Aircraft Routing in the Presence of Hazardous Weather

A Dissertation Presented

by

Joseph Michael Prete

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

August 2007

Stony Brook University

The Graduate School

Joseph Michael Prete

We, the dissertation committee for the above candidate for the

Doctor of Philosophy in Computer Science degree, hereby recommend

acceptance of this dissertation.

Joseph S.B. Mitchell – Dissertation Advisor
Professor, Department of Applied Mathematics and Statistics

Michael A. Bender – Chairperson of Defense
Associate Professor, Department of Computer Science

Esther M. Arkin
Professor, Department of Applied Mathematics and Statistics

Alan Tucker
Distinguished Teaching Professor,
Department of Applied Mathematics and Statistics

This dissertation is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Aircraft Routing in the Presence of Hazardous Weather

by

Joseph Michael Prete

Doctor of Philosophy

in

Computer Science

Stony Brook University

2007

Air traffic control in the 21st century will require the application of modern computer science techniques in order accommodate the future air travel demands of a more mobile population. One of the biggest challenges to air travel throughput is the presence of hazardous weather, which causes delays, cancellations, and rerouting of aircraft. The Flow-Based Route Planner (FBRP) is an algorithmic system designed to route flows of aircraft between designated origin and destination points while avoiding hazardous, time-varying weather systems. The objective is to compute shortest (minimum-time) routes that are available for safe passage of aircraft during a specified window of time, while avoiding time-varying hazardous weather constraints that come from a weather prediction model. To maximize throughput, multiple routes are required, and these routes must be chosen to avoid conflicts arising among aircraft on different routes. While the general form of this constrained optimal routing problem is NP-complete, the FBRP applies heuristics to constrain the path search algorithm in order to obtain good solutions within a reasonable running time. The FBRP is examined as a tool for solving routing problems, as a method for resolving airspace conflicts, and as a capacity estimation tool. It is shown to be practical and is compared with historical flight data and with alternative methods.

Table of Contents

Table of Figures

Acknowledgements

The author thanks Dr. Joseph S.B. Mitchell, the author's advisor, for five years of good advice, research direction, and immense patience. Without Dr. Mitchell's assistance, this thesis could not have been completed.

Dr. Mitchell, Dr. Jimmy Krozel, Steve Penny, Valentin Polishchuk, Dr. Phil Smith, and Dr. Anthony D. Andre are co-authors of the papers and articles that this thesis draws much of its material from. Their contributions have been essential to the research, whether directly as co-authors of papers or indirectly for advice, technical input and assistance, domain knowledge, and the formulation of algorithmic problems.

Dr. Michael Bender was a crucial advisor and mentor during a large part of the author's graduate career. His advice and ideas are remembered to the present day. The author would like to thank him for his guidance.

The author thanks the following for technical guidance that helped direct and focus the research effort: Dr. Matt Jardin, the NASA Technical Monitor; Rob Fong, the NASA VAMS Technical Monitor; Doug Isaacson, the NGATS Contract Monitor. The author also thanks Dr. George Meyer for valuable technical guidance on the application domain.

This research was funded by NASA Ames Research Center, via a grant to Stony Brook University through J.S.B. Mitchell under contract NAG2-1620; via a subcontract from Metron Aviation, with support under contract NAS2-02075 for the NASA Ames Research Center's Virtual Airspace Modeling and Simulation Project; via the NASA Ames Research Center under the Next Generation Air Transportation System Project contract NNA07BA84C; via several grants from the National Science Foundation (CCR-0098172, ACI-0328930, CCF-0431030, CCF-0528209); and via a grant by the U.S.-Israel Binational Science Foundation (2000160).

The author would like to thank NASA Ames Research Center and the Virtual Airspace Modeling and Simulation (VAMS) Project Manager, Mr. Harry Swenson.

Chapter 1: Introduction

1.1 Background

1.1.1 Current Operations

Air traffic control (ATC) encompasses technologies and techniques that direct aircraft from one point in space-time to another, with specified constraints, while maintaining separation between all aircraft. Aircraft are normally separated from each other in the horizontal plane by a separation standard δ as well as by a vertical distance of 1000 feet. They are also kept at a different minimum distance ε from all hazardous weather. Historically, these techniques have relied on human decision making and planning, both in the form of real-time decision making by pilots, aircraft dispatchers, and ATC personnel, and in the form of preplanned solutions that cover a wide range of weather scenarios, and which can be selected quickly in the face of a complex weather problem. The air traffic control system's current design is intimately involved in the selection of theoretical problems to examine, so it is worthwhile to outline its current operations.

Airspace above the United States is divided into twenty centers and roughly 600 sectors, each center containing several sectors – although a sector may straddle the boundary between two centers. Each of these centers and sectors is a simple polygon and is managed by one or more air traffic controllers. Controllers receive positional and other reports from aircraft and monitor any available radar in order to instruct aircraft, but the instructions are necessarily short and to the point while being specifically designed to avoid potential miscommunication on the radio.

Once an aircraft is making an approach to an airport, it is managed in the transition airspace, which extends for 200 nautical miles around all major airports (with potential for overlap between transitional airspace zones). Today, aircraft approaching an airport are normally routed on Standard Arrival Routes or STARs, while, similarly, Standard Instrument Departure Routes or SIDs are followed by aircraft leaving an airport. These routes are fixed and do not change to reflect current traffic patterns, demand, or hazardous weather around the airport.

There is a large network of navigational aids available for pilots and controllers to use in order to determine the location of an aircraft, but the data is only accurate enough to allow for a standard 8-nmi-wide lane for an aircraft to follow. Global Positioning System technology, while very precise, has not yet been deployed on the majority of aircraft and is not considered required equipment for aircraft at this time. Aircraft using instrument flight rules fly from navaid to navaid, using directional/range information to guarantee their position, and relying on controllers to advise them of other aircraft in the area.

1.1.2 Future Scenarios

It is expected that in the near future the demands on the national airspace will continue to increase as they have in the past. Current methods of air traffic control are insufficient to handle this demand in the face of hazardous weather and other constraints.

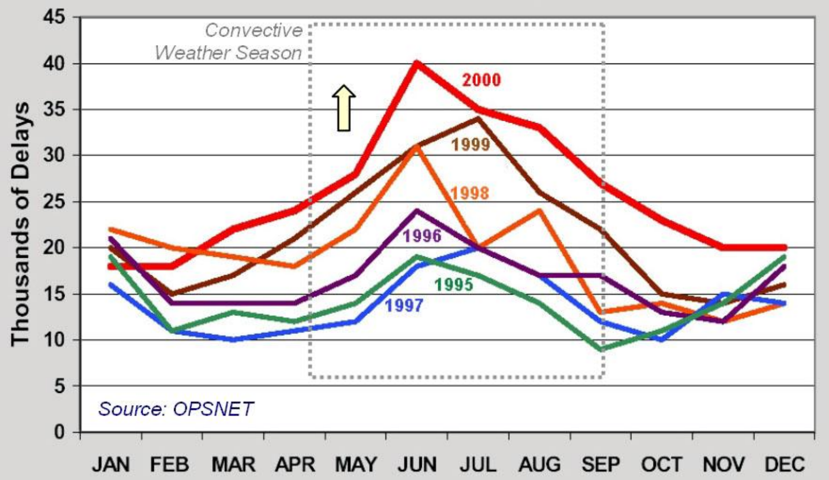


Figure 1. Yearly trends (1995-2000) for weather-related delays.

In order to improve the capacity of the national airspace, it is desirable to find new ways to route aircraft safely around hazardous weather.

1.2 Motivation

Air traffic control has many constraints, although we do not address all of the constraints when defining a problem statement for FBRP. When a single aircraft is required to remain five miles from any other aircraft and from hazardous weather, the sky transforms from a clear and empty space to a crowded and cluttered highway of air traffic. Hazardous weather systems extending over hundreds or thousands of miles are commonplace and are a significant constraint in the choices made by pilots and controllers. Fuel constraints and fuel weights place a premium on energy efficiency and can make or break the airlines that bind the country together. Wind conditions change the possible speed and efficiency of an aircraft, so that a great-circle flight is no longer the quickest and most efficient way to make a trip.

This human-focused approach has several problems which make ATC an obvious candidate for computer science research. One, there are fundamental limits on the ability of a single person to monitor and guide many aircraft towards their destinations. Two, even under the assumption of constant altitude, the weather avoidance problem is a three-dimensional problem that is difficult to properly visualize. Three, statically solving

NWS Level	Color	Rainfall Rate (mm/hr)	Reflectivity (dBZ)	Type
0	None	<0.49	dBZ<18	None
1	Light Green	0.49 - 2.7	18≤ dBZ<30	Light Mist
2	Dark Green	2.7 - 13.3	30≤ dBZ <41	Mod.
3	Yellow	13.3 - 27.3	41≤ dBZ <46	Heavy
4	Orange	27.3 - 48.6	46≤ dBZ <50	Very Heavy
5	Deep Orange	48.6-133.2	50≤ dBZ <57	Intense
6	Red	>133.2	57≤ dBZ	Extreme

Figure 2. NWS Standard Reflectivity Levels and weather classifications.

routing problems is a virtual guarantee that the solutions will be far from optimal, leading to wastes of time and fuel for thousands of flights every year and man-hour losses measured in the millions. The large error margins used by human air traffic controllers mean that the likely solution to a problem will be to delay or cancel flights even when there was no need to delay or cancel most of the affected flights.

There are clear and obvious advantages to the computational approach. Even when finding an optimal solution is NP-hard, good approximations are comparatively easy to solve for. They are likely to result in better solution rates and higher-quality solutions, increasing the amount of navigable traffic while making each individual flight more efficient. A computational approach is also capable of dealing with far more flights than a human flight controller could, and of plotting courses for those flights that require less leeway for solution error.

1.3 Assumptions

We have made several assumptions which hold for virtually all of our experiments and results. Unless otherwise stated, the assumed horizontal separation standard for all aircraft is 5 nmi. When looking at transition airspace problems, we generally focus on arrival routes, and not departure routes. The problems are very similar, but the arrival problem is slightly harder because departures can always be assumed to be well-spaced coming from the airport. For the transition airspace, the routing problem is assumed to be between the 200-nmi range ring and the metering fix, while for en route routing problems, we only treat one sector at a time. We do not address the questions of how to coordinate between sectors, how to transition an aircraft from en route flight to the transition airspace, or how to guide aircraft from the metering fix to the runways. We have assumed that all aircraft under consideration can be considered to be at a single flight level for purposes of conflict avoidance.

1.4 Problems and Results

We solve several problems in this thesis. In the first chapter, we examine a system, the Flow-Based Route Planner (FBRP), devised to solve two-dimensional dynamic routing problems. This system forms the foundation for most of the other research. In the second chapter, we examine the results of the initial application of the FBRP to the transitional airspace around airports, and examine some of its performance characteristics. In the third chapter, we investigate another method of routing aircraft around dynamic obstacles, Variations on Standard Arrival Routes. Additionally, we investigate using the FBRP to simulate free flight of aircraft individually to their destinations rather than using precalculated routes. All three methods are compared for the transitional airspace problem.

Chapter 2: The Flow-Based Route Planner

The Flow-Based Route Planner (FBRP) is an algorithmic system designed to route flows of aircraft between hazardous, time-varying weather systems to get between designated start and finish points. If a shortest path is defined by time, and must remain open for a specified amount of time, with constraints that change with time, then how can we route it? If we need to route many such paths, how can we resolve conflicts among them to allow each path to remain short while routing as many paths as possible?

While the general form of the problem is NP-complete, we can constrain the paths (and the associated searches) in specific ways to get good solutions with a reasonable running time that are highly practical for the application domain we are examining.

2.1 Overview

The FBRP solves a set of routing requests given a set of static constraints and a dynamic weather forecast specified by a sequence of time slices of weather. The routes are defined in such a way that any aircraft that fall into a particular routing request during a particular time window will follow a specified set of routes that are valid during that time window only. Each route, specified with a time window of validity, is referred to as a *flow*. The routes are dynamic – each specified time window has its own set of routes. The weather is also dynamic – the algorithm routes aircraft in such a way that the aircraft will avoid predicted weather that will occur in the course of its flight.

The computed routes obey several important constraints: they avoid approaching within some specified distance of hazardous weather; they obey a horizontal separation standard with respect to aircraft on other routes (flows); they keep within some specified range of allowed headings; and they have low complexity (defined as the number of waypoints on the route). Within the specified constraints, the routes are computed to be optimal with respect to distance among all routes that lie on the search network, according to an objective function (e.g. the total time of flight).

Each route is computed in two phases. First, we calculate an optimal route from start to goal (e.g., in the transition airspace, from a point on the 200 nm range ring to a metering fix), using a shortest-path algorithm on a densely connected search grid. Second, we simplify the computed route by calculating the route of least complexity that is sufficiently close to the original route.

Only those links within the search network that obey the constraints are considered to be usable. Feasibility of a link requires checking it against the weather forecast data as well as against other flow routes that have already been specified, in order to ensure weather separation standards and lateral separation standards.

2.2 Problem space

The FBRP works in an XYT world, with two dimensions of space and one of time. The area of interest *A* is defined to be some region of space, normally a polygon (for sectors of the airspace) or a circle (for the transitional airspace around airports). Within the area of interest are a set of constraints *C*. The constraints can be represented in several ways, for example as pixel maps of obstructed areas that are valid for particular time intervals, or as polytopes in XYT space. The XY space is not necessarily Euclidean; in the application of the algorithm, it is necessary to use the spherical geometry of the Earth itself. In practice, distances are measured as great-circle distances throughout the algorithm.

Aircraft within the area of interest *A* are modeled as single points, which on the scale of a single sector is approximately true: aircraft are typically less than 300 feet long. Aircraft must maintain a specific amount of separation between themselves and other aircraft, in order to reduce the chances of a mid-air collision to as close to zero as possible. This constraint is the horizontal separation standard δ.

In the case of a series of aircraft, all using the same route, and minimally separated, the aircraft are modeled as a series of contiguous segments of that route rather than a number of points. The importance of this is that it reduces the geometric complexity of the problem while being essentially the same model. This is discussed further below.

Occasionally there are regions of restricted airspace. A restricted airspace can be reserved for military aircraft or for protective reasons (i.e. no civilian aircraft may fly within a certain region around the White House). The restriction can also be for traffic management; for example, the area within fifty miles of an airport is normally divided up into arriving-flight regions and departing-flight regions; the one cannot go into regions reserved for the other. Normally an airspace restriction is a long-term designation; as such, it is applicable throughout the entire time span that the algorithm operates in.

The transitional airspace of an airport is the 200-nautical-mile-radius region directly around it. Aircraft that are landing at, or departing from, an airport are handled separately from aircraft that are passing by an airport within this region. If the Flow-Based Route Planner is being used to plot arrival and departure routes for the transitional

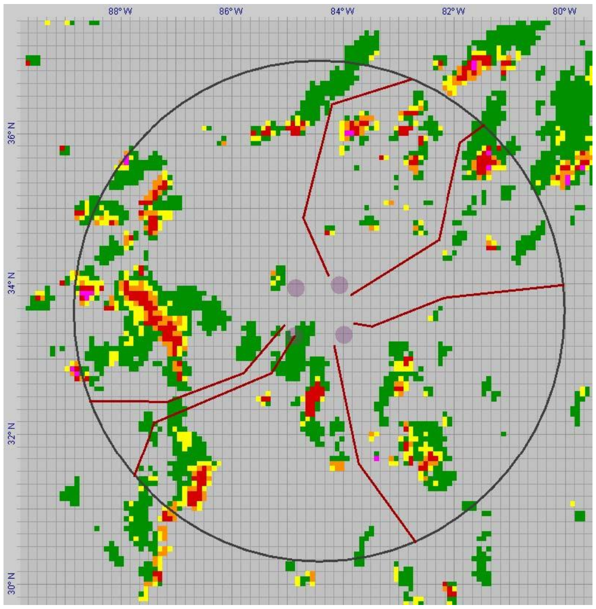


Figure 3. A sample search space with solution routes.

airspace of an airport, then there are a few other elements involved in the problem. The range ring *R* is the 200-nmi-radius circle that defines the boundaries of the transitional airspace. Every airport has a set of arrival fixes MF_{ai} and departure fixes MF_{di}. These are points in XY, through which all aircraft pass on the way to or from the runways. All arrival routes in transitional airspace begin on *R* and end at one of the arrival metering fixes MF_{ai} (or a point very close to one). All departure routes in transitional airspace begin at one of the departure metering fixes MF_{di} (or a point very close to one), and end on *R*. (Routing aircraft from the arrival metering fixes to the runways, or from the runways to the departure metering fixes, is beyond the scope of this document.) As discussed above, the 50-mile-radius region directly around an airport is divided up into arrival-exclusive and departure-exclusive zones. Normally there will be one such exclusive zone per metering fix, and every point of the inner 50-mile-radius region will be in one of those exclusive zones.

