each chromosome represents a candidate resolution route.

A **population** is a collection of chromosomes that compete against each other to determine the "best" chromosome (i.e., the final resolution). The size of the population (N_{pop}) is the number of chromosomes it contains. The selection of the population size is one of the "tuning" characteristics of the GA. For the pattern-based approach, values of N_{pop} between 10 and 20 worked sufficiently well.

A **generation** is a state of the population. An initial population is generated for the first generation and then the population is modified (chromosomes are added/deleted/modified) from generation to generation through natural selection, mating and mutation. The number of generations (N_{gen}) is another tuning characteristic of the GA that

[##] Some references refer to the chromosome as a genome.

impacts convergence. For the pattern-based approach, N_{gen} = 10 worked sufficiently well, though the algorithm has been run with values as high as N_{gen} = 20.

Natural selection is the process of ranking the chromosomes within a generation (using a **fitness** computed for each chromosome) and then eliminating the lower-ranked chromosomes. The eliminated chromosomes will be replaced through mating (see below) to set up the next generation. The number of chromosomes kept after natural selection (N_{keep}) is another GA tuning characteristic. The choice of N_{keep} is related to the choice of N_{pop}. For the pattern-based approach, N_{keep} = 5 worked sufficiently well with N_{pop} between 10 and 20, but the combination N_{pop} = 20, N_{keep} = 10 has also been used.

Mating is the process by which the genes of two existing chromosomes (called **parents**) are combined to create new chromosomes (called **offspring)**. These offspring are used to re-populate the population after natural selection (that is, to add $N_{pop} - N_{keep}$ offspring to bring the population size back up to N_{pop}). Generally, the highest-ranked chromosomes are most likely to be selected as parents. Mating propagates the characteristics of these chromosomes to more of the population, but in new combinations and variations. Mating is the "local convergence" part of the GA algorithm.

Mutation is the process by which genes are randomly modified to change existing chromosomes into new candidate solutions. In a typical GA, the mutation occurs "in place" and the old chromosome value is not kept. In order to ensure that the next generation includes chromosomes that are at least as good as the best of the existing generation, a number (N_{elite}) of the highest ranked chromosomes are not allowed to mutate. For the pattern-based approach, N_{elite} = 2 worked sufficiently well. Since mutation is a purely random process, it enables the GA to search parts of the solution space away from the local convergence area.

Mutation is that part of GA that "looks" for global optima by randomly searching the solution space. If a mutated chromosome rises to the top of the population rankings, then it will be used during further mating and will drive the convergence (through mating) towards that part of the solution space. The main tuning characteristic of mutation is the mutation rate (μ) which determines how many parameters across all the mutable chromosomes are to be mutated. A typical GA will therefore select $N_{mut} = \mu \cdot (N_{pop} - N_{elite}) \cdot N_{var}$ genes at random to mutate. Increasing the mutation rate will slow convergence, but decrease the chance of converging to a local (not global) optimum. For the pattern-based approach, μ = 0.2 (that is, a 20% mutation rate) worked sufficiently well.

B. Overall Design for Conflict Resolution

The concepts of genetic algorithms lend themselves to a number of radically different implementations. At one extreme, once the chromosomes have been laid out in data structures in a computer program, the mating and mutation functions might be allowed to manipulate the chromosomes as strings of binary bits with no regard for the meaning of the data. This has strong parallels with the biological model. For the application of GA to conflict resolution, however, it was deemed more important to find an actual solution to the problem in a limited amount of time using limited computing resources than it was to simulate any particular model of evolution. The decision was therefore made to maximize the probability that each chromosome (whether in the initial population or as a result of mating or mutation) would be "viable," that is, that it would at least be possible to construct a realistic route (with or without conflicts) from that chromosome.

Accordingly, each chromosome in the GA for conflict resolution represents an attempted resolution according to a specific maneuver pattern. The meaning (and even number) of the genes is dependent on which pattern is to be executed. That is, the genes of the chromosome — the atoms of data to be manipulated by mating and mutation — are just the parameters of that chromosome's maneuver pattern. For example, a Lateral Offset pattern chromosome would have five genes (N_{var} = 5) corresponding to the parameters l_m, dir, d_{start}, d_{lat}, and d_{off}. The genes represent the recipe for changing the active route into the resolution route.

Moreover, the values of the genes refer to the actual units of the resolution route; for example, whenever the gene for d_{lat} has the value 5 it means an offset of 5 nautical miles. This means that when a chromosome for an offset pattern with d_{lat} = 5 passes this gene to an offspring (or keeps this gene's value after a mutation), the new (or modified) chromosome will also attempt to resolve the conflict using an offset of 5 nautical miles, and will not rescale the offset due to some other gene or genes that might have changed (as could happen if the value were stored as a percentage of an allowable range). The idea behind this was that the actual size or shape of the resolution maneuver represented by the chromosome would only change when mating or mutation "intended" for it to change. Otherwise, the value would be maintained at its geometric value. These design decisions have specific consequences for the random generation, mating, and mutation chromosomes that are described in more detail in later subsections.

Figure 6 illustrates the overall sequence of the genetic algorithm.

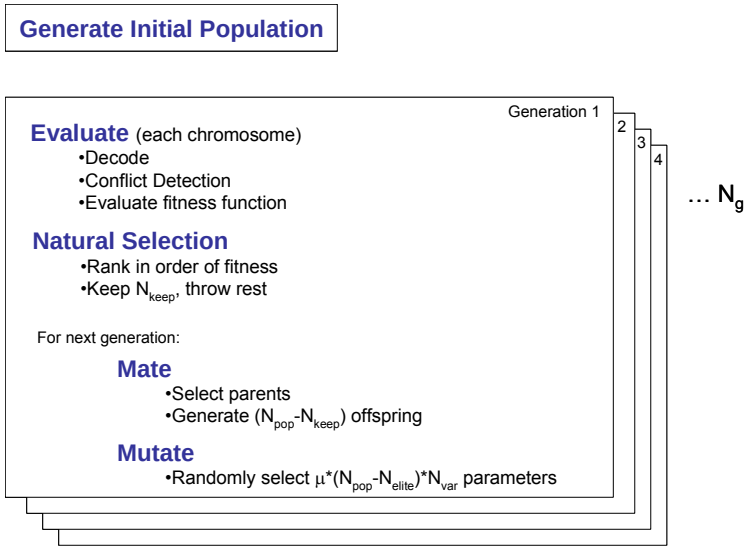

Figure 6. Execution sequence of the genetic algorithm.

The algorithm begins with the generation of the initial population. This initial population is just the creation of N_{pop} randomly generated chromosomes.

For every generation, each chromosome in the current population is evaluated in preparation for natural selection. The evaluation process starts with decoding the chromosome. For the parameter-based CR, chromosome decoding starts with creation of a candidate resolution route by applying the maneuver pattern parameters to the active route. In other words, decoding changes the route from the solid line to the dashed line in Figures 2 or 3. This route is then integrated to create a 4D trajectory prediction of the aircraft's future state. This trajectory is then compared against all known hazards (traffic and area hazards) and conflicts are determined. The maneuver parameters, resultant route, trajectory and CD results are all fed into the evaluation fitness function (described below). From this information, the evaluation fitness function returns a numeric fitness value that can be compared against the other candidate resolutions.

Once all chromosomes are evaluated, they are ranked by their fitness value with the highest (best) ranked chromosome having the lowest numeric fitness value (i.e., a minimization approach).

Once the fitness evaluation has been completed and the chromosomes ranked, only the top N_{keep} are kept. To re-populate the population, mating is performed to generate $N_{pop} - N_{keep}$ new chromosomes. The details of mating are described below. Once the population is full again, mutation is performed on $\mu \times 100$ percent of the genes of all non-elite chromosomes. In other words, $\mu \cdot \Sigma N_{var}$ genes are randomly mutated, where ΣN_{var} is the total number of genes summed over all chromosomes except the top N_{elite} chromosomes. (The total is not necessarily an exact multiple of $(N_{pop} - N_{elite})$ as in the typical GA, because different maneuver patterns may have different numbers of parameters.) The actual genes to mutate are randomly selected from the non-elite chromosomes. If a gene cannot be mutated (for example, if the parameter has only one possible value[***]), then that gene is not mutated and a new gene is randomly selected to be mutated instead.

Upon completing mating and mutation, the algorithm proceeds to the next generation and evaluates the newly generated chromosomes. This continues until N_{gen} generations have been evaluated. The output of the algorithm is the best conflict-free resolution route from the final generation; if all resolution candidates had conflicts, the output is an error message.

C. Fitness Function

For proper behavior in a genetic algorithm, the evaluation fitness function must be able to quantify the relative "goodness" between candidate solutions. All GA approaches evaluate a population's current generation of chromosomes and rank them in order of "best" to "worst." This ordering is important for selecting which chromosomes provide the genetic material for the chromosomes that will comprise the population's next generation. The rank order impacts:

- Which candidates are classified as "elite" chromosomes (i.e., passed unchanged to the next generation).

[***] This can happen when one parameter is a discrete set and is dependent on other chromosome parameters. Since only one parameter is mutated, the other values may restrict this set to a single value

- Which chromosomes are removed to make room for new chromosomes (only the top N_{keep} chromosomes are kept).
- Which chromosomes are available as parents for mating (only the top N_{keep}) and what is the relative probability of their being selected as a parent (higher probability for better ranked chromosomes).

This makes the selection of an appropriate fitness function critical for the successful convergence of the genetic algorithm.

In the application of GA to conflict resolution, the fitness function must achieve two distinct objectives: convergence to a conflict free solution and optimization of the resolution route. For the pattern-based GA approach, these two objectives are handled by using two separate fitness function approaches:

- A general "conflicted fitness function" (the same for all chromosomes, independent of pattern[†††]) if the candidate resolution has a conflict or does not meet its required time of arrival.
- A pattern-specific optimization fitness function (unique to each pattern) if the candidate resolution is conflict free and meets its required time of arrival.

The use of a conflicted fitness function that is separate from the optimization fitness function, along with a penalty system described next, enables the pattern-based GA to ignore optimization until a conflict free solution is achieved. Though this segregated approach may cause the algorithm to fail to identify a globally optimal resolution (or at least slow the convergence to this global optimum), it does enable the algorithm to expedite convergence to a conflict free solution. Since finding a conflict free solution is significantly more important for conflict resolution than optimization, this trade off is acceptable. A description of each of these fitness functions is provided in the following sections.