Since aircraft are landing and taking off at airports, it might reasonably be asked whether this converts the problem from a problem in XYT to a problem in XYZT space. In order to keep the problem simple and the solution feasible, we take advantage of the altitude profiles of arriving and departing flights, which are consistent enough to allow us to treat the transitional airspace as if it were flat; even if we were not to treat it as flat, the aircraft would still be required to be at similar enough altitudes to each other that horizontal separation would still be necessary at all points.

The area of interest normally contains hazardous weather formations, which aircraft are required to avoid. We normally represent weather data as image maps. A large grid of weather data is assembled from terrestrial or satellite observations, which are not necessarily on an even grid, or is the output of a weather prediction algorithm. The standard weather severity measure is the reflectivity index. For purposes of the project, reflectivity values of 35 or greater are considered to be too hazardous to travel through; therefore the weather map is reduced to passable and impassable regions. Weather measurements and predictions are done at specified intervals (normally every five minutes) rather than in a continuous data representation.

2.3 Routes and Flows

A route, for purposes of this document, is the path that a single flight takes, or is intended to take, from start to destination. A route π is composed of a series of points in space-time w_i. The route π flows from w_0 to w_1 to w_2, etc. until it reaches its destination at w_n. For routes generated by the route planners described in this document, each of these waypoints is a point in XYT space, not just a position, so that each segment of the route has an implicit speed (which is generally a constant, *v*).

A flight is a route that is valid for a single aircraft. At any given point in time, a flight is a single point. Likewise, at any given XY point, there is a single point in T that the flight crosses. That single point (in time or space) is the space-time location of the aircraft taking the designated route.

A flow *F* is a route that is valid for a particular time interval $[t_F^s, t_F^e]$. At any given point in time, a flow is a contiguous set of points, a series of line segments outlining where aircraft may legally be within the flow at that time. Likewise, at any given XY point, a flow is an interval, and at any point in that interval, an aircraft may legally be present, and expect to be able to follow the flow to its final destination unimpeded by hazardous weather or aircraft outside of the flow. The segments of a flow, seen in XYT space, are parallelograms perpendicular to the XY plane; each aircraft within the flow traces a flight within the confines of the flow.

2.4 Problem statement

The flow-based routing problem is in two layers. One, how can we solve the problem with regards to a single flow? Two, how can we solve the problem with regards to multiple, non-intersecting flows?

Each instance of a single-flow FBRP problem is defined in terms of the origin o_F, destination d_F, speed v, a desired time interval $[t_F^s, t_F^e]$ for the route, and the airspace A. (The definition of the airspace includes the value of the horizontal separation standard δ.) The output is a well-formed route π_F that connects o_F to d_F, such that any route starting at o_F in the time interval $[t_F^s, t_F^e]$ and going at the specified speed v, is guaranteed safe passage to d_F. (Alternatively, aircraft could also be guaranteed that, should they start at a time interval determined during the search process, they will arrive at d_F in the time interval $[t_F^s, t_F^e]$.)

In the kind of environment where flow-based routes among moving weather is desirable, it is also desirable to be able to maintain such routes for long periods of time, and between multiple point pairs. However, because weather formations move, there is a limit to how long a single route can still remain open, since hazardous weather may eventually cross the route and block it. This gives us the multiple-flow FBRP problem. An instance of the multiple-flow FBRP problem consists of a large number of flows $\boldsymbol{F}$ and the airspace A. For purposes of implementation, it is assumed that all starting time intervals are at regular intervals and do not overlap; normally the implementation itself only accepts beginning and ending times and the length of the interval, and determines the time interval of each flow itself.

2.5 Constraints

Each constraint $C_i(t)$ has a required distance ε_i that all routes must maintain from it. This distance is not the same for all constraints – although all route constraints currently must share the same value – and for many constraints it is normally zero. We define XY-parallel distance as follows: for the segment H, the given constraint $C_i(t)$, and a given time t, find the closest distance of approach between H and $C_i(t)$ within the plane $T=t$. Minimize over all possible choices of t to find the XY-parallel distance between H and $C_i(t)$. If the XY-parallel distance from a segment to a constraint is less than the required separation from that constraint, then the segment is not legal. A legal route consists of a series of legal segments, each of which shares an endpoint with its predecessor and successor.

2.5.1 Hazardous Weather

Aircraft are not permitted to fly through, or usually even too close, to hazardous weather, except under special circumstances – for example, isolated hazardous weather cells that do not represent genuine obstacles.

Hazardous weather is represented as a series of grids of reflectivity measurements or predictions. (The weather data grid is independent of the search space grid discussed later; each grid square is normally two nmi across, although for experiments across the national airspace we used a grid size of eight nmi.) Each grid point of weather data represents the measured or predicted weather, within the area of interest, at one particular point in time. Thus, the entirety of the weather is represented as a series of such grids more-or-less evenly distributed across the time of interest; the i^{th} weather sample is at time T_i.

Since weather is normally predicted at particular intervals rather than continuously, each hazardous-weather image map covers a specified time slice, during which any hazardous region in that image map is considered impassable. It is possible to interpolate between weather samples to produce a continuous representation of weather, but this is a computationally costly process and requires somewhat arbitrary assumptions about how to interpolate between time slices. The algorithm takes the conservative stance that a weather prediction as valid between the previous and next predictions; thus, if predictions are spaced at five-minute intervals, a weather prediction will be considered valid for five minutes before and after the actual time stamp for that particular prediction. (The first prediction is considered valid at all times before it; the last prediction is considered valid at all times after it.)

In order to determine if a particular segment of a flow is legal according to the weather, several steps are required. First, in order to implement the weather-avoidance-distance constraint, the weather image data is convolved: a copy of the weather image is made at double resolution, and a convolution is done with a circular kernel of radius equal to the weather epsilon value. The convolution is not a conventional convolution, but instead a maximum-value one; the convolved value of a particular cell is equal to the

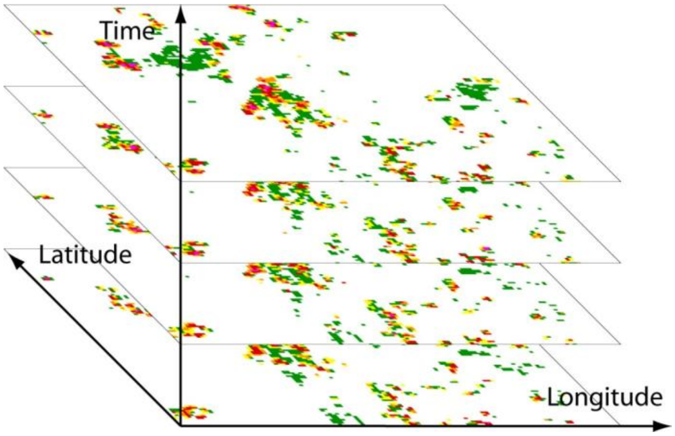


Figure 4. Weather is represented as a series of samples in time using sampled data (historic or predictive) from the National Weather Service.

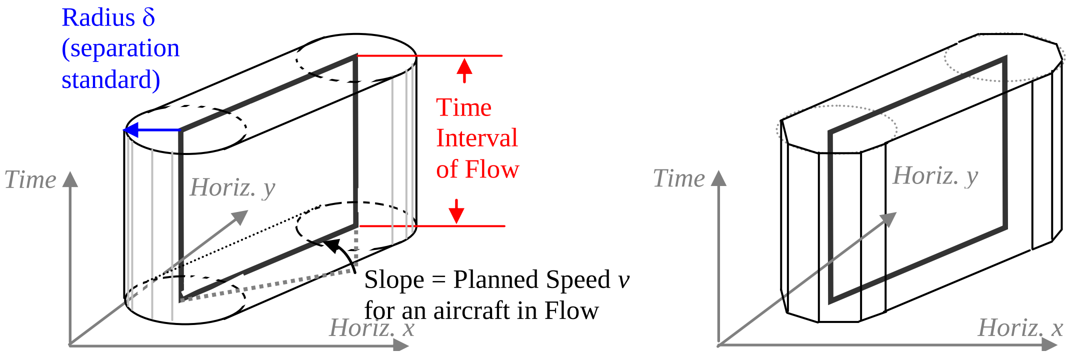

Radius δ (separation standard)
Time Interval of Flow
Time
Horiz. y
Horiz. x
Slope = Planned Speed v for an aircraft in Flow
Time
Horiz. y
Horiz. x

Figure 5. The constraint region of space-time corresponding to a segment for aircraft in a flow where no aircraft protected airspace zone is allowed to cross over the bounding region.

largest value of all weather cells that are within the shape of the kernel around it. Taken in the continuous domain, this is logically equivalent to a wild-fire algorithm.

Once this operation is done, the algorithm can simply check if the line-segment part of a route segment intersects any hazardous weather regions. The algorithm trims the segment to that portion which will intersect a particular weather time slice, and then does a scan-conversion of the segment to find out what weather pixels it will cross. Crossing any weather pixel that registers as hazardous weather immediately means that the segment is illegal. The actual implementation is flexible on this point; the algorithm could conceivably be modified to provide a cost function more complex than the simple step function used, with no change to the theoretical running time.

Since the epsilon value for weather avoidance is constant for a given scenario, the weather convolution can be calculated and stored ahead of time for each time slice of weather data. In practical application, the original and convolved weather data are the most relevant cause of the large memory requirements of the algorithm; see below.

2.5.2 Existing Routes

In the multiple-flow FBRP problem, flows are not permitted to approach more closely than the horizontal separation standard, δ. Because an individual flow is, at any point in time, a segment rather than a single point, this constraint is more complicated than it appears at first glance. Two routes conflict under the same circumstances that a route conflicts with a constraint; if the two routes, at any point in time, approach within δ miles of each other, then the two routes conflict and are not legal.

The process of route-route intersection checking is somewhat difficult in the flow-based case. As above, two routes are in conflict if their closest approach, as measured within the XY axis, is less than a specified constant (typically this value is five miles). Any pair of segments from two flows (which must necessarily be separated by at least five miles) is effectively a pair of parallelograms in XYT space. The XY-distance question becomes much more difficult to solve because a pair of points is only in violation if they are in the same Z plane. Any known approach to calculating the XY distance between two parallelograms is costly, complicated, and difficult, so a different approach is taken, based on three-dimensional K-dop polytope intersection checking.

Each parallelogram-segment of an existing route can be represented (for route conflict checking) by an extruded cylinder, as illustrated below – in practice we approximate the extruded cylinder with a polytope. These polytopes become the input to a polygon collision detection algorithm based on bounding-volume hierarchies of *k*-dops. A generalized bounding-volume hierarchy would not be sufficiently fast, so domain-specific timing information is incorporated, when possible, to eliminate virtually all prior routes from consideration; when this is done, it can be done because a great deal is known about the exact shape that a route must take in space-time.

The original implementation was set up such that the bounding volume hierarchy was a standard binary tree. However, because of high spatial locality within the triangles that make up a particular segment of a route, and even within a route itself, this is far from an optimal method; the results of a *k*-dop query at any given node of the tree correlate far too strongly with those of its parent and children. Fortunately the *k*-dop concept is easily used even in a more general tree-like structure. It was discovered that queries were far faster when *k*-dops were used at two discrete layers: first, the route itself; second, each segment of the route. The *k*-dop bounding volume is sufficiently tight that two extruded-cylinder routes, such that their *k*-dops intersect, are very likely to intersect as well.

At the level of segment-segment checks, one segment – the one to be conflict-checked – is calculated as a normal parallelogram (i.e. an extruded cylinder of radius zero). The other segment – from the existing route – is precalculated and stored as an extruded cylinder of double the conflict radius. Intersection-checking these two is logically equivalent to checking the original regions (by analogy with Minkowski sums) as long as a single point from the parallelogram is checked to see if it is inside of the double-size extruded cylinder. If one of these two checks fails, then the two route segments are in conflict, and the new candidate is rejected.

See [KHMSZ] for a thorough treatment of *k*-dops, which are generalizations of axis-aligned bounding boxes to convex polytopes having *k* discrete orientation facets.

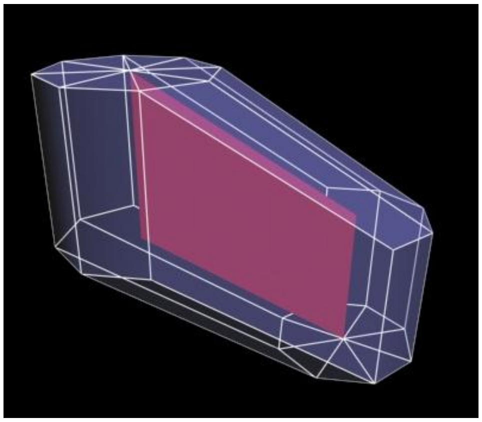


Figure 6. The polygonal approximation of the extruded cylinder used for route conflict checks.

The red parallelogram is the region of space-time that corresponds to a segment for an aircraft along a flow, where the "height" of the parallelogram indicates the time interval for the flow, and its slope indicates the speed of the aircraft. The transparent blue polyhedron is the region that must be kept clear of hazardous weather and other aircraft (or more precisely, other aircraft's protected airspace). This illustration is only for one segment of a route; the entire route is represented by a chain of such polyhedra.

2.5.3 Polygonal Weather

Normally weather is represented by image maps, because that is the default storage method for hazardous weather in the application domain. However, it is just as reasonable to assume that hazardous weather is stored as polygonal regions, either time-sliced (as with image map weather) or prescribed directions of movement, or as polytopes in XYT space.

In order to handle weather of this type, it is simple and easy to convert it into polytopes in XYT (if it is not already), and then treat it similarly to route-route intersection. The algorithms for route-route intersection do not make any assumptions other than that the routes are represented by polytopes, so this is a viable strategy. For polygonal weather that is mostly static in time – that is, the polygon shape of the weather is a constraint for some specific time period, before and after which it does not exist – it is also possible to trim individual segments to that part which exists in the relevant timeframe of the polygonal weather, and then directly do distance checks between the segment and each of the polygon's edges. In practice this is the method used.

2.5.4 Range Ring, Sector Boundaries, No-Fly Zones and Other Static Constraints

The range ring and other sector boundaries, no-fly zones, existing reserved spaces for static routes, and other static constraints are easy to check for conflicts. Each of these types can be represented by polygons in XY space. From there it is easy to determine whether a segment crosses into an illegal region of space exactly as though the constraint were polygonal weather.

2.5.5 Elliptical Constraint

Some scenarios require the use of a routing boundary in the shape of an ellipse – for example, when generating coded departure routes between airports across the country, there are no pre-existing boundaries other than the borders of the United States. In general an ellipse is a close approximation of the region that is considered "useful" for rerouting aircraft. The elliptical constraint is also easy to specify and easy to check for during the route search.

2.5.6 Orientation

For purposes of the ATC application, it is important that aircraft remain within Θ degrees of the direction between origin and destination; Θ must be less than 90°. This constraint enforces monotonicity on the solutions, which is valuable in many applied versions of this problem. While in theory the shortest route could involve doubling back, in practice such routes are not desirable in real applications, and including their possibility is extremely costly in both time and space of solution.

2.5.7 Route Complexity

It is important that routes be simple enough to be quick to both transmit and record by air traffic controllers and pilots with a limited budget of attention. The refinement stage implements this by solving for the route with the fewest segments, rather than the shortest route. Additionally, we forcibly cap the number of segments in a route at a constant; no route with too many segments is accepted as a solution.

2.6 The single-route Flow-Based Route Planner algorithm

In this section, we examine the technical details of the shortest-path search that finds one individual route through dynamic obstacles (including pre-existing routes).

The single-route problem is solved through a variant of *A** search [Ni]. The *A** search has significant advantages over other graph search techniques (e.g., Dijkstra's algorithm), especially in situations, such as ours, in which an effective lower bound on the remaining path length is easily determined. The search is guaranteed to terminate with an optimal path (just as breadth-first methods, such as Dijkstra's algorithm), but the search is goal-directed, exploring the most promising extensions to the path first. In theory, *A** may explore just as many nodes of the search network as Dijkstra's algorithm, but in practice, the *A** search is often substantially more efficient.

The algorithm searches on an implicitly defined grid of points and connections between those points, in order to provide a first-order approximation of the shortest route. The algorithm then refines and simplifies that route, using the first-order approximation and specifically-placed Steiner points around that approximation as the graph for a second search. The resulting routes are both simpler and usually shorter than the originals.

2.6.1 General properties of the search graphs

Each individual vertex w_i of a search graph $G=(\mathbf{W},E)$ is a point in XY space, with an associated T value $w_{i,t}$. However, different T values are not considered to distinguish points; we can take this liberty because the application does not permit the route to cross the same point twice. So as the search algorithm examines branches of the search tree, the only version of the point that is stored in the search tree is the lowest-cost version; the time value recorded in that point is the earliest known time to get to that position in XY space. Thus the only way to get to any w_i in XY space is by the shortest non-crossing route to w_i, and the algorithm will only examine routes that get to each individual point by the shortest possible route, even if the optimal solution would require being at that point later in time. This is an important factor in the behavior of the algorithm.

Each edge E of G either exists or does not exist based on the legality of the associated possible segment, as one would expect. The edge costs are proportional to the great-circle distance between the respective points, as per the spherical geometry of the problem space. Since all graphs are defined implicitly, edges that should not exist are simply given cost values of infinity.