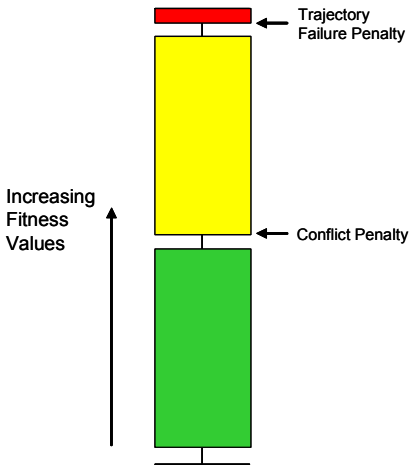

Figure 7. Classes of chromosome fitness.

To separate the optimization from conflict resolution, a penalty value is added to the results of the conflicted fitness function. This value is selected to ensure that all conflicted fitness function results exceed the maximum value of the worst case optimization fitness function result. This ensures that any chromosome with a conflict will be considered less fit than one without a conflict, thereby creating a "conflicted class" of chromosomes as shown in yellow in Figure 7. A chromosome without a conflict is in the "conflict free class" (green region in the figure) and is not given a penalty. Through the use of elite chromosomes, the best chromosome remains unchanged in the next generation. Therefore, as soon as at least one conflict free solution is identified, from that generation forward the "best" chromosome will always be a conflict free resolution.

A similar approach is used to remove chromosomes with bad routes or other trajectory failures from the population. A trajectory failure penalty is used in place of either fitness function in these cases. This penalty value is chosen to always exceed the worst case conflicted fitness function result, thus creating a "failure" class of chromosomes (red region in the figure). Since the bottom $N_{pop} - N_{keep}$ chromosomes are eliminated from the population in preparation for mating, failure chromosomes are always the first to be eliminated.

1. Conflicted Fitness Function

When ranking chromosomes containing conflicts, the Genetic Algorithm must determine which chromosomes are most likely (through mutation and mating) to generate conflict free offspring within later generations of the population. Furthermore, this "ability to be de-conflicted" must be quantifiable in a single value, the result of the conflicted fitness function. For the pattern-based GA approach, this trait is quantified by estimating the size of the **local conflict region** on each side of the chromosome's resultant 4D trajectory.

[†††] The conflicted fitness function actually is different for lateral versus vertical patterns, but this is based on the different relevant conflict geometries for these types of patterns and is not conceptually different.

Ideally, the conflict region would be defined as the region covered by all points where the ownship would be in a loss of separation (LOS) for all possible perturbations of its trajectory in the dimension of variation (left/right for a lateral maneuver, up/down for a vertical maneuver). But whether or not a particular off-track point would be in LOS often depends on when the aircraft arrives at that point, which is dependent on how the ownship's trajectory is perturbed. In practice, it seems impractical to assess all possible perturbations (a search space of many dimensions), and so the pattern-based GA algorithm uses the simplifying assumption that the trajectory is perturbed in a direction perpendicular to the direction of travel without any shift in along track times. For example, if the trajectory is perturbed one mile to the right, it means that at any time *t* the perturbed position will be one mile from the predicted (unperturbed) position at time *t*, in a direction perpendicular to the path. For each detected conflict, therefore, the local conflict region is defined as the locus of points along the trajectory that are in separation loss as the trajectory is perturbed perpendicularly to the left/right (for a lateral maneuver pattern) or up/down (for a vertical maneuver pattern) up to a predetermined distance from the predicted path.

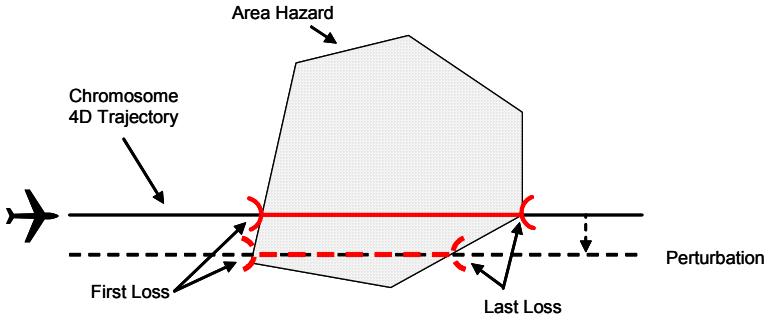

Figure 8. Example of local conflict region definition

The simplest illustration of this definition is when the trajectory of a candidate lateral resolution penetrates a static (non-moving, time invariant) area hazard as in Figure 8. For a static area hazard, the actual local conflict region is clearly visualized since along track time variations do not impact LOS; it is just the boundaries of the area hazard itself. As shown in the figure, as the trajectory is perturbed laterally to the left or right, the locus of points in separation loss are just the trajectory points within the area hazard. If the trajectory is perturbed sufficiently to the left and right, the local conflict region encompasses the entire area hazard.

For the pattern-based GA, a candidate chromosome's "ability to be de-conflicted" is equated with a measure of its resultant trajectory's penetration of all local conflict regions. Specifically, for each separation loss predicted along the trajectory's length, the conflicted fitness function finds the area of the conflict region to the left of (or below) the predicted trajectory, and the area of the conflict region to the right (or above), and takes the lesser (minimum) of these two areas. The chromosome's conflicted fitness value is then the sum of all of these minimum areas. The intuitiveness of this concept can best be illustrated with a simple area hazard example.

Figure 9 shows the resultant 4D trajectories from two candidate chromosomes and their penetrations of the same static area hazard. Since this example assumes that the resolution pattern is a lateral maneuver, the conflicted fitness function returns the smaller of the area hazard areas left or right of path. The minimum areas for the two chromosomes are shaded in the figure. When the two candidate chromosomes are being ranked by the pattern-based GA, no knowledge about how the algorithm may modify the lateral paths in later generations is

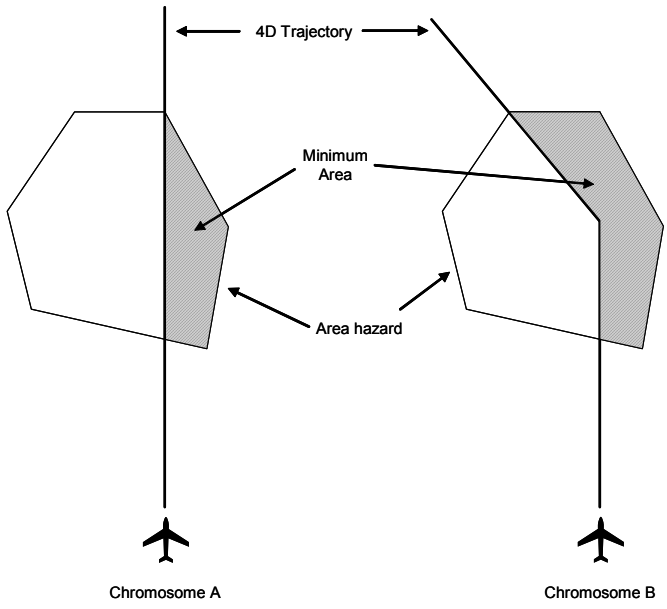

Figure 9. Conflicted fitness example

used.[‡‡‡] The two chromosomes are evaluated solely based on their penetration of the local conflict region. In this example, chromosome A is considered more fit than chromosome B since it penetrates less of the area hazard.

In the case of traffic hazards, the conflict region depends not just on the ownship trajectory but also on the relative motion of the traffic aircraft. The local conflict region for the conflicted fitness function can be estimated efficiently using the approach illustrated in Figure 10.

First, it is assumed that both the ownship and traffic trajectories have been divided into segments along which the aircraft travel in straight lines at constant groundspeed. Lateral turns and flight segments during which there are significant changes in flight path angle or groundspeed are all to be modeled by multiple segments such that the motion along each segment can be assumed linear.

Under these assumptions, for a single pair of segments the conflict region produced by a single traffic aircraft for a lateral maneuver can be determined by finding the first and last LOS points as the start of the segment is perturbed left or right perpendicular to the ownship trajectory segment as shown in Figure 10. The LOS points can be found by projecting the ownship's initial position on the segment, for example S_1, in the direction of motion relative to the traffic (that is, parallel to the vector V_{rel}). When this line intersects the lateral separation zone around the traffic aircraft—that is, at points F_1 and L_1 when we start at S_1—we must project the intersection points in the direction of the traffic aircraft's velocity (parallel to the vector V_t) onto the ownship's path (that is, onto a line parallel to V_o). Starting at S_1, then, we find that our first and last LOS occur at F_1' and L_1'. When we perturb the ownship's trajectory so that it starts at S_2, however, we find that points F_2 and L_2 on the separation zone result in LOS points at F_2' and L_2' on the perturbed trajectory segment. Continuing this process as we perturb the ownship trajectory left or right, the resulting first and last LOS points trace the boundary of the shaded ellipse shown in the figure.[§§§]

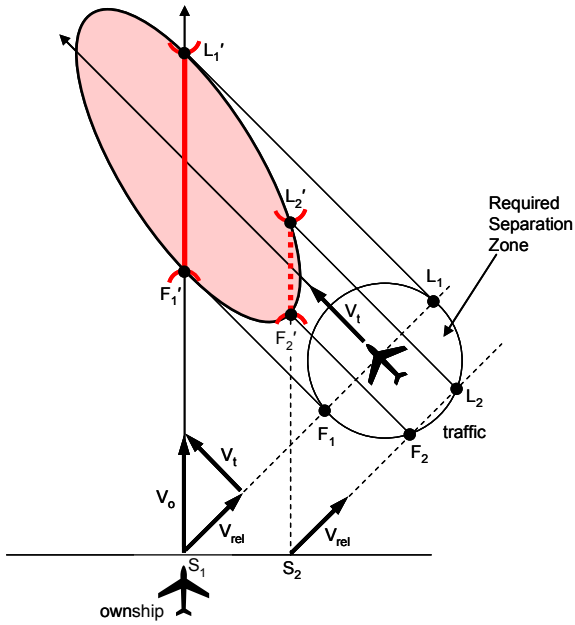

Figure 10. Geometry for estimating local conflict region for traffic aircraft hazards

Depending on exactly where the LOS points fall for this pair of segments, all, part, or none of the resulting ellipse might contribute to the local conflict region. For example, any portion of the ellipse that is "behind" the ownship's initial position is excluded, since the trajectory segment cannot "sweep" through those points. The algorithm therefore clips the ellipse using a polygon. In the typical case, two sides of the polygon are parallel to the ownship trajectory segment and are separated from it by the maximum "sweep" distance (left/right), and the other two sides cross the ownship's trajectory segment, one intersecting the segment at the later of the ownship or traffic starting times, and the other intersecting the segment at the earlier of the ownship or traffic ending times. In

[‡‡‡] Using information about the "direction of change" for later generations would add heuristics to the genetic algorithm approach, creating a hybrid algorithm. Though hybrid approaches can improve convergence in some situations, they can hurt convergence in others. It was decided to not add heuristics to the pattern-based GA.