Since the graph is implicitly defined, and each edge can change in legality based on what the start time of the edge is, it is important to note that the only legal edge that exists is the earliest one that can be constructed from its start node. If w_i can be reached earliest at time t, then the only edges in the graph originating at w_i are those that start at t. If this were not the case, then the size of the graph would increase according to the number of different times that a particular node could be reached, making it computationally infeasible to actually search the graph. Since the original application already considers routes that take less than the shortest route to a particular point to be bad routes, this is not a significant limitation. Most other applications would most likely want to carefully constrain the size of the search graph in other ways so as to ensure that the graph can actually be successfully searched.

It is currently an open question as to what kind of point distribution is best for the searches conducted by the algorithm. For the initial search an augmented grid is used, and for the refinement search a specialized set of Steiner points is used.

2.6.2 The Start and Goal

Depending on the requirements of the routes in question, there can either be a single legal endpoint on the range ring, or several in a small section of the range ring, or points scattered evenly across the entire range ring. The choice depends on the exact nature of the problem to be solved; for example, if a route from some general direction is desired, because a large number of aircraft typically come from that region, then it may be reasonable to find a route from any point in that region.

It is similarly possible to have multiple possible endpoints at metering fixes. Since hazardous weather at the metering fix is much more likely to be an obstacle than hazardous weather in the general airspace, it is heavily advantageous to allow aircraft to arrive at or depart from more than one point in the vicinity of the relevant metering fix. These points are at similar distances from the airport and close to the original metering fix. In this way the runways associated with that metering fix can remain open and useable.

For similar reasons, in en route routing, we can also provide multiple start and goal points for a route. An aircraft (or series of aircraft) may reasonably be expected to divert slightly from their original flight plan in the interest of avoiding hazardous weather, even at the points of entry and exit of a zone.

In implementation terms, multiple entry and exit points are handled by augmenting the graph to have a single start and goal node, and multiple connections from the start node to the possible entry nodes, and from the goal node to the possible exit nodes. This is a standard method for dealing with multiple start and goal nodes in search problems that reduces the problem back down to having a single start and goal node.

2.6.3 The Grid Search

The grid search finds an initial approximation to a shortest-path between o_F and d_F. The route must satisfy all of the specified constraints: avoiding hazardous weather by at least the safety clearance ε, maintaining the horizontal separation standard with respect to already established flow routes, obeying the turn and orientation constraints, and so on.

The grid search graph uses a specified-size fine grid Γ to cover the problem space; typically Γ is 128 or 256 units square; for typical applications this results in grid squares spaced one to three miles apart. The grid is connected by connecting each vertex on the grid to every vertex in its immediate neighborhood. “Immediate neighborhood” is defined as any other vertex whose position in the graph is no more than K units away in both X and Y; K is the connectivity constant of the graph. If two edges go in the same

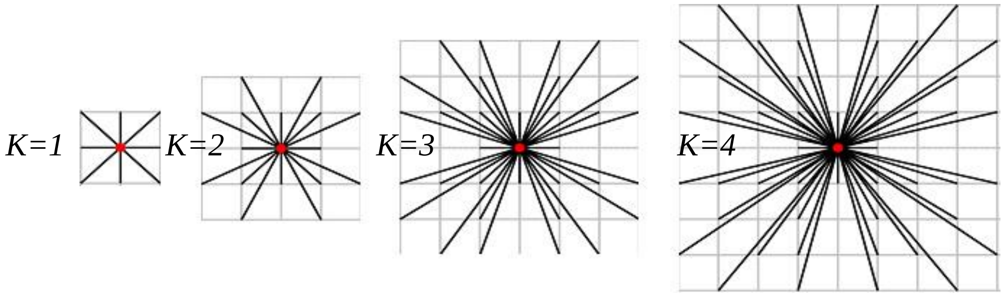


Figure 7. Connections from a central point to its neighbors, for connectivity constant *K*=1, 2, 3, 4.

Higher values of K produce a larger number of edges linking the point to its neighbors at a wider variety of orientations. This has the benefit of improving the approximation of the Euclidean metric, at the cost of increasing the search complexity, as the network size increases.

direction, the longer of the two is discarded. Formally speaking, vertex $\Gamma_{x,y}$ connects to $\Gamma_{x+i,y+j}$ if *i* and *j* are relatively prime. For example, if $K=4$, grid point (a, b) connects to grid points $(a + 1, b + 1)$, $(a + 1, b + 2)$, and $(a + 3, b + 4)$, but not to $(a + 2, b + 2)$, $(a + 3, b + 3)$, $(a + 4, b + 4)$, or $(a + 2, b + 4)$.

The effect of this is that every possible direction that could be taken by an edge, without going outside of the immediate neighborhood, is covered exactly once by an outbound edge from each node.

The start and goal vertices are connected to the grid via extra edges on the graph; the start vertex is connected to its nearest neighbors in the graph within a small distance, while all vertices connect directly to the goal vertex. While this doesn't improve theoretical performance, the majority of searches will skip a large number of nodes because the search will connect directly from the last node on the route that sees the goal node, directly to the goal node.

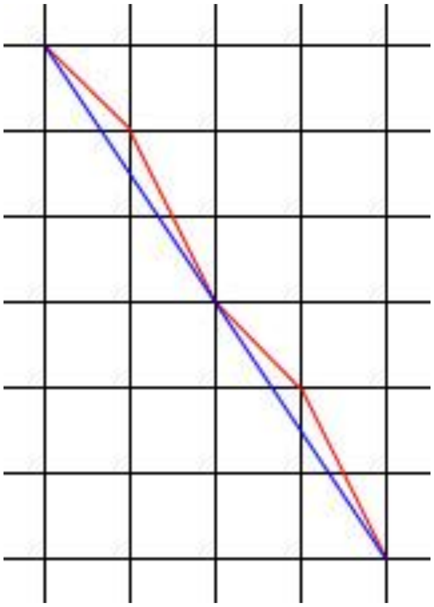


Figure 8. The shortcomings of grid-based search for short routes.

Regardless of the choice of the connectivity constant, the grid-based route will almost always be longer and more complex than a straight-line route would be.

In order to actually find an optimal route, we search on this grid. The grid is implicitly defined: that is, the grid structure is not stored, but instead is algorithmically generated as we need it, including all vertices, all segments, and the cost of all segments. The full grid would take an enormous amount of time and space to calculate, as the properties of the grid at point w_i depend on exactly when an aircraft can reach w_i, and a full grid calculation would require keeping records for all possible times that an aircraft could reach w_i. Additionally, since any particular grid edge is only traversed

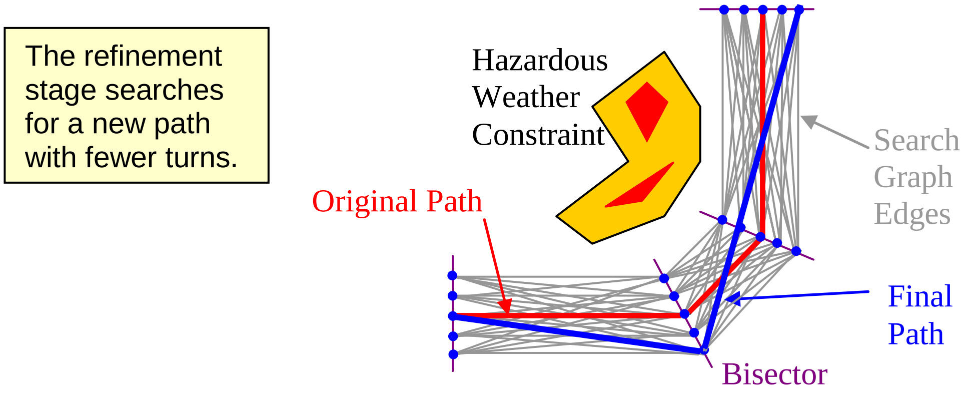

The refinement stage searches for a new path with fewer turns.
Hazardous Weather Constraint
Original Path
Search Graph Edges
Final Path
Bisector

Figure 9. The second phase of the FBRP.

Candidate turn points are placed along the angular bisector of the route π_F, and a new solution route within a tolerance distance of τ from π_F.

once, there is no particular benefit to constructing an edge before we need it, or storing the resultant edges, except when they are stored in the search tree.

2.6.4 The Refinement Search

The output of the grid search is a route π, with waypoints (w_0, w_1, … w_n). Typically, π is a good approximation to the actual shortest route, but it is also of far higher complexity (i.e., having far too many waypoints) than is permissible in ATC applications. It is also longer than the true shortest route, because of the limited number of angles of travel that the connectivity constant allows. The refinement search post-processes π using another shortest path search, but the route is optimized in terms of its complexity. The refined route is π_F, the solution route.

In order to improve the quality of the route, it is used as the input to a new search. The refinement search performs a search for a path over the complete graph of a specifically defined set of Steiner points (candidate waypoints) S. We define a set S of $|\pi|*(2M + 1)$ points as follows, where $|\pi|$ denotes the number of waypoints in route π:

1. Each waypoint w_i is in S.

2. We place M evenly spaced points on each side of w_i, spaced out at the tolerance distance of τ between each, along the angular bisector of the route π at point w_i.

The refinement graph is complete with respect to legality; all vertices are connected to all other vertices so long as the resulting edges are legal. The (legal) edges of this graph cost unity, so that the “shortest” route is the one with the fewest edges. It is not guaranteed that the resulting route will be shorter; however, it is provably limited by the length of the original route, using triangle inequality.

There is one constraint that is only present in the refinement search: edges that are shorter than some specified length L are illegal.

2.7 The multiple-route Flow Based Route Planner algorithm

The application domain of the FBRP is aircraft flows during hazardous weather situations. There are multiple source points, multiple destinations, and aircraft continue to fly and need routes indefinitely. This introduces two new elements to the FBRP: route-to-route conflict avoidance and multiple routing. Route-to-route conflict avoidance requires that any segment be tested for legality against any segments whose positions are already determined; this will be discussed below.

It is fairly easy to prove NP-hardness for the problem of finding two non-intersecting paths that are separated by epsilon distance through a field of N constraints, such that the sum of the path lengths is minimized; this is true even in two dimensions. This result is directly applicable to the multiple-route FBRP problem. Instead of directly solving the hard problem of how to optimize two routes at once, the algorithm uses a relatively simple approach. Several random orders of solution are taken for any given set of routes to be solved simultaneously, and the best set of solutions found is taken. This is not done for all scenarios, however, since some have far too many routes to choose among several random orders; in this case only one random order is used.

Multiple routes are handled by placing flows at fixed intervals in time according to their duration: if all flows are X minutes in duration, and the beginning time for the problem is S, then flows are attempted for times (S, S + X, S + 2X, S + 3X…) for each origin-destination point pair. Each of these sets of flows is then routed from earliest flow to latest, so that earlier flows get priority over later ones. While this doesn't provide any guarantees of optimality, it is highly practical and efficient for most scenarios because the orientation constraint can be, and is, structured to guarantee that no flow is able to take more than twice the optimal time to reach the destination. In this way we can guarantee that a flow never blocks more than one of its own successors.

2.8 The Parameters

The FBRP allows for fine-grained control over many aspects of the desired problem and the path-searching process.

```
For each time window do {
   For each of N random orderings of the routes to be solved do {
      For each route, in order, do {
         Search for a legal route $R_1$ from U to V of minimum length
         Generate waypoints at each node of R
         Search for a nearby route $R_2$ minimizing the number of waypoints
         If route $R_2$ uses at most k waypoints {
            Report it
         } Else {
            Continue searching
         }
      }
      Compare route set S to $S_O$, the optimal set found so far
      If S has more successful routings than $S_O$, then $S \rightarrow S_O$
   }
   Keep set $S_O$
}
```

Figure 10. Pseudocode illustrating the solution process.

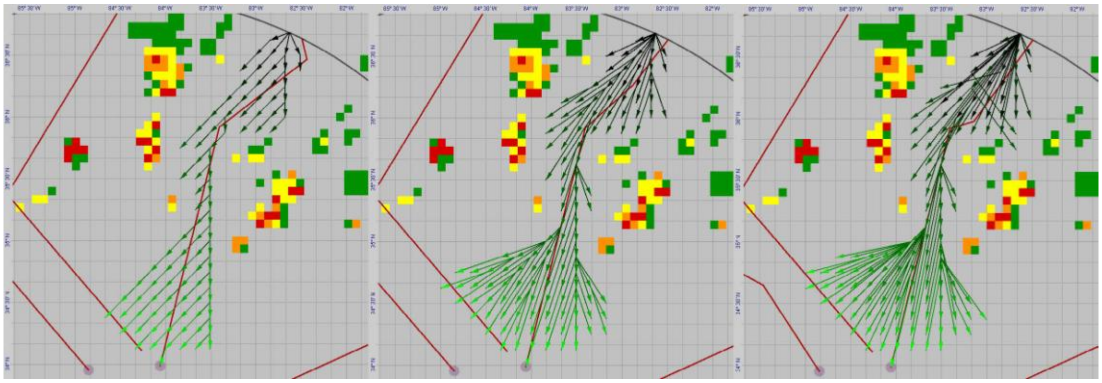


Figure 11. The effects of the connectivity constant on graph searches.

From left to right, searches with connectivity constant 1, 3, and 5. All three searches cover approximately the same region of airspace, but a higher connectivity search looks at a wider range of possible angles. We have found that a connectivity constant of 3 is sufficient to produce good results, so this is the default choice of this

2.8.1 Weather margin of safety

The algorithm is completely independent of how conflicts with hazardous weather are defined or determined, as long as the definition is based on a binary rule ("go" or "no-go"). It is also possible to introduce a cost rule that is based on an increased cost for hazardous weather, although this has not been implemented.

2.8.2 Aircraft margin of safety

The algorithm will accept any number as the separation standard between aircraft. The algorithm currently requires that all aircraft have the same safety margin parameters, but this is already the case in standard ATC procedure and systems. There is no particular algorithmic need to maintain the same separation standards for all pairs of aircraft, however, since each segment of each route is examined independently.

2.8.3 The density of the search grid

Doubling the resolution of the search grid increases the running time, of course, but it also improves the quality of the output routes, to a point (beyond which no significant improvement is possible, and running times become prohibitive.)

2.8.4 The connectivity constant

Increasing the connectivity constant K increases the number of connections from one node to another in the grid. This is another tradeoff of quality versus running time, although experimentally the best value is $K = 3$.

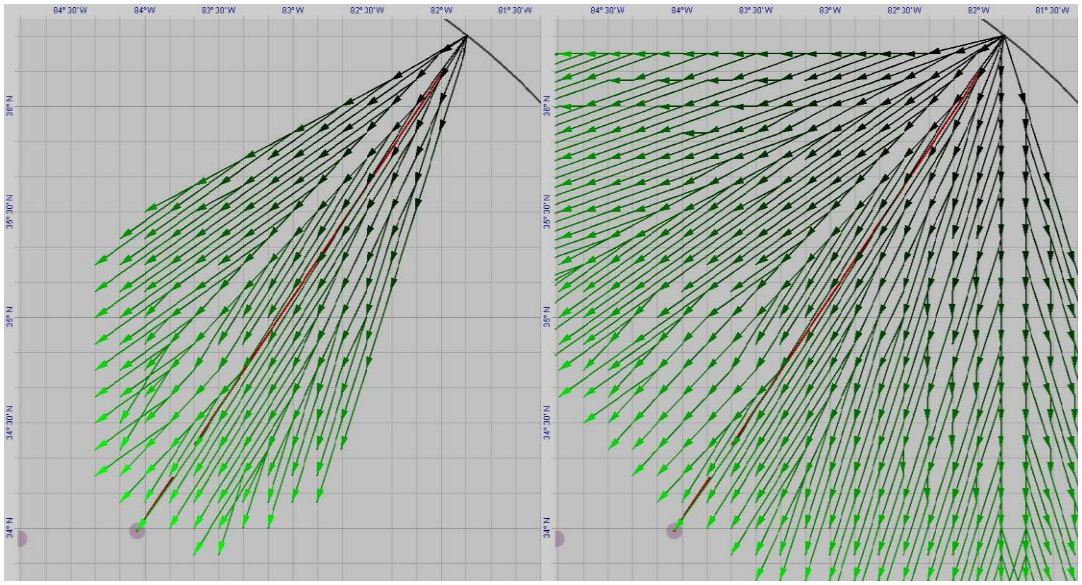


Figure 12. The effects of the angle constraint on graph searches.

On the left, we see the search as it is conducted with an angle constraint of *Θ*=30°. The black and green arrows represent the segments that were examined over the course of the search; the brightness of the segment represents when it was examined in the search (i.e., distance from the starting waypoint). On the right, we see the same search except with an angle constraint of *Θ*=60°. A much larger area is searched, which takes more time but also allows more routes to be found. These search trees were generated with breadth-first searches, for clearer illustration of the angle constraint feature; an *A search would have immediately headed for the**

2.8.5 The angle constraint

Any degree value for *Θ* in the range [0, 90) is acceptable – routes must remain monotonic with respect to *some* direction in order for the algorithm to work correctly, because of the inherent assumption that a route will never cross itself. Strengthening the angle constraint may also require increasing the connectivity constant, in order to increase the number of discrete headings available to aircraft in the first search stage.