[§§§] In the special case where the relative velocity is perpendicular to the ownship path, and in the case of zero relative velocity, either the LOS points will trace two parallel lines bounding a strip in the plane, or there will be no LOS at all. In all other cases the LOS points trace an ellipse, though it may not be positioned as shown in the example.

other words, this pair of segments can only contribute LOS points when both aircraft are predicted to be on their respective segments, and only within a limited distance from the ownship's path. In the case where the ellipse lies entirely outside the clipping polygon, the pair of segments does not contribute any area to the local conflict region. The total contribution to the local conflict region for this traffic aircraft is the union of the regions contributed by all pairs of segments between the ownship and the traffic.

The total local conflict region need not be solely due to one hazard; it is the culmination of all the hazards in the area near the resolution path. It may even be partially determined by hazards with which the ownship is not originally in conflict, but that contribute LOS points only on one side of the ownship trajectory when it is "swept." Parts of the conflict region where the ownship would be in LOS simultaneously with multiple hazards are not multiply counted; a given point is either in the conflict region (whether due to one hazard or several) or out of the conflict region. Hence in the general case, the local conflict region is the union of all the conflict regions produced by one or more hazards, including area hazards and/or traffic aircraft.[¶¶]

Using the local conflict region enables the pattern-based GA's fitness function to evaluate the airspace around a conflicted trajectory, not just along it, to distinguish a chromosome's relative rank. This is a critical aspect of the algorithm that enables the successful convergence to conflict free solutions. Figure 11 illustrates an example of the criticality of this approach. In the figure, the two cases are identical except that in Case A, the candidate chromosome's trajectory passes through the middle of an oncoming line of traffic, whereas in Case B the candidate trajectory crosses one end of the line of traffic. In both cases, the detected conflict on the candidate trajectory is identical except for the identity of the conflicting traffic aircraft (aircraft 2 in case A and aircraft 1 in Case B) and possibly the absolute position of the conflict in the airspace (not shown). If the fitness criterion were based solely on the size, shape, and position of the conflict relative to the ownship trajectory and not on the conflict region surrounding the trajectory, then these two candidate solutions would have equal fitness values. Moreover, there would be candidate trajectories both left and right of the current candidate in each case (passing at equal distances on either side of the conflict aircraft) that would be ranked better than the current candidate and equal to each other. Such a fitness function would inform the GA that there were local optima on routes that passed between two aircraft, even though it is impossible to pass between these traffic aircraft without conflict. This could cause the GA to spend excessive resources probing these parts of the airspace. Case B, however, is intuitively closer to being conflict free than Case A and any new candidate trajectory to the left of the current trajectory in Case B would be even closer to resolution or would resolve the conflict entirely. These properties are clearly represented by the

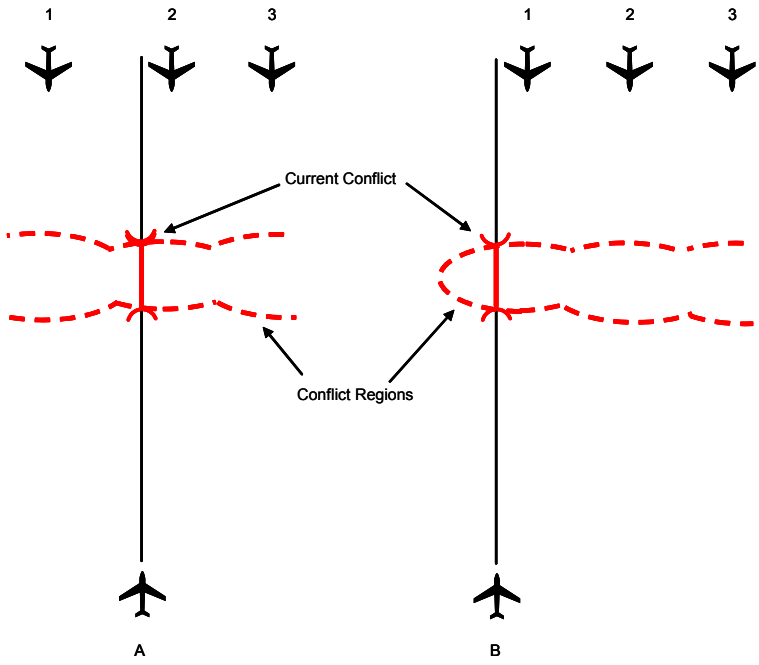

Figure 11. **Example of the local conflict region's spatial impacts**

[¶¶] In order to perform all the necessary calculations, it is convenient to use polygons to approximate the elliptical regions that result from traffic hazards. Clipping these regions as well as taking the unions of regions can then be performed using efficient and relatively simple algorithms on polygons.

fitness function based on local conflict region (in red), which assigns a better fitness to Case B than to Case A (due to the small conflict region on the left side of the trajectory) and would assign an even better fitness to any variant of Case B that passed further to the left of traffic aircraft 1. Therefore this fitness function may be expected to "guide" the GA towards the hole in the oncoming line of traffic aircraft.

A subtle, but important, effect of using the local conflict region estimate as the foundation of the conflicted fitness function is that it works just as effectively for multiple traffic and area hazards as it does for a single hazard. Each hazard potentially adds to the conflict region, but the number of hazards that the ownship is in conflict with at any given location is irrelevant; only the fact that separation has been lost is relevant. Also, whether the source of the loss is an area hazard or a traffic hazard is irrelevant. All locations in loss are equally bad and must be avoided. This enables the algorithm to deal with area and traffic hazards in a naturally consistent way without having to create an artificial approach to weighing the impacts of different hazards. This ensures that the fitness function doesn't abnormally favor the avoidance of one type of hazard over another, which is a heuristic that may work well in some situations, but not in others.

2. Fitness when an RTA Is Not Met

The fitness function accounts for whether an RTA (if any) in the resolution route is met in addition to any conflicts that may arise. If the candidate resolution route fails to meet an RTA, the "conflict" penalty is added to the fitness function even if no conflict with a hazard has been found (though only one penalty is added if both a conflict and an unmet RTA exist). The rationale for this is that the missed RTA makes the proposed route unacceptable as a resolution, just as a conflict would.[###] In addition to the conflict penalty and the cost for local conflict regions (if any), the fitness function finds the difference between the estimated and required times of arrival at the time-constrained fix and adds to the numerical fitness a cost proportional to the absolute value of this time difference.

3. Optimization Fitness Function

If the trajectory is conflict-free and meets all identified RTAs, it will be assigned a fitness value based on optimality (range shown in green in Figure 7). This fitness value is computed according to a formula specific to the maneuver pattern used by the chromosome. That is, for each maneuver pattern, there is a function that implements the optimization criteria presented in the descriptions of the maneuver patterns above. These functions are typically based on the geometry of the resolution maneuver, assigning optimality based on user-defined preferences for that specific maneuver.

D. Creating Random Chromosomes

In order to fill the first generation of a population, the GA creates N_{pop} new randomly-selected chromosomes. For each chromosome, first the maneuver pattern (which determines the type of chromosome) is selected at random from the patterns available in the given population. (The relative probabilities of the patterns need not be equal.) The genes of the new chromosome are then set to random values.

Due to the geometry of the active route and certain other obvious requirements (such as the required distance to initiate a turn before a fly-by waypoint or the need to initiate the resolution maneuver before the conflict first loss location), not all values to which the genes of a chromosome might have been assigned will result in a viable resolution route. To reduce the chances of creating non-viable resolution routes, the algorithm:

- arranges the parameters of each maneuver pattern in a definite order (in the descriptions of the maneuver patterns earlier, the parameters are listed in this order)
- assigns each gene a fixed parameter index in sequence from 1 to the number of genes
- places limits or restrictions on the value of each gene depending on the active route geometry and the values of lower-numbered parameters

In other words, the genes are initialized in the sequence defined by their index numbers and the permissible values for a gene may depend on genes that have already been initialized. The approach does not guarantee a viable resolution route, but is designed to eliminate values or combinations of values that can easily be ruled out in advance; dramatically reducing the number of non-viable resolution routes created within the population. For example, a leg that is too short to support even a minimal-size offset maneuver would be eliminated from the set of

[###] Other approaches could be investigated. For example, an RTA only penalty that is less than the conflict penalty could be added, creating a (more desirable than conflicted) "Conflict Free - Unmet RTA" class of chromosome.

potential maneuver legs allowed for a Lateral Offset Pattern chromosome. This approach improves the efficiency of the GA; a critical design requirement when the size of the population and number of generations is limited.

E. Mating

The genetic algorithm uses a rank-weighted algorithm when selecting chromosomes for mating. That is, this algorithm selects a "mother" (first parent) and "father" (second parent) from among the N_{keep} surviving chromosomes of the previous generation, with a probability distribution such that the probability of selecting the i-th best chromosome as the "mother" is $N_{keep} + 1 - i$ times as large as the probability of selecting the worst chromosome. (That is, the best chromosome is N_{keep} times as likely as the worst, the second best is $N_{keep} - 1$ times as likely, and so forth.) For example, if $N_{keep} = 4$, the probabilities assigned to each of the four chromosomes are 0.4, 0.3, 0.2, and 0.1 respectively. The probability of selecting each chromosome as the "father" is computed similarly, except that the chromosome already selected as the "mother" cannot be selected again, and so the other chromosomes' probabilities are increased proportionately. This prevents a chromosome from mating with itself. In addition, if the population contains chromosomes for multiple different patterns and the randomly selected chromosomes represent two patterns that cannot mate with each other, another pair of chromosomes will be selected at random (currently, only chromosomes of the same pattern type are able to mate). Usually two offspring will be created when a pair of chromosomes is mated, but if just one chromosome is needed to complete a new generation, only the "first" offspring will be produced.

In order to mate two chromosomes of the same pattern type, a crossover index is selected randomly from the same sequence of parameter indexes that are used in the initialization of a random chromosome (see above). The values of the genes of each offspring are then determined as follows (where m_n is the value of the mother's nth parameter and p_n is the value of the father's nth parameter):

For a discrete gene[****] with parameter index i:

If i is below the crossover index:
value = m_i (first offspring)
value = p_i (second offspring)
If i is at or above the crossover index:
value = p_i (first offspring)
value = m_i (second offspring)

For a continuous gene[††††] with parameter index i:

If i is below the crossover index:
value = m_i (first offspring)
value = p_i (second offspring)
If i is at or above the crossover index, randomly select a value β from the continuous range from 0 to 1:
value = $\beta \cdot m_i + (1 - \beta) \cdot p_i$ (first offspring)
value = $\beta \cdot p_i + (1 - \beta) \cdot m_i$ (second offspring)

In words, discrete genes at or above the crossover index are swapped, where continuous genes are blended. As in the case of a randomly generated chromosome, not all conceivable combinations of parameter values will yield a viable chromosome. The procedure for setting the value of a gene in the offspring therefore performs a "range check" after the steps shown above. If the value of the gene is not within the set or range permitted for that gene given the geometry of the active route and the values of genes already chosen (that is, with lower parameter

[****] Discrete genes occur when the genes represents a member of a finite set. The maneuver leg (l_m) is a classic example of a maneuver parameter represented by a discrete gene

[††††] A continuous gene is any gene that varies continuously between two extreme (minimum and maximum) values. The start distance (d_{start}) is a classic example.

indexes), the value of the gene is brought back into the legal range, if possible. In the case of a continuous gene whose value was outside of the permitted value range, the value is set to the closest limit value. In the case of a discrete gene, if the gene's value is not within the allowable set, the offspring chromosome is rejected (which may result in the need later to mate another pair of chromosomes in order to replace the missing offspring).