2.8.6 The length constraint

Output segments can be constrained to be always above some specified minimum length, *L*, simply by removing all such segments from the complete graph of the second search phase.

2.8.7 The complexity constraint

The complexity constraint is effectively an upper bound on the number of waypoints. We have designed an option into the algorithm that will constrain the number of waypoints to be at most a specified maximum, *k*, based on the dynamic programming (Bellman-Ford) techniques of [KLM]; however, this option is still under implementation. The currently implemented algorithm can be used to impose a bound on the number of waypoints by searching (Phase 2) for a minimum-waypoint route within a specified tolerance distance of the route computed in Phase 1. (Theoretically, this method cannot guarantee it will find a *k*-turn route when one exists, but in practice it is highly effective at minimizing the number of waypoints.)

2.9 First-Come First-Served Free-Flight

While the FBRP is normally applied, and was first designed, for flow-based routing of aircraft, it can be applied to free-flight scenarios as well.

The First-Come First-Served (FCFS) Free Flight scenario is defined as follows. A series of aircraft will arrive on the range ring, one by one, with specified intended metering fixes as their destinations; as they arrive, route them to the desired metering fix, obeying the constraints.

The Free Flight algorithm uses a search in space-time (as in the Space-Time Flow method of [CKLM]), treating already routed flights as constraints to be avoided (according to the minimum separation standards between aircraft), along with severe weather constraints, no-fly zones, and quadrant constraints (confining an aircraft to the quadrant initially entered).

Free Flight routing allows maximum flexibility in terms of individual aircraft routing; however, by allowing each flight to be greedily routed, the Free Flight solution can result in some routes being particularly efficient, at the expense of blocking off a passage for other flights. Furthermore, there is no attempt in the Free Flight algorithm to sequence aircraft optimally.

It is relatively simple to apply the FBRP to the free-flight case. We can easily treat each individual flight as a single flow of duration zero. In order to simulate free-flight properly, each flow representing a flight has an origin point on the range ring that represents its desired position and time of entry, and a destination at the metering fix. Once this set of flows has been generated, we can solve them individually in any particular ordering. There are, however, a few specific modifications that must be made.

2.9.1 Solving Order

There is no practical way to solve any sizeable number of permutations of the flows, because each flow is a single flight, and there are hundreds of flights each hour. However, the free-flight approach itself assumes autonomy of the pilots; any globally-minded solution would itself violate the terms of the approach. We use the simple method of solving earlier routes first, for several reasons – one, they are more likely to have priority, and two, sweeping forward in time allows us to make other needed optimizations.

2.9.2 Route Polytopes

Since each segment of a route has zero duration, we can eliminate a lot of triangles from our route-intersection data structures. Normally each segment requires roughly 52 triangles. However, if a route has zero duration, then many of these are unnecessary and the shape of the segment collapses to a skewed cylinder, requiring only 32 triangles to represent.

2.9.3 Sweep Elimination of Old Route Polytopes

On average, twelve hours of flow-based route planning for the transitional airspace will include roughly 192 routes, each of which has on average four segments;

the associated approximating polytopes contain roughly 52 triangles, for a total of 39,936 triangles in the K-dop tree structure. Twelve hours of free-flight planning, on the other hand, will include roughly 3600 routes, each of which also has on average four segments; these segments, having zero duration, only require 32 triangles each; at the end of the simulation roughly 460,800 triangles will be stored in memory. While the K-dop structure allows us to check only a tiny fraction of these triangles, the intersection time becomes significant when every single segment that the search examines needs to be intersection-checked.

In order to reduce this time, a sweep cone (centered at the airport) is used to eliminate old routes. Because of the angular constraints and the fact that the free-flight aircraft are routed in order of time, we know that any route that is a specified distance behind the current aircraft's start time cannot possibly interfere with the current route. The distance is determined by the angular constraint, which provides an upper bound on the length of any route. Before each free-flight aircraft is actually routed, we check the set of existing triangle intersection structures. Any that corresponds with a route that cannot possibly interfere with the current route is discarded outright.

Under this modification, a typical routing operation must intersection-check potential segments against only fifteen other routes, rather than hundreds or thousands. Since intersection checking is the majority of the operational cost of the algorithm, this greatly speeds up free-flight route planning.

2.10 Free-Flight Required Time of Arrival

The standard multiple-route Flow Based Route Planner works on the assumption that we are trying to guarantee that the set of flows that are attempted covers all possible start times for the aircraft that are traveling in the flows. However, in the case of airport landings, we might want to work the opposite way, so that every possible time of arrival at the landing metering fix is covered by some flow. The Required Time of Arrival (RTA) concept is that, in order to maximize usage of the metering fixes and runways, flights should be scheduled to arrive at metering fixes at particular times, and must route with that assumption in mind. In the application domain, when routing flights under the RTA assumption, it may or may not be important to give individual flights a particular desired entry point at the range ring.

In order to accommodate such circumstances, we can make a few simple modifications to the FCFS Free Flight algorithm. The algorithm itself remains totally unchanged; however, all flows are routed, in last-to-first order, from the metering fix, backwards in time, to the range ring. This allows us to fix the aircraft in time at the metering fix, and discover the appropriate time for the aircraft to enter the transitional airspace around the airport.

According to the demands of the situation, it is simple to modify the search graph so that range ring vertices are weighted in one of three ways: one vertex can be the goal point of the search, a number of them can be a goal point but weighted to favor a particular region of the range ring, or any range ring vertex can be a goal point. The latter is particularly useful for empirically calculating the maximum throughput capacity of the transitional airspace with significant hazardous weather formations, which

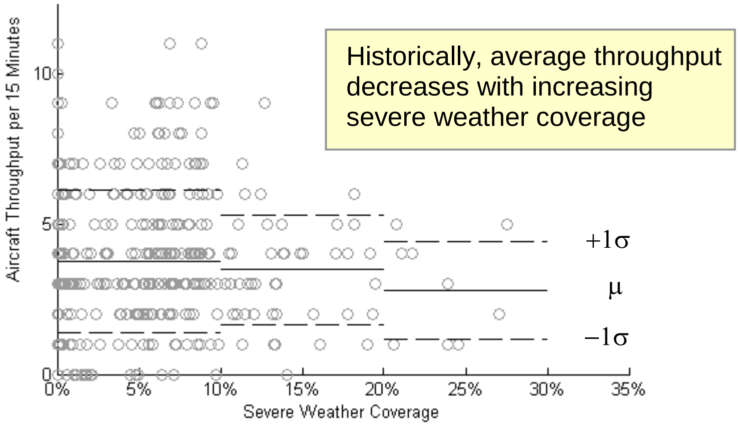


Figure 13. Historical throughput rates vs severe weather coverage.

provides a baseline for comparison of different approaches to the hazardous-weather routing problem.

2.11 Output Performance Metrics

It is necessary to define performance metrics to evaluate different algorithms on the same ATM scenario (as in "Weather Avoidance in Transition Airspace"), or examine different ATM scenarios to see how they compare (as in "Capacity Estimation for Level Flight with Convective Weather Constraints"). During the course of the research, two such metrics were defined: throughput and complexity.

2.11.1 Throughput

Throughput is measured in terms of raw aircraft per hour. No particular attention is paid to their headings, their destinations, how far they have traveled or any other measurement. Since throughput is measured on already-routed aircraft, it is a given that all counted aircraft are in fact legal flights.

Throughput is calculated either in terms of the number of starting aircraft or the number of completing aircraft in a given period of time, as appropriate to the particular study. All aircraft are represented as points; either an aircraft is considered to have completely entered the region, or it has not entered at all. If the calculation is made in terms of starting aircraft, then any aircraft that enters the region of interest between the beginning (inclusive) and end (exclusive) times of the simulation is counted as having been serviced during that time for throughput purposes. If the calculation is made in terms of routing completions, then of course it is based on the number of aircraft that exit the region of interest (either by landing or by traveling to another sector).

The maximum throughput across a point is easily calculated based on the speed and separation of the aircraft: $\tau_{\max} = \frac{vt_s}{60\sigma}$, where *v* is the given velocity (nm/hour) at the point (assumed to be constant), σ is the minimum separation (nm), and t_s is the sampling period (min). If there are a limited number of points that are legal entrance or exit points for the aircraft – particularly if the minimum separation constraint prevents any two points from being close together – then the maximum throughput is easily calculated as the sum across all legal entrance/exit points. For virtually all of our experiments, v = 420 nmi/hour, and σ = 5 nmi. Therefore 84 aircraft per hour can fly past a particular point; if there is room for N routes in an airspace, then 84N aircraft/hour can fly through that airspace.

2.11.1.1 Empirically Calculating Maximum Throughput

In order to have a better sense of success or failure of a routing solution, it can be useful to know, for a particular weather scenario, an upper bound on the amount of throughput that can actually be expected at the airport metering fixes. One reasonable way to do this is to set up flights so that they are guaranteed to be piped through the metering fix at the maximum possible rate, and see how many flights are actually able to be routed.

To perform this calculation, we use the RTA Free Flight algorithm, with one minor modification: every single metering fix has as many flights as it possibly can, lined up as tightly as possible given the aircraft horizontal separation standard. Once this calculation is complete, we can simply count how many aircraft were successfully routed.

2.11.2 Airspace Complexity

Airspace complexity is often measured in terms of dynamic density. In [RTCA], dynamic density is described as the "essential factor affecting conflict rate in both the en route and terminal airspace." These factors are traffic density, complexity of flow, and separation standards. In several investigations (16, 17, 18) on dynamic density, the relative importance of factors effecting dynamic density were determined. These investigations typically determine dynamic density as a linear combination of multiple factors. However, reviews of the literature (19, 20) do not report any single agreed upon model for dynamic density.

Most studies consider only a few of the top ranking factors in a dynamic density measurement for airspace complexity:

- The density $\delta = N / A_{ref}$ of aircraft, which is the number N of aircraft per reference area A_{ref}. The reference area is typically defined to be a sector or a circular region.
- The average proximity of neighboring aircraft δ_{NN} (the average nearest neighbor distance): $\delta_{NN} = \frac{1}{N}\sum_{i=1}^{N} \min_{i \neq j}\{d_{ij}\}$, where d_{ij} is the distance between aircraft i and aircraft j at time t.
- The average Points of Closest Approach (PCA) distribution, given by: $\delta_{PCA} = \frac{1}{N}\sum_{i=1}^{N} PCA_i$, where PCA_i is the distance of closest approach between aircraft i and any other aircraft j, over a look ahead window of time.

Studies often use one of these factors or some linear combination of these factors, with the relative weighting determined by various issues relevant to the study.

In virtually all of our research, the goal has been to maximize throughput, without regard to complexity. Since this increases the density of aircraft while decreasing their proximity, any of the aforementioned complexity heuristics would go up as a direct consequence of success of our algorithms. Because of these issues, metrics from the literature that focus on density or proximity to neighboring traffic will not add discriminative value to our research. Instead, we define a new complexity metric, based on both density and the relative similarity of headings of neighboring aircraft.

The complexity metric we utilize in our study was originally defined in [Krozel et al., 2007]. It is defined at points of a regular square grid within an airspace. The complexity at grid point p is defined in terms of a weighted average of the variance of the velocity vectors of aircraft in the neighborhood of p, scaled according to the distance of the aircraft from p. Specifically, for some radius *R*, for each aircraft a_i, a scaling factor, $s_i(p)$, is introduced:

$$s_i(p) = \frac{\max\left(R - d(a_i, p), 0\right)}{R} \tag{1}$$

where $d(a_i,p)$ is the distance from a_i to *p*. This scaling is such that the contribution of an aircraft to the airspace complexity at *p* falls off linearly with distance from *p*, up to the radius *R*. (Any continuous monotonically-decreasing function, such as a Gaussian, could be applied as well, as long as it is very close to zero at radius *R*.) Flights for which $d(a_i,p)>R$ are not considered in the calculation of the complexity for point *p*. The average contribution of all aircraft is computed, with aircraft a_i's contribution scaled according to $s_i(p)$.

For an instant in time *t*, we define the average local velocity and variance of velocity as follows. Let $V_{avg}(p,t)$ be the local average velocity vector in the neighborhood of grid point *p* at time *t*, scaled according to the factors $s_i(p)$:

$$V_{avg}(p,t) = \frac{\sum s_i(p) v_i(t)}{\sum s_i(p)} \tag{2}$$

where $v_i(t)$ is the velocity (vector) of aircraft a_i at time *t*. The (scalar) quantity $\left\| v_i(t) - V_{avg}(p,t) \right\|^2$ gives the squared deviation of the velocity of aircraft a_i from the local average velocity vector in the neighborhood of grid point *p* ($\|u\|$ is the Euclidean length of vector *u*). Intuitively, the larger this quantity, the more variation there is in the velocity vectors, as contributed by aircraft a_i, in the neighborhood of point *p*. Summing over all aircraft, and scaling by $s_i(p)$ to account for the distance from point *p*, we obtain the expression for the scaled-contribution velocity variance at point *p*, at time *t*:

$$Var(p,t) = \sum s_i(p) \left\| v_i(t) - V_{avg}(p,t) \right\|^2 \tag{3}$$

Our complexity metric is based on a linear combination of this variance term and a density term, $N(p,t)$, defined to be the number of aircraft at time *t* within distance *R* of point *p*. The overall composite complexity metric takes into account velocity variation and density:

$$C(p,t) = \lambda_1 \sum s_i(p) \left\| v_i(t) - V_{avg}(p,t) \right\|^2 + \lambda_2 N(p,t). \tag{4}$$

In our experiments, we use λ_1=0.36, λ_2=2, and *R*=35 nmi. The overall complexity at time *t* in a given region of airspace is obtained by summing *C(p,t)*, over all grid points within the region. The time is discretized into 1-min intervals to get a better time-averaged view of complexity. We sum the complexity values over 15-min intervals to get the complexity values reported.

2.12 Running time

The heart of the flow-based route planner is the A* search algorithm, which is a variant of breadth-first search. Each edge, however, does not have a constant cost, because it must be directly checked against the weather and existing edges. (The cost in time and memory to compute this ahead of time would be larger than the real-world cost of the algorithm itself.)

To check an edge against weather, there are two non-constant steps: the convolution operation and the scan-conversion. The convolution result can be calculated once for each weather time slice and stored, so that the total cost across the entire algorithm is equal to the cost of all the convolutions: O(Wd * Wd * Wt * epsilon * epsilon), where Wd is the size of each weather sample along each side, while Wt is the number of weather time slices. For the scan-conversion, in the worst case an edge can cross O(Wd) pixels and span every single weather sample, for a weather check cost of O(Wd * Wt) per edge.

To check an edge against other edges, we must calculate its polytope approximation and then query the K-dop tree with it. In the worst case this query could take O(n) time, as every node of the tree may need to be checked. If c is the maximum number of segments per route and n is the number of flows being solved, then the query will take O(cn) time.

For real-world running time information, see the section "

7.6 Running Time" on page 89.

2.13 Running space

In order for the algorithm to run, it needs the space required by the search, plus all space required by the constraint sets – the weather files and the existing route segments. (The static constraints are constant in their space requirements.)

The weather constraints theoretically and practically require space proportional to the resolution squared, multiplied by the size of the area of interest, multiplied by the number of time samples. In practice, the area of interest for our algorithmic runs is 480 nautical miles square, twelve hours long, and each weather cell in the convolved weather is one nautical miles on a side, with samples every five minutes; so therefore there are $(480)^2 * (12 * 60 / 5) = 33,177,600$ weather cells. Each actual weather cell is one byte, leading to 33 megabytes of space for weather alone.

Existing route segments are stored as a list of K-dop trees. The lists themselves have no special structure. The K-dop tree is simply a binary tree with constant-size auxiliary information stored at each node (the coordinates of the K-dop bounding volume's faces). The route segments themselves require the storage of two polygons at the bottom of a K-dop tree structure. Each of these is linear in the number of route segments. Since the number of segments per route is at most a constant, the size of the stored route segments is O(CN), where N is the number of routes and C is the maximum number of legal segments per route.

The search itself can, at worst, search every single possible vertex and edge in the graph, checking each edge at most once. Because of the construction of the initial search graph, we know that, if K is the connectivity constant, then each node connects to at most $O(K^2)$ other nodes. Therefore the edges $E = O(K^2V) = O(K^2D^2)$ where D is the size of one dimension of the grid and the graph must store information for at most $O(K^2D^2)$ edges in order to run to completion.

For the refinement search, the situation is very similar. Because of the structure of the search and the angular constraint, the maximum number of segments in a completed initial search is O(D), the size of one dimension of the grid, so that the number of vertices in the refinement search is O(DM), with M the number of Steiner points per original point, and the number of edges to examine and store is $O(D^2M^2)$.