This design, employing a single crossover point and swapping/blending genes at or "above" the crossover, is designed to provide adequate "mixing" of the genes of the two parents (and an adequate probability of creating an offspring distinct from all other chromosomes) while at the same time maintaining a good likelihood that the offspring will be viable, since the genes before the crossover index are all copied from a single viable parent chromosome (and are therefore a viable combination of genes). The (not coincidental) fact that higher-indexed genes tend to be continuous (and so can usually be adjusted to new limits as required) also helps to produce viable offspring.

The need for range checking could be reduced (in continuous genes) if the numeric values of the genes were interpreted relative to the permissible ranges of values of the corresponding parameters (i.e., a percentage of the total range) rather than to literal parameter values as in this design. This design alternative was rejected on the argument that it would not as effectively propagate the actual characteristics of the parent chromosomes to their offspring as was desired.

F. Mutation

Before performing mutation, the GA computes the total number of genes summed across all the chromosomes in the current generation of the population (which may vary from generation to generation due to different mixtures of patterns), and multiplies this figure by μ to find N_{mut}, the total number of genes to mutate. It then performs the following procedure N_{mut} times:

1. Select a chromosome at random from the population.
2. Select a gene at random within the selected chromosome.
3. Mutate the selected gene; that is, set it to a new random value.

In some cases involving discrete genes (very rarely for continuous genes), Step 3 cannot be executed because the gene is already set to the only possible value it can have under the given active route geometry and the values of the genes with lower indexes. In that case, the mutation fails.

Even if it is possible to mutate the selected gene, the chromosome must be checked for viability after Step 3 by range checking the values of the genes that have higher indexes than the mutated gene (and whose permissible values might therefore depend on the mutated value). For each such gene in sequence, the algorithm finds what range of values are now permissible, and either brings the gene into its legal range (for a continuous gene, as in the mating procedure) or causes the mutation to fail.

If the mutation fails for any reason, the chromosome is restored to the state it had before the mutation and a new chromosome and/or gene is selected for mutation instead.

In the application of GA to a population of a single maneuver pattern type, the chromosomes all have the same number of genes, so a uniform distribution for chromosomes in Step 1 would result in a uniform distribution over all genes. In the current implementation of GA for CR, the numbers of parameters in the various chromosomes in a population have been equal or nearly equal, so a uniform distribution over chromosomes leads to a nearly uniform distribution over genes. If the numbers of parameters differed significantly between chromosomes, the distribution over chromosomes could remain uniform, yielding a probability for each gene that is inversely proportional to the size of its chromosome, or the chromosomes could be weighted so as to yield a uniform distribution over all the genes. Which design would perform better is an open question.

Preliminary Results: Performance Characteristics of the Implemented Algorithm

The new CR algorithm has been implemented within the existing AOP software prototype[13] and successfully demonstrated under a variety of traffic and area hazard scenarios. Currently, only the Lateral Offset, Direct Intercept Path, and Cruise Step Climb/Descent Patterns have been implemented. The Path Stretch Pattern is under development and is scheduled for completion in September of this year. The scenarios used for demonstrating the new CR algorithm's performance were created as overall AOP demonstration scenarios and do not represent an exhaustive set of CR test scenarios covering a wide variety of conflict geometries, hazard loads, etc. They do, however, represent typical conceptual scenarios under which the CR algorithm is expected to have acceptable performance. For these scenarios, the CR algorithm reliably converged to the expected conflict-free, minimal maneuver geometry resolution paths.

The AOP prototype is implemented in Visual C++ (with a very small amount of legacy FORTRAN code) and runs on a workstation running Windows 2000 with dual Xeon processors rated at 2.4GHz or better. The AOP shares this workstation with two other systems in the simulation environment (see reference 13 for details), but one processor is dedicated to the AOP process. Within the AOP software, the strategic CR function runs on its own thread, but shares the dedicated AOP processor with other AOP functionality. The current specifications for the strategic CR algorithm are:

- time horizon for conflict detection (CD look-ahead) = 10 minutes
- time horizon for conflict free resolutions (Cr look-ahead) = 20 minutes
- $N_{gen} = 20$; $N_{pop} = 20$; $N_{keep} = 10$; $N_{elite} = 2$; $\mu = 0.2$

For a typical simulation scenario, the new strategic CR function is able to complete two parallel populations (one vertical and one lateral with one maneuver pattern each) for 10 traffic aircraft and 6 area hazards averaging 4.9 seconds per population. During a typical run, trajectory generation takes approximately 40% of the processing time, CD calculations performed during CR take approximately 20% of the processing time, and the remaining 40% is attributed to the genetic algorithm (including fitness function calculations). Since additional AOP functionality is being performed in parallel with the strategic CR function, these results represent conservative values for the algorithm itself under the defined scenario (population completion times under 4 seconds have been observed for just the algorithm).