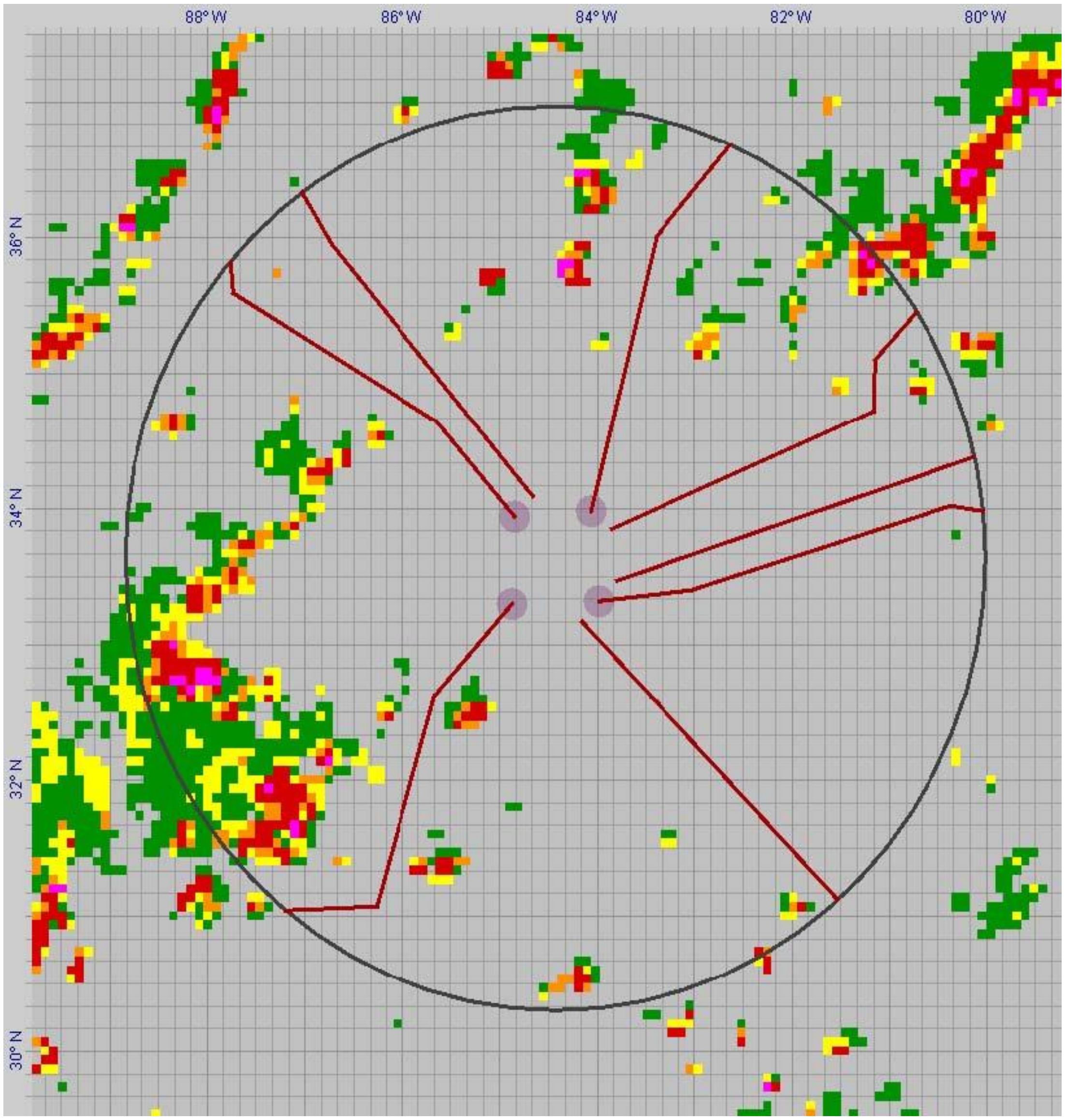


Figure 14. Arrivals at Hartsfield International Airport, Atlanta

The algorithm was run with historical weather data from June 27, 2002, from 20:00 to 24:00. The routes shown are valid between 20:00 and 20:20. The violet circles are the locations of standard arrival metering fixes. Each metering fix is assigned three waypoints on the range ring based on historical flight data; the algorithm then finds routes from the range ring to the arrival fix.

In this particular case, three routes were attempted to each of the four metering fixes, but some were blocked entirely by hazardous weather and could not be routed.

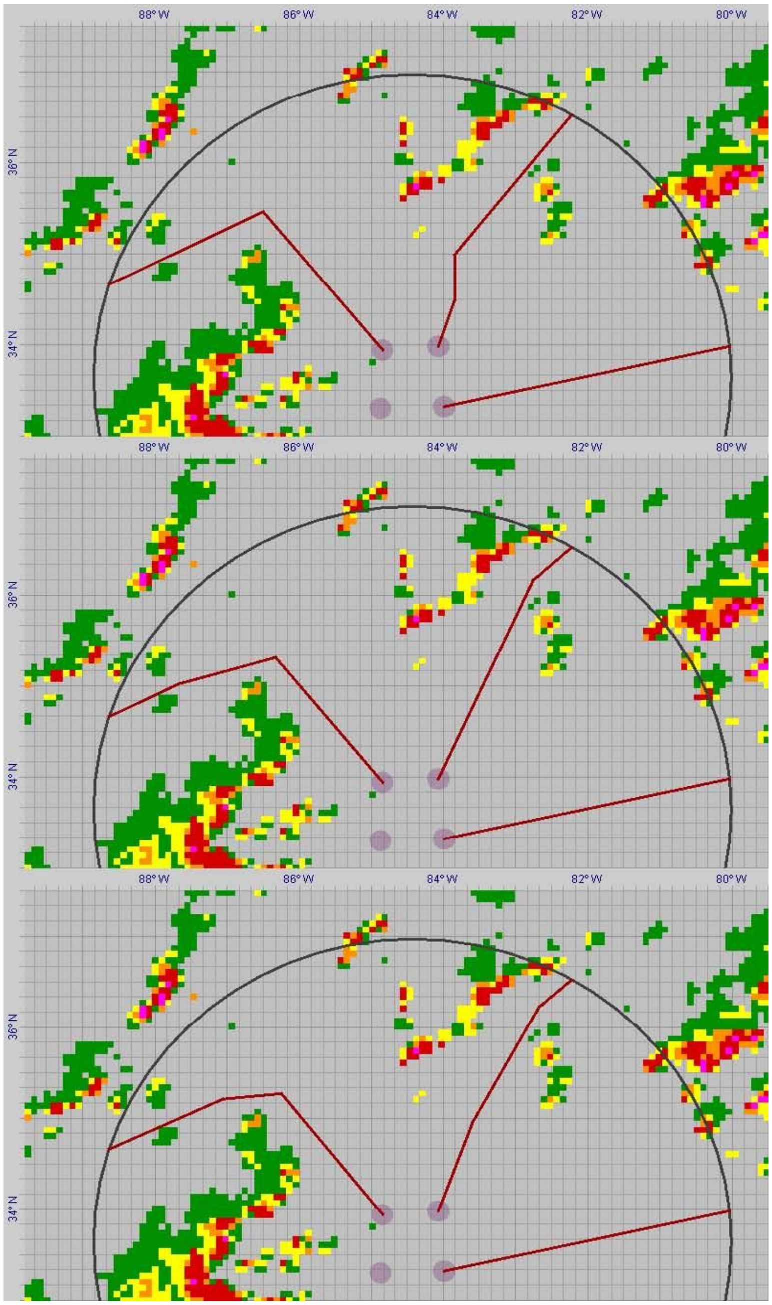


Figure 15. The effects of different grid sizes on the final results.

In this case only one route was associated with each metering fix. The three diagrams represent grid sizes, from top to bottom, of 16, 64, and 256 grid lines across. The area of interest is 480 nmi square, so that the three diagrams represent, from top to bottom, a cell size of 30 nmi, 7.5 nmi, and 1.875 nmi respectively.

The first solution illustrates the difficulties of a large grid size; note that the western route takes a very large detour around hazardous weather. The second solution (with a 7.5 nmi cell size) is very close to ideal route lengths. The third solution, with very small 1.875 nmi cells, is no better than the second – but took much longer to compute.

These routes are valid for arrivals at Hartsfield International Airport, Atlanta, on June 27, 2002, between 21:20 and 21:40.

Chapter 3: Weather Avoidance in Transition Airspace[1]

3.1 Introduction

Air Traffic Management (ATM) in the airport transition airspace works well under normal operating conditions. However, during inclement weather, the reduction in available non-convective airspace limits capacity, adversely affecting throughput. The Aviation Capacity Enhancement Plan lists weather as the leading cause of delays greater than 15 minutes, with terminal volume as the second leading cause. As shown in Figure 1, weather-related delays, which have steadily increased in the convective weather seasons prior to Sept. 11, 2001, are likely to get worse as the volume of air traffic increases in the future.

While the weather has not fundamentally changed over the years, delays have steadily increased because there have been greater demands placed on limited resources such as runways, navigational fixes and jet routes. Those resources experience significant capacity reductions during severe weather events. Weather avoidance maneuvering that is performed is typically "reactive" to deviate around hazardous weather cells and is not designed to be optimal in any way. Safety constraints dictate that aircraft must remain separated from one another as well as from hazardous weather. As aircraft deviate around weather, they stray from the preferred direct-to route and require neighboring aircraft to "keep out of their way" (Air Traffic Control (ATC) provides the function of safely separating aircraft). Neighboring aircraft are typically spaced out at greater distances behind a leading aircraft in a flow as ATM personnel increase mandated Miles-In-Trail (MIT) between aircraft, which effectively reduces the flow's throughput and adds to the delays experienced during severe weather events. Thus, not only are aircraft delayed due to a path stretching effect of weather avoidance maneuvering, they are also delayed due to MIT restrictions imposed to slow the traffic flow rate and make it more manageable during weather events.

This chapter compares three solution approaches to synthesizing (by computer automation) weather avoidance maneuvers for flows of aircraft in the transition airspace – within a roughly 200 nmi range to the metering fixes (MFs) of an airport. Using current National Airspace System (NAS) operations as a baseline, new route planning solutions are investigated.

3.1.1 Variations on Standard Arrival Routes (STARs)

The Navigation aids (Navaids) that constitute the STARs of an airport are varied by a set amount of lateral separation to design a new STAR as a function of time of day

[1] The material in this chapter is adapted from Krozel, J., Penny, S., Prete, J., and Mitchell, J.S.B., "Comparison of Algorithms for Synthesizing Weather Avoidance Routes in Transition Airspace," *AIAA Guidance, Navigation, and Control Conf.*, Providence, RI, Aug., 2004.

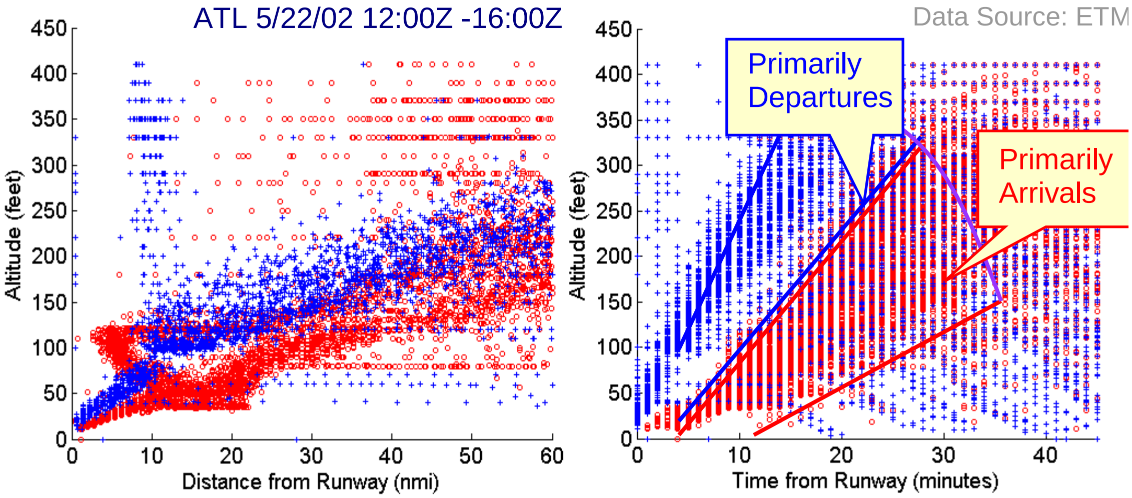


Figure 16. Arrivals and departures typically descend and climb at different rates.

that safely avoids hazardous weather. Aircraft are metered at the 200 nmi range ring (with speed control applied on the routes) so that only one aircraft arrives at a MF at a time, regardless of the weather avoidance route chosen.

3.1.2 Flow-Based Route Planner (FBRP)

Multiple, non-intersecting routes are designed to lead from a 200 nmi range to airport MFs. Multiple routes are synthesized for each MF, based on maximizing the total number of routes that both avoid hazardous weather and do not intersect each other (to reduce complexity and avoid the need for metering the flights at additional points). Given a set of routes leading to the MFs, the route that minimizes the distance traveled, subject to the constraints, is chosen for each aircraft. Aircraft are metered (at the 200 nmi range with speed control applied on the routes) so that only one aircraft arrives at a MF at a time, regardless of the weather avoidance route chosen.

3.1.3 Free-Flight

Free Flight routes are synthesized to avoid hazardous weather and to separate aircraft from those already routed in the transition airspace. The algorithm is “greedy” in the sense that each aircraft acts independently, selecting a best safe route between the 200 nmi range ring and the MF, subject to the constraints imposed by earlier aircraft arriving into the airspace. Separation requirements implicitly create the metering of aircraft at the MF. Two forms of Free Flight are studied. In Required Time of Arrival (RTA) Free Flight, aircraft are required to arrive at a MF at the RTA allocated by ATC to maximize throughput while avoiding conflicts at the fix. In First-Come First-Served (FCFS) Free Flight, aircraft are routed on a FCFS basis as they arrive at the 200 nmi range ring.

3.2 Theoretical Problem Statement

The transition airspace problem for ATM is a 4-Dimensional (4D) problem that includes arrival and departing traffic heading to/from an airport. However, due to current

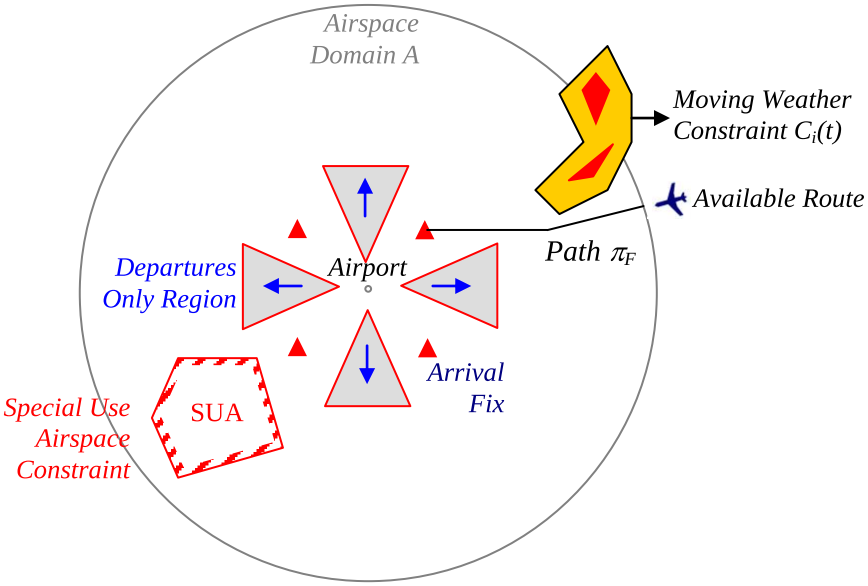


Figure 17. **Geometry and constraints for the arrival transition airspace problem.**

ATM practices, the dimensionality and complexity of this problem is reduced and simplified in many ways in today's ATM system. For example, aircraft are routed only on well-defined jet routes and aircraft only turn at Navaids. Analysis of the routes leading to and from Atlanta's Hartsfield International Airport (ATL), for instance, indicates that the arrival and departure traffic have standard climb and descent profiles that separate the arrival and departures vertically. Horizontally, within roughly 50 nmi of the airport, the arrival and departure traffic are also separated by ATC low altitude sectors that are designated as either arrival sectors or departure sectors. Some of these same practices that reduce the dimensionality of the problem statement are retained in our problem statements to make our solutions operationally feasible and compatible with today's ATM system.

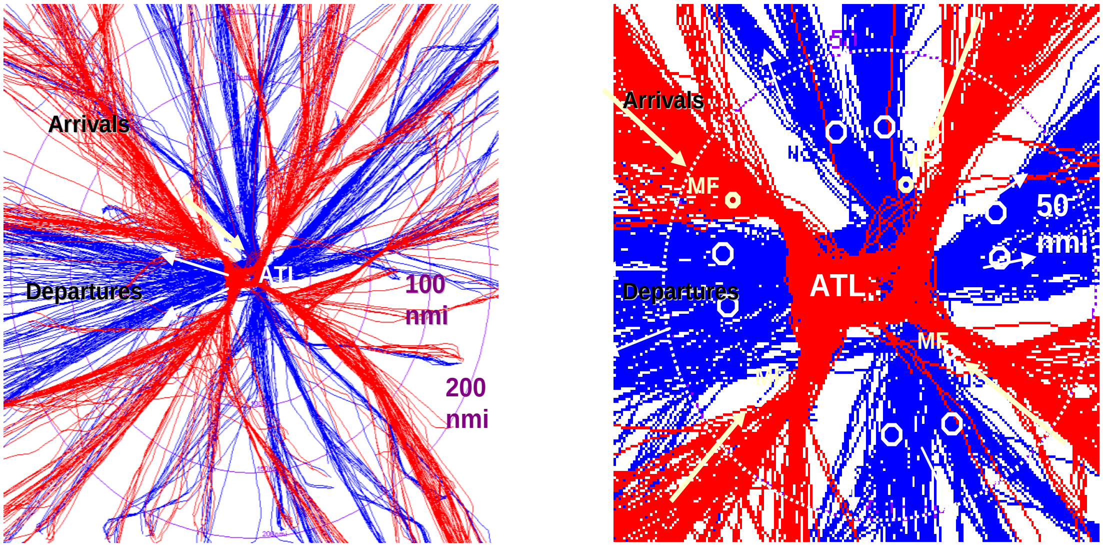
Optimizing traffic flow in the transition airspace must take into consideration safety, efficiency, workload (complexity), operational feasibility (supporting infrastructure), and roles and responsibilities (for pilots and controllers). These are taken into consideration as mathematical problem statements are developed for operationally feasible methods of routing traffic around hazardous weather cells in the transition airspace.

3.2.1 Problem Geometry and Constraints

Each desired route is specified with a starting waypoint, o_F, and a destination waypoint, d_F. o_F represents the point on or near the 200 nm range ring at which a flow of aircraft is to enter the airspace, and d_F represents the metering fix to which these aircraft are to be routed. We are given a specification of the 200 nm radius circle centered at an airport, and a set of start and end times, T^s_F and T^e_F, respectively; these represent the time window for flow *F*. Normally these start and end times are spaced at a particular time increment, and are the same for all origin-destination pairs. The speed *v* of aircraft within the flow is also specified. More generally, a speed profile may be specified; however, this option is not yet implemented.

Our problem is described as follows. For each flow *F*, our goal is to calculate a legal route from o_F to d_F for the set of aircraft that arrive at o_F during that time window. All aircraft that arrive at time *t* (where $T^s_F \leq t \leq T^e_F$) will start at waypoint o_F and travel along the route at speed *v* until they reach the goal waypoint d_F. Thus, the route at any given time is not just a point but is a subroute of length equal to $T_i\, v$; at any given point, the route extends over a time window of duration (T^e_F - T^s_F). Each route is to be computed so that it is as short as possible – or so that it can minimize some more complex objective function – subject to many constraints. We outline them below, but a complete description can be found in the first chapter:

- The route avoids hazardous weather by a distance of at least a specified safety margin, ε. The weather intensity threshold (above which weather is considered hazardous, and is to be avoided completely) is user-specified.
- The route maintains the specified horizontal separation standard δ with respect to all other flow routes already determined.
- The route contains no more than *k* waypoints.



(a) 200 nmi Plan View

(b) 50 nmi Plan View

Figure 18. Arrivals and departures typically are separated horizontally near the airport.

- The route contains no segments of length less than some minimum length L.
- Route segments are only permitted to be within Θ degrees of the direction from o_F to d_F.

All routes that are going towards the same metering fix at the airport and arriving at the same time are considered a single *fan-in*. In order to maintain the separation standard, the goal points are spaced slightly apart at the metering fix. The aircraft are assumed to be metered so as to produce the desired arrival rate for that metering fix.

3.2.2 Aircraft Dynamics and Aircraft Flows

Aircraft are modeled as points in motion within A. Aircraft dynamics are specified in terms of bounds on the speed and the magnitude of acceleration. While acceleration bounds give rise to bounds on the radius of curvature of flight, we assume that the scale of our solution space is large enough that we can approximate aircraft dynamics with a simple representation of piecewise-linear flight segments connected at waypoints. Such is the standard notation for flight plans used today. We assume that there is a (location-based) *speed profile* $v: A \rightarrow [v_{min}, v_{max}]$, which maps each point $(x,y) \in A$ to a nominal speed $v(x,y)$ within the interval of possible aircraft speeds, between v_{min} and v_{max}. An aircraft at position (x,y) is expected to fly at a speed within the range $[v(x,y) - \Delta_v, v(x,y) + \Delta_v]$, where $\Delta_v > 0$ is a specified speed tolerance. In the transition airspace problem, the speed profile allows one to model the fact that aircraft are changing speeds as a function of distance from the airport, according to standard ascent or descent rates and ATC practices.

Note that our model does not explicitly take into account the fact that different classes of aircraft may travel at different speeds. We assume that all aircraft are expected to travel according to a single ascent or descent speed profile along any given route. Furthermore, our implementation and experiments are based on using constant speed profiles, in which $v(x,y) = v$, where v is the average speed for crossing the transition airspace.

A set of flow demands $\boldsymbol{D}$ represents flow requirements for aircraft passing through A. Each *demand* $D=(o_D, d_D, [t_D^s, t_D^e]) \in \boldsymbol{D}$ consists of an *origin* of the flow $o_D \subset A$, a destination of the flow $d_D \subset A$, and a time interval $[t_D^s, t_D^e]$ over which the demand occurs, from start time t_D^s to end time t_D^e. During the time interval $[t_D^s, t_D^e]$ a stream of aircraft are arriving at o_D heading to the destination d_D. We define o_D and d_D to be sets, in general, rather than singleton points, to allow flexibility in where exactly a flow is to originate or terminate. In the transition airspace, for example, we may define the origin o_D for arriving flights to lie within an arc on the range ring R, and the destination to be either a *MF location*, a destination point d_D fixed in space for all time t, or a *variable MF location*, a destination d_D variable in location and time t, chosen to maximize throughput.

When varying the MF location, one of either 3 or 9 possible discrete MF locations along an arc centered at the original metering fix location are considered.

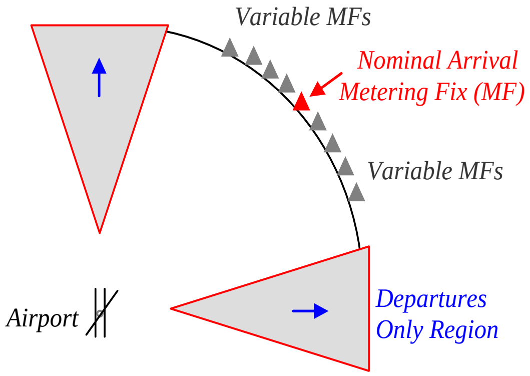


Figure 19. Nominal arrival Metering Fix (MF) and variable MF locations.

Define a *flow F* to be a route $\pi_F \subset A$, from a point o_F to a point d_F , and an associated time interval $[t_F^s, t_F^e]$ during which the flow is *active*. A flow *F* is *feasible* if, for any $t_0 \in [t_F^s, t_F^e]$, an aircraft that arrives at o_F at time t_0 and flies exactly along the route π_F according to the speed profile function, is at all times safe with respect to the set of constraints *C(t)*. More precisely, flow *F* is feasible if for any $t_0 \in [t_F^s, t_F^e]$ and any $t \in [0, t_\pi]$, we have:

$$(x(t_0 + t), y(t_0 + t)) \in A \backslash \{C \in C(t_0 + t)\}$$

locating at time t_0+t a point along π_F that arrives at point o_F at time t_0 and moves along π_F according to the speed profile function. (Thus, this location is obtained by integrating the speed profile function along the route $\pi_{F.}$) Here, t_π denotes the time it takes to fly along π_F, from o_F to d_F , with speed that matches the speed profile function.

Flows are designed not to interfere with each other in the transition airspace. A *set of flows* $\boldsymbol{F}$ is *feasible* if each flow $F \in \boldsymbol{F}$ is feasible and for any pair of flows, $F_1, F_2 \in \boldsymbol{F}$ and any time $t \in [t_{F_1}^s, t_{F_1}^s + t_{\pi_1}] \cap [t_{F_2}^s, t_{F_2}^s + t_{\pi_2}]$, the position of an aircraft at time *t* along the route π_1 for F_1 is separated from the position of an aircraft at time *t* along the route π_2 for F_2 by at least the horizontal separation distance δ.

A feasible flow *F* is said to satisfy demand *D* for time interval Δt if it starts and ends according to the demand ($o_F \in o_D$ and $d_F \in d_D$), and its time interval has an overlap of length Δt with the time interval of *D*: i.e., $\| [t_F^s, t_F^e] \cap [t_D^s, t_D^e] \| = \Delta t$.

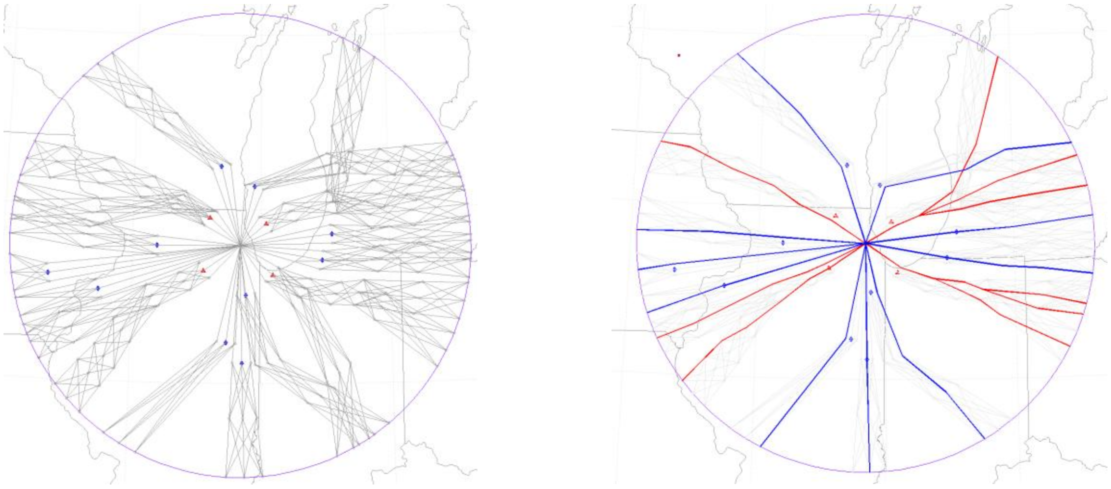
The throughput associated with a feasible set of flows is defined to be the total sum of the lengths of the demand time intervals that correspond to feasible flows that satisfy the demand. More precisely, the *throughput, $\mu_{\boldsymbol{F}}(D)$, of F provided for demand D* is defined to be the total length (measure) of the subintervals of $[t_D^s, t_D^e]$ that are satisfied by at least one flow in a given feasible set $\boldsymbol{F}$ of flows. The *total throughput* of the set $\boldsymbol{F}$ is defined to be $\sum_{D \in \boldsymbol{D}} \mu_{\boldsymbol{F}}(D)$.

A flow can be thought of as an “opportunity" for routing aircraft to satisfy demand; it is a “highway in the sky" that is open for a specified interval of time. In space-time, a flow corresponds to a (continuous) family of trajectories having the same geometry in space (following the same sequence of waypoints), but having starting times that vary within a certain window of time; thus, a flow can be visualized as a “swept” trajectory in space-time, for which there is an opportunity for multiple aircraft to follow safely the same trajectory, spaced according to MIT restrictions. A feasible set of flows represents an opportunity for multiple trajectories having distinct geometry, while guaranteeing safety, both in terms of weather constraints and in terms of horizontal separation standards.

To simplify the presentation of results, we only compare results for arrival routes, and do not analyze the coupling of arrival routes and departure routes; nonetheless, our algorithms and solution approaches are quite general and are easily applied to the problem of synthesizing arrival and departure routes simultaneously. For instance, the same algorithm that is used to design variable STARS can be used to design variable Standard Instrument Departure Routes (SIDS). Furthermore, we do not investigate the synthesis of weather avoidance routes from the MFs to the runways. These routes can be synthesized using similar approaches to those that are applied here to the transitional airspace; see Krozel et al ([KLM]) for an algorithmic solution. Finally, we do not explicitly address vertical separation issues, because real-world data suggests that the arrival and departure routes can be separated to different flight levels based on standard arrival descent profiles and standard departure climb profiles that can be designed to avoid vertical conflicts.

3.3 Algorithmic Solutions

Algorithms were developed for the three solution methods for routing either flows of aircraft or individual aircraft (in the Free Flight method). A set $\boldsymbol{W}$ of *waypoints* defines the points at which a flow/route is allowed to turn. Each of the methods involves computing one or more routes in a search graph, $G=(\boldsymbol{W},E)$, with node set $\boldsymbol{W}$ and edge set E. Historically, these locations have been specified by the physical positioning of Navaids, physical navigation aid devices located on the surface of the Earth. With current navigation equipment, though, points of $\boldsymbol{W}$ can now be located at essentially any point of airspace A. Depending on the model described below, the waypoints will be defined as either fixed locations or they will vary as a function of the location of the constraints. Further, we consider cases in which the MF is either fixed in location, as dictated by current operational standards, or variable in location, allowing the MF to move in location within a small tolerance.



(a) Underlying graph of variable SIDs/STARs

(b) Planned routes (arrivals red, departures blue)

Figure 20. The ORD weather avoidance search graph based on SIDs and STARs.

3.3.1 Variable STARs

The Variable STARs method creates alternate routes around hazardous weather using the STARs as a baseline and additional waypoints that are varied within a fixed lateral offset from the STAR. The underlying search graph $G=(\boldsymbol{W},E)$ is defined as follows: the set $\boldsymbol{W}$ of nodes of G consists of the STAR Navaids, together with waypoints defined as shifted copies of each Navaid, plus or minus a lateral offset. The lateral offset varies from 8 nmi at the metering fix to 16 nmi at the 200 nmi range, defined by a linear function $\omega\big(d(a_0,w)\big)$, where $d(a_0, w)$ is the distance from the airport a_0 to the waypoint $w\in\boldsymbol{W}$. Let w_i be a waypoint that is part of a specific STAR, and let $d(a_0, w_i)$ be the distance from the airport a_0 to the waypoint w_i. If c_1 is a circle centered at the airport with radius $d(a_0, w_i)$, and c_2 is a circle centered at the waypoint w_i with radius $\omega\big(d(a_0,w_i)\big)$, then the 2 points of intersection of circles c_1 and c_2 define the alternate (shifted) waypoints for w_i.

We define the edges E within graph G as follows. For each STAR, we denote the STAR's original (n + 1) base waypoints by w_0, w_1, …, w_n, with $w_0 = a_0$, and the 2n new alternate waypoints by $w_{1,1}$, $w_{1,2}$, $w_{2,1}$, $w_{2,2}$, … , $w_{n,1}$, $w_{n,2}$. Edges E in the graph connect w_0 to each of the waypoints w_1, $w_{1,1}$ and $w_{1,2}$, and connect each vertex at stage j, (i.e. w_j, $w_{j,1}$ and $w_{j,2}$), to each vertex at stage (j + 1), (i.e. w_{j+1}, $w_{j+1,1}$, $w_{j+1,2}$). For each time interval $[t_F^s, t_F^e]$, we search the graph, using an A* algorithm, for an optimal path from w_0 to one of 3 goal points, w_n, $w_{n,1}$ and $w_{n,2}$,. The resulting route π_F defines a flow F over $[t_F^s, t_F^e]$, which satisfies demand over this time interval.

In operation, the lateral offsets may be communicated easily between controllers and pilots (verbally or with automation), and familiarity with SID and STAR Navaids is not lost. The method maintains close similarities with current ATC operations while providing some flexibility and automation for hazardous weather avoidance. Such

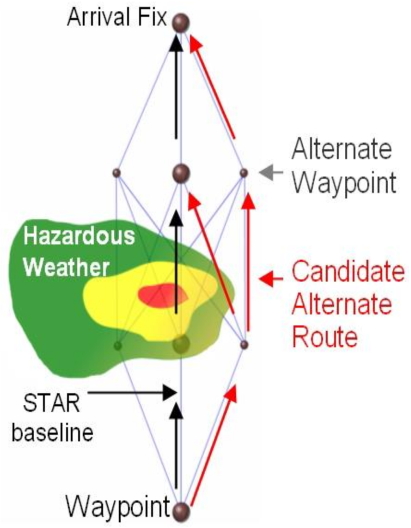


Figure 21. Alternate weather-avoidance waypoints relative to STAR waypoints.

similarities are clearly a benefit during the deployment stage of a route-synthesizing algorithm.