These early performance results are consistent with the acceptable performance of the previous strategic CR function. In the above scenario, the number of generations and the population size have been increased to their maximums to stress the new algorithm's performance. Successful convergence to conflict free solutions has been observed with half the population size and half the number of generations. Also, the implementation has not yet been optimized for computation performance, so further improvement in population completion times is anticipated. Since the performance of the CR function is inexorably tied to the test scenario, exhaustive testing of this function under a wide variety of conflict scenarios would be required to draw any statistically significant conclusions about the ultimate acceptability of the algorithm. That being said, the algorithm's early performance results are encouraging.

Concluding Remarks

A new strategic conflict resolution algorithm for NASA Langley's AOP DST has been presented to address evolving research requirements for resolution routes with enhanced user acceptability. This CR algorithm is based on pre-defined maneuver patterns and uses a new genetic algorithm approach for selection and optimization of resolution routes. A new conflicted fitness function that uses estimates of the local conflict region to find conflict free "holes" between hazards has also been presented. The basic CR algorithm has been successfully implemented and demonstrated for the Lateral Offset, Direct Intercept Path, and Cruise Step Climb/Descent Patterns. The use of parallel populations to return a single lateral and vertical resolution for display to the flight crew has also been demonstrated.

Several next steps are planned for the continued development of the algorithm. Significant testing to determine the algorithm's acceptability under a wider range of conflict scenarios is planned. Enhancements to the competition approach between different patterns in the same population will be developed. Extensions of the mating algorithms for parents of different maneuver patterns with similar (but not exactly the same) genes will be designed. Development of a complex/flexible maneuver pattern and serial populations will be performed to handle higher complexity conflict scenarios. And tuning of the GA parameters (N_{pop}, N_{keep}, μ, etc.) will be investigated to improve the convergence rate, as necessary.

Experiments investigating key aspects of autonomous flight concepts using the new CR algorithm are planned for later this year and next. During development for these experiments, it is anticipated that new requirements for the AOP's strategic CR algorithm will be discovered. Modifications to the CR algorithm to support these new requirements will be performed, as necessary.

Acknowledgements

This effort was conducted for NASA's Langley Research Center under subcontract to Raytheon Corporation. The authors would like to thank the NASA researchers and the Raytheon contract staff for their continued support. The authors would also like to acknowledge Steve DePascale's (L-3 Titan) support in designing and implementing several of the current maneuver patterns.

References

[1]"Final Report of the RTCA Task Force 3: Free Flight Implementation," RTCA, Inc., Washington DC, October 1995.

[2]Cotton, W.B.: "New Directions in Air Traffic Control at Kennedy Airport," Massachusetts Institute of Technology, 1965.

[3]Wichman, K., Carlsson, G., and Lindberg, L.: "Flight Trials: "Runway-to-Runway" Required Time of Arrival Evaluations for Time-Based ATM Environment," Proceedings of the 2002 AIAA Conference on Guidance, Navigation, and Control, Monterey, CA.

[4]Green, S.M., Bilimoria, K.D., and Ballin, M.G.: "Distributed Air/Ground Traffic Management for En Route Operations," Air Traffic Control Quarterly, Vol. 9, No. 4, 2001.

[5]Ballin, M.G., Sharma, V., Vivona, R.A., Johnson, E.J., and Ramiscal, E.: "A Flight Deck Decision Support Tool for Autonomous Airborne Operations," Proceedings of the 2002 AIAA Conference on Guidance, Navigation, and Control, Monterey, CA.

[6]Mondoloni, S., Conway, S.: "An Airborne Conflict Resolution Approach Using A Genetic Algorithm," Proceedings of the 2001 AIAA Conference on Guidance, Navigation, and Control, AIAA-2001-4054, Montreal, QC.

[7]Barhydt, R., Kopardekar, P., Battiste, V., Doble, N., Johnson, W., Lee, P., Prevot, T., Smith, N.: "Joint NASA Ames/Langley Experimental Evaluation of Integrated Air/Ground Operations for En Route Free Maneuvering," USA/Europe ATM Seminar 2005.

[8]Doble, N., Barhydt, R., and Hitt, J.: "Distributed Conflict Management in En Route Airspace: Human-in-the-Loop Results," 24th DASC, Washington, 2005.

[9]Doble, N., Barhydt, R., and Krishnamurthy, K.: "Airborne Management of Traffic Conflicts in Descent with Arrival Constraints," 24th DASC, Washington, 2005.

[10]Doble, N., Barhydt, R., and Krishnamurthy, K.: "Pilot Preference, Compliance, and Performance with an Airborne Conflict Management Toolset," 2005 International Symposium on Aviation Psychology.

[11]Mondoloni, S., Ballin, M., and Palmer, M.: "Airborne Conflict Resolution for Flow-Restricted Transition Airspace," AIAA's 3rd Annual Aviation Technology, Integration and operations (ATIO) Technical Forum, November, 2003.

[12]Haupt and Haupt: "Practical Genetic Algorithms," 2nd Edition, Wiley, 2004.

[13]Karr, D.A., Roscoe, D.A., and Vivona, R.A.: "An Integrated Flight-Deck Decision-Support Tool in an Autonomous Flight Simulation," Proceedings of the 2004 AIAA Conference on Modeling and Simulation Technologies, Providence, RI.