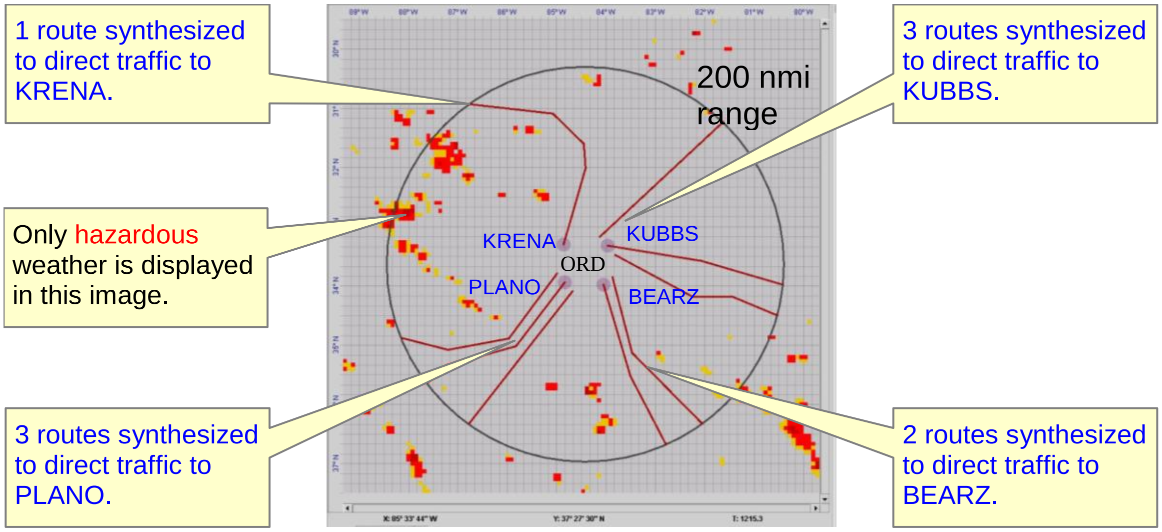
A limitation of this method is that synthesized routes may have no choice but to penetrate hazardous weather. In such a case, the flow is not feasible, and aircraft are restricted from using the route. Aircraft are not permitted to enter the airspace until the algorithm considers the next time interval and can synthesize a feasible route that avoids hazardous weather. In operation, this would lead to route closures with consequent delays assigned upstream. MIT restrictions of 5 nmi consistent with current-day minimums are maintained along the solution route.

The Variable STAR method may use either fixed MF or variable MFs. When varying the MF, the Variable STAR method uses one of 3 possible discrete MF locations 8 nmi apart, centered at the original metering fix location.

3.3.2 Flow-Based Routing

The FBRP algorithm (described in the first chapter) is used to calculate a number of non-intersecting weather avoidance routes that lead from the range ring *R* to either one fixed MF or one of k_{MF} ($k_{MF}=9$ in our experiments) evenly spaced variable MFs along an arc through the nominal MF. (In practice we only calculate three routes per metering fix for this application.) Each single route is partitioned into a set of flows having short time gaps between their respective arrival time windows, in order to account for any MIT restrictions. The creation of multiple non-intersecting routes in multiple flows adds another layer of complexity to the algorithm, since horizontal separation standards must be enforced. The algorithm computes multiple routes incrementally, enforcing separation standards with respect to each route already in place.

Because conflicts can arise between already routed flows and flows yet to be routed, it is necessary to search among several possible orderings of routes, in order to find solutions to all routes (if such hazard-free routes exist). Note that while all three routes may potentially be in use, by the time aircraft arrive at the MF, they will have been controlled by speed to be separated longitudinally by 5 nmi. Thus, at the MF a continuous flow of aircraft is achieved even though the aircraft may be using different routes to arrive at the MF.



1 route synthesized to direct traffic to KRENA.

Only hazardous weather is displayed in this image.

3 routes synthesized to direct traffic to PLANO.

3 routes synthesized to direct traffic to KUBBS.

2 routes synthesized to direct traffic to BEARZ.

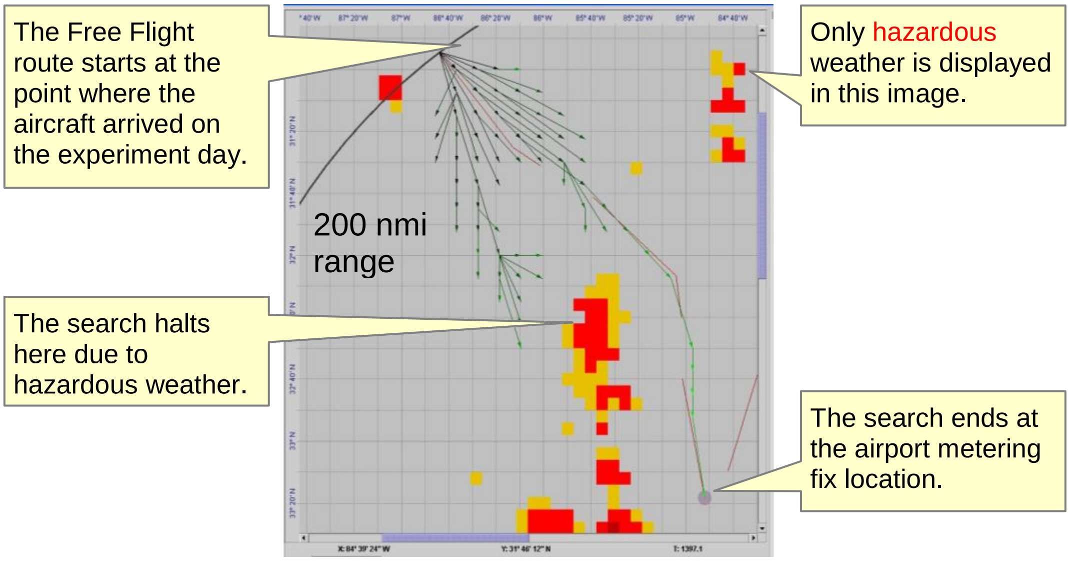
Figure 22. Sample flow-based routing results at ORD.

The ORD weather avoidance search identifies non-intersecting routes that direct independent flows from the transition airspace boundary to metering fixes.

For this application, grid points were spaced roughly 6.25 nmi apart; for a 400 x 400 nmi airspace around an airport, there are roughly 64 x 64 = 4096 grid points.

3.3.3 Free Flight

The Free Flight algorithm is designed to synthesize routes that enable individual flights to be routed conflict-free and clear of dynamic, hazardous weather constraints. As each flight arrives at the 200 nmi boundary, it is individually routed to the arrival metering fix corresponding to its quadrant of entry. The optimal path of each aircraft is routed without regard for how the decisions of the aircraft might adversely affect



The Free Flight route starts at the point where the aircraft arrived on the experiment day.

The search halts here due to hazardous weather.

Only hazardous weather is displayed in this image.

The search ends at the airport metering fix location.

Figure 23. A search identifies the optimal Free Flight route avoiding hazardous weather.

upstream aircraft (e.g., a non-optimal sequence). The solution is designed to mimic a distributed system where each aircraft non-cooperatively plans its own route.

The Free Flight algorithm uses a search in space-time (as in the Space-Time Flow method), treating already routed flights as constraints to be avoided (according to minimum separation standards between aircraft), along with severe weather constraints, no-fly zones, and quadrant constraints (confining an aircraft to the quadrant initially entered).

Free Flight routing allows maximum flexibility in terms of individual aircraft routing; however, by allowing each flight to be greedily routed, the Free Flight solution can result in some routes being particularly efficient, at the expense of blocking off a passage for other flights. Furthermore, there is no attempt in the Free Flight algorithm to sequence aircraft optimally.

We examine two variations of Free Flight: RTA and FCFS. RTA routes individual aircraft based on an optimized schedule at the metering fix that guarantees full throughput given the minimum separation requirement. This has two drawbacks: (1) routes must be generated in advance (e.g. 1 hour) of the aircraft arriving at the 200 nmi range, and (2) flights are given RTAs at the 200 nmi range to which they must adhere. FCFS is less restrictive, generating routes for the aircraft as they arrive at the 200 nmi range. While this allows more freedom for the flight operators, there is a potential for loss in capacity utilization.

The Free Flight method may use either fixed or variable MFs. When varying the MF, the Free Flight method uses one of 9 possible discrete MF locations along an arc centered at the original metering fix location.

3.4 Experiment Design

The focus of the experiments was on the Atlanta Hartsfield International Airport (ATL). ATL executes a standard 4-cornerpost fix arrangement and is a major hub with significant throughput. This allowed generalization of results, while still giving the ability to compare with historical data. Today, ATL's typical clear-weather Airport Arrival Rate (AAR) is about 100 arrivals per hour, but this rate is significantly reduced when severe weather is present. Algorithmic solutions were generated for the following time periods:

- ATL 5/22/2002 12:00-23:59 Z (8 am - 8 pm local time) (Clear to Light Weather)
- ATL 6/26/2002 12:00-23:59 Z (8 am - 8 pm local time) (Light to Severe Weather)
- ATL 6/27/2002 12:00-23:59 Z (8 am - 8 pm local time) (Light to Severe Weather)

National Convective Weather Diagnostic (NCWD) data is generated from radar Vertically Integrated Liquid (VIL) data and lightning data over the United States. The hazardous weather coverage for any given moment is simply the percent of area that is NWS Level 3 or higher relative to the total area recorded for a rolling 30-minute time period. A 30-minute time period is used because it is the approximate time it takes for an aircraft to travel from the 200 nmi range to the metering fix. For experiments with a 15-minute update time, this essentially translates to a 45-minute look-ahead at the predicted weather. An example of the severe weather coverage is shown in **Error! Reference source not found.**. Ranges of the severe weather coverage were given qualitative labels for use in discussion.

To investigate a comparison between weather avoidance algorithms implemented in today's (2002) capacity situation at individual metering fixes (4 metering fixes each for ATL), the same initial conditions were applied to the following:

- Real-World Data (Demand Baseline) – defines the initial conditions at the transition airspace boundaries and the baseline statistics for real-world weather avoidance routes; all other techniques use the same initial conditions but synthesize weather avoidance results.
- SID/STAR-based graphs (2 routes per metering fix).
- Flow-Based Route Planner (3 routes per metering fix).
- Free Flight (each aircraft self optimizes to 1 metering fix).

In all cases, the algorithms are forced to adhere to the required minimum separation of 5 nmi over the metering fixes. FBRP flows utilizing 2 or 3 routes to a single metering fix must be merged at or before the metering fix. For the flow-based routes, this is enacted through the scheduling algorithm during the simulation of flights on the routes. Therefore, it has no effect on the generation of routes. For the Free Flight approach, this separation requirement at the metering fix is a result of the constraint that aircraft maintain 5 nmi separation at all times.

Optimal routes were generated on the search graphs generated from the STARs and SIDs for each of the experiment scenarios. The algorithm assigned a cost to the routes based on the maximum weather present from the current time to the next re-planning update time plus 30 minutes (for the approximate travel time). To ensure safety, a candidate route was closed completely if hazardous weather (NWS Level 3 or above) intersected any part of it. The lateral offset for the alternate waypoints varied from 8-16 nmi as the route extended further from the airport. The following parameters were varied:

- Re-planning Update Time: 10, 15, 20, 30 minute re-plan updates.
- Severe Weather Safety Margin: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 nmi safe distance from hazardous weather.

The re-planning update time is the time that each route is active before a new route may be generated. All results are shown with 15-minute update times and a minimum 1 nmi weather separation.

The FBRP algorithm was run using three routes per metering fix. The following parameters were varied:

- Lateral Separation: 2, 4, 6, 8 nmi between routes.
- Re-Planning Update time: 10, 15, 20, 30 minute re-plan updates.

- Severe Weather Safety Margin: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 nmi safe distance from hazardous weather.

For the sake of comparison with the variable STAR approach, a lateral separation of 8 nmi was used. All results are shown with 15-minute update times and a minimum 1 nmi weather separation.

The Free Flight routes were generated by using the historical flight data as a basis for the starting positions and cross times for flights at the 200 nmi range. The separation requirement for aircraft was set at 5 nmi (for both lateral separation and MIT). The following parameter was varied:

- Severe Weather Safety Margin: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 nmi safe distance from hazardous weather.

The resulting routes for the flow-based methods were processed using a scheduling algorithm. The flights were scheduled to the route entry points (at the 200 nm range) and then at the metering fixes, given a speed profile based on historical data and a required 5 nmi separation at the metering fixes. Flight metrics were computed by simulating flights on those routes and comparing with statistics gathered from historical flight data. Flights generated in the simulation are based on initial conditions that match the crossing times and positions of historical flights. Speed profiles for the aircraft were based on historical data to approximate the effect of winds. In the flow-based techniques, it is assumed that flights are organized and routed before arriving at the 200 nmi range so that they arrive on the synthesized routes. In the Free Flight approach, the flights arrive along the 200 nmi range based on historical demand and must self-organize (each aircraft is not allowed to violate the airspace already used by upstream aircraft) into a flow over the metering fix within the transition airspace.

Output metrics were used to compare between each of the transition area weather avoidance algorithms, thus providing an equivalent basis for comparison. Throughput was measured at the metering fixes. This was computed as the number of aircraft crossing the metering fixes (calculated at the closest point of approach to the metering fix). Weather penetration was calculated as “true” or “false,” given a specified clearance (safe-distance) from hazardous weather (NWS Level 3 or above). To insure safety as a constraint in the algorithms, weather penetration was required to be 0 for any specified weather clearance. Complexity was measured as the number of changes in each aircraft’s nearest neighbor. Thus, a uniform flow of aircraft following each other one after another would be very low complexity, aircraft following distinct weather avoidance routes (flow-based or Free Flight) would potentially be a higher complexity, and a set of aircraft all flying in different directions would be the highest complexity.

In addition to simulations based on historical flight data, simulations were run to estimate the maximum possible throughput given a constant demand (rather than a fluctuating historical demand). These simulations were run for the same time periods, with identical hazardous weather data.

3.5 Comparison of Algorithms

We compare algorithms based on both throughput and complexity metrics; see the first chapter for definitions and specific discussion. For the sake of these experiments, throughput is measured at the metering fixes, which are the destination locations for all inbound aircraft.

Each approach for weather avoidance routing – historical, STARS, FBRP, and Free Flight (FCFS, RTA) are compared based on 1) throughput as a function of weather severity and 2) airspace complexity as a function of weather severity.

The illustration below shows historical throughput data for ATL. Each point represents a 15-minute time interval during the experiment period in one of four quadrants centered at the airport. The actual historical throughput rates are dependent on demand. In clear weather, periods with lower demand will clearly exhibit lower throughput. We observe periods with higher weather coverage experiencing lower average throughput and conclude there is some combination of lowered demand and lowered capacity in the transition airspace during these periods. Using our algorithms, we intend to restore guaranteed capacity (within a given error tolerance) to the transition airspace, thereby increasing demand during periods of severe weather coverage.

Each algorithm was applied to identical weather scenarios. The maximum throughput was computed for each time period in the experiment scenario, as shown in figures 24 and 25. Each data point represents a 15-minute time interval during the experiment period in one of four quadrants centered at the airport, exactly correlated with the above measurements of historical throughput rates. The difference between historical throughput and the maximum throughput computed with algorithmically generated routing is also shown.

Note that some maximum throughput rates are zero. This is a result of the nature of flow-based routing, and is an indication that a single fixed route cannot be created that will be feasible for the full 40-minute time period required to maintain a flow of traffic through the transition area. Given its greater flexibility, the FBRP method has far fewer time periods with zero maximum throughput when compared with the variable STAR approach.

While feasible routes could not be generated in these cases, a few historical flights managed to traverse the transition airspace. This is either because they penetrate weather that the algorithms are forced to avoid, or because single aircraft may safely pass through the weather; in contrast, our work is based on routing flows of multiple aircraft rather than single aircraft. Nonetheless, the historical flights are less constrained than the algorithmic approaches.

With greater flexibility comes greater throughput; this is evident by the flexibility of the routing method as well as by the selection of the hazardous weather safety margin ε. Among the flow-based methods, starting with the variable STAR approach with fixed metering fixes, to variable STAR with variable metering fixes, to FBRP with fixed metering fixes, to FBRP with variable metering fixes, the number of periods with improvement over historical throughput rates steadily increases as safety margin is reduced.

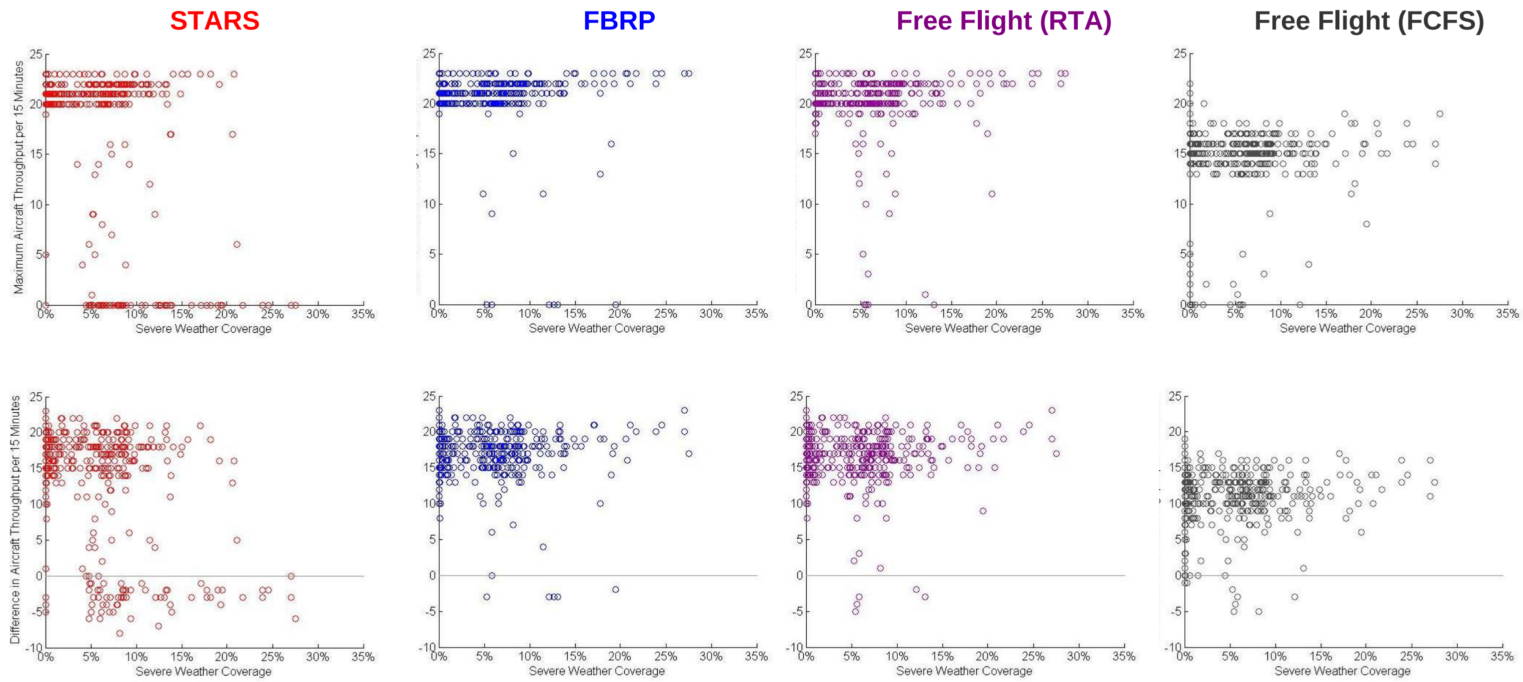


Figure 24. Maximum realizable throughput rates at the fixes for the Fixed Metering Fix method.

1 nm weather separation and 15 minute update times.

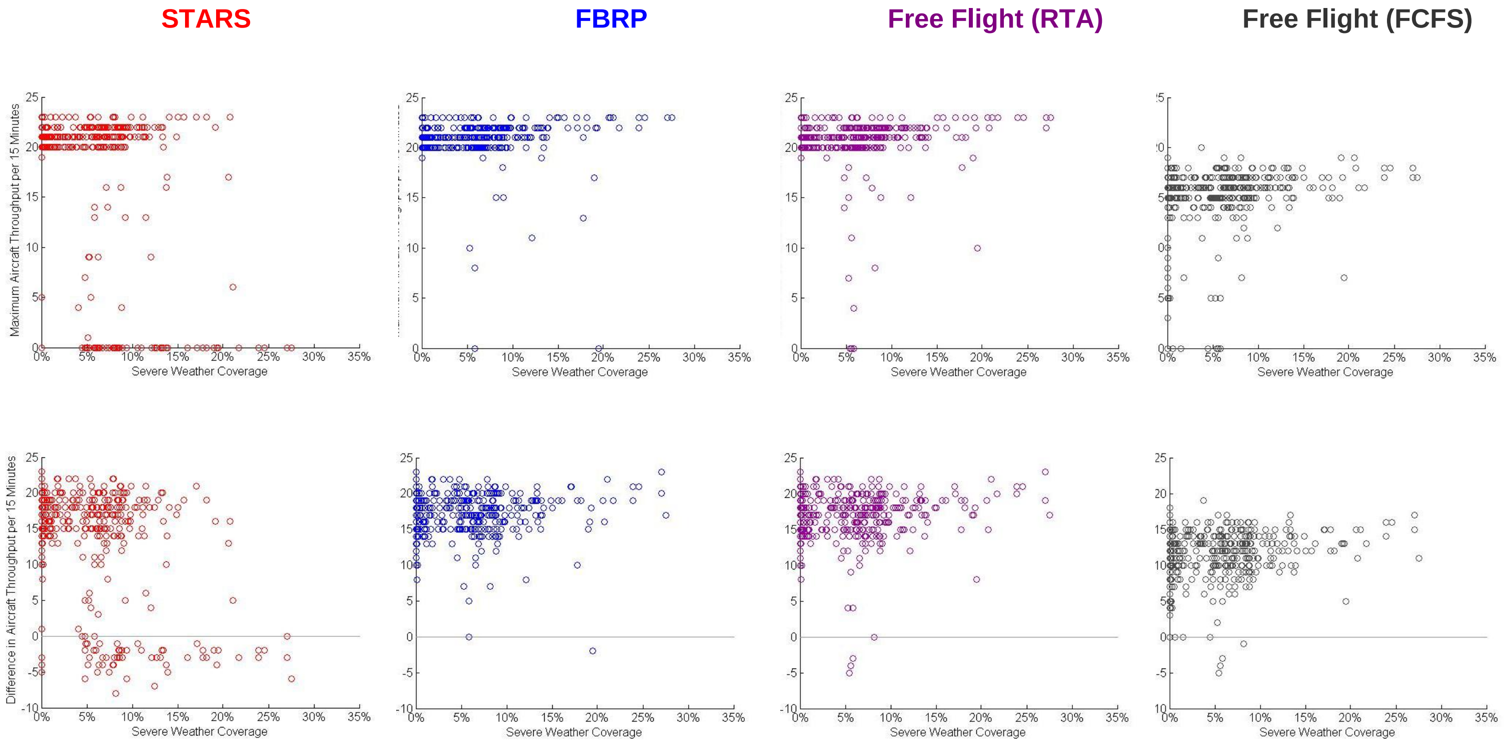


Figure 25. Maximum realizable throughput rates at the fixes for the Variable Metering Fix method. 1 nm weather separation and 15 minute update times.

3.5.1 Alternate Waypoints for Variable SIDS and STARS

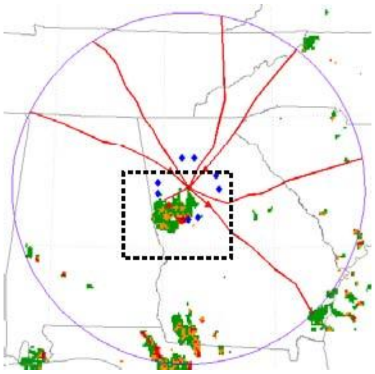


Figure 26. Route blocked by a small area of hazardous weather at the southwest fix.

The alternate waypoint approach is generally the least flexible of the three approaches. When large lines of thunderstorms intersect the base STARs, it is not possible to extend the routes far enough to avoid the hazardous weather (due to the underlying graph structure). Specifically for the overall routes, this is a hindrance because it may be possible for a flow of aircraft to fly around a large storm system. However, close to the metering fix such flexibility in distance is not required. There were cases where small shifts in the metering fix locations could maintain a flow of aircraft into the airport, where a rigid metering fix would have been blocked due to hazardous weather.

3.5.2 Flow-Based Route Planning to the Metering Fixes

The FBRP results demonstrate an increased flexibility in the general routing of flows through the transition airspace. At high levels of severe weather coverage, the FBRP routes are routable, whereas the equivalent Variable STAR routes are not. Breaks in lines are found more frequently than with the Variable STAR approach. The FBRP does, however, show weakness at the endpoints of the routes. When weather impacts the metering fixes, or in some cases the entry points, the routes are designated as un-routable. This explains the slightly weaker performance of the FBRP routes during lower severe weather coverage levels. Periods with moderate weather were sometimes blocked due to scatter weather systems. These could be 'picked through' by individual aircraft, but were not conducive to forming flows.

3.5.3 Free Flight

Intuitively, one would assume a Free Flight solution – a solution with the fewest number of constraints – to always perform better than a flow-based solution. However, this was not the case. In our algorithmic implementation of Free Flight, each flight was "greedily" routed – essentially meaning 'first-come, first-served'. Some solution routes could be particularly efficient while blocking the passage of other flights, resulting in a reduction in performance over a theoretically optimal "unconstrained" solution. Nonetheless, our model of a "greedily" routed set of aircraft is likely to resemble Free Flight where pilots and airlines self-optimize without regard for how their actions might affect others. While a theoretically optimal unconstrained solution should always be at least as good as the flow-based optimal path solution, optimizing the movement of all aircraft individually increases computational complexities and makes the system optimized (as opposed to self-optimized) Free Flight solution impractical.

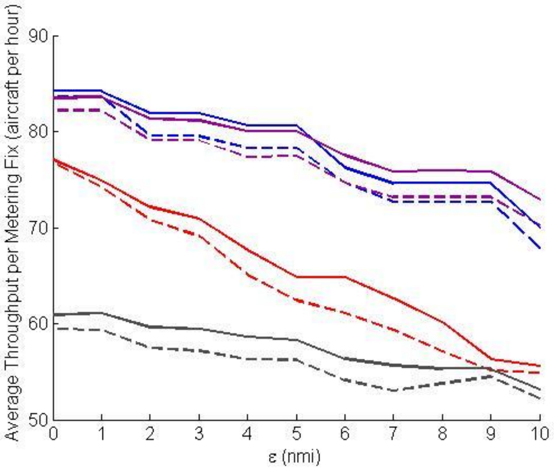


Figure 27. Average throughput with Fixed Metering Fixes (FMF) and Variable Metering Fixes (VMF) using STAR, FBRP, Free Flight (RTA) and Free Flight (FCFS) (15 minute update times).

The two Free Flight methods show drastically difference performance. Free Flight (RTA) was designed to make maximum use of arrival slots at the arrival metering fixes. The limiting factor for this method was the potential for conflicts with other aircraft routed through the airspace.

Free Flight (FCFS) experienced frequent under-usage of capacity at the metering fix. Because the flights were routed as they arrived at the 200 nmi range, with no organization of how or when to arrive, minimum separation at the metering fix could not typically be achieved. Because we required speed to be constant for these experiments, sequencing could only be achieved through vectoring of the aircraft. Increased vectoring would result in greater constraints for all following aircraft attempting to generate routes to the arrival fix.

While greater flexibility in routing affects an increase in maximum throughput, it also increases airspace complexity. The Free Flight methods show the greatest degree of complexity, as the flow of aircraft was designed to be maximized, not organized. The flow-based techniques have a greater degree of organization, and therefore have lower complexity.

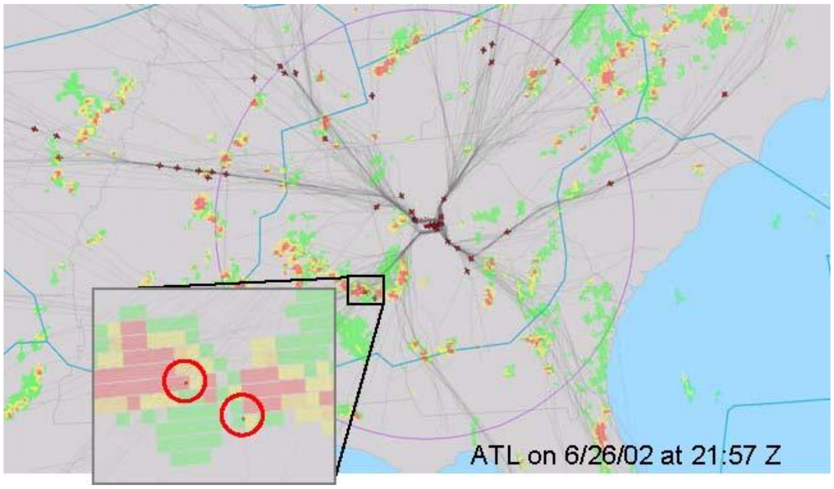


Figure 28. Historical flights penetrating hazardous weather.

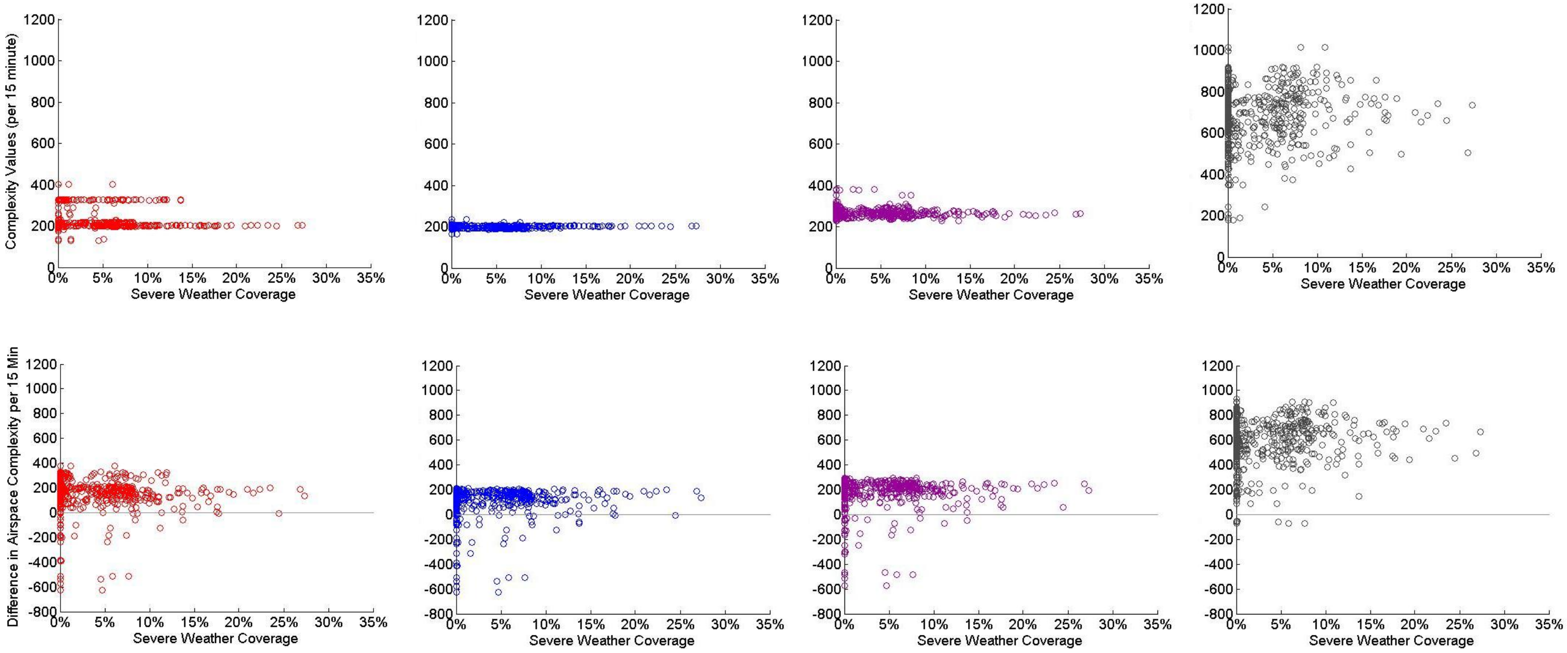


Figure 29. Complexity metric for typical and maximum throughput rates.

Fixed Metering Fix method using 1 nm weather separation and 15 minute update times.

3.6 Results

Significant differences in the routing methods occur due to route closures forced by infeasible weather situations. Note in Figure 26 a specific instance where severe weather is located over or just in front of the metering fix and thus this constraint prohibits a flow through the metering fix for the 15-minute time period of the planning time horizon. The performance of both flow-based methods was highly dependent not only on weather severity, but weather cell location. If even a small set of severe weather cells were located directly over the metering fix or over route entry points, then the weather could be sufficient to block all routes connecting to those points.

The flow-based techniques were tested using varying safety margins for hazardous weather avoidance. With a minimum of 0 nmi weather safety margin (i.e. the flight can get as close to hazardous weather as necessary, but may not penetrate it), there were far fewer route closures. The algorithm performed best under this condition. When increasing the minimum weather safety margin, the throughput decreased, as expected, and delays increased.

Figure 31 and Figure 32 provide example output from the alternate waypoint and FBRP algorithms. Across the board, all of the severe weather avoidance algorithms provided relatively similar average flight times per aircraft. The significant differences in performance arise where the algorithm is not flexible enough to avoid the weather. There are subtle differences for this condition to be met in each of the algorithms. Both methods have some flexibility in where the route entry points are; however, a small portion of hazardous weather at the entry point or the metering fix can have enough impact to render the route unusable. For example, from 12:00 to 17:00 on 6/26/02 (Figure 32), hazardous weather sweeps across the southern metering fixes, blocking all routes first from the southwest then from the southeast.

Flow-based methods show weakness when significantly complex (broadly scattered) weather systems prevent a routable solution. For example, during the period from 18:00 to 23:00 on 6/26/02 (Figure 32) widely scattered storms prevent the routing of flows into the southeastern fix, HUSKY. For these situations there are potential routes available to a smaller subset of aircraft, and in such a case, another method should be used to recover this lost capacity, such as the Free Flight method.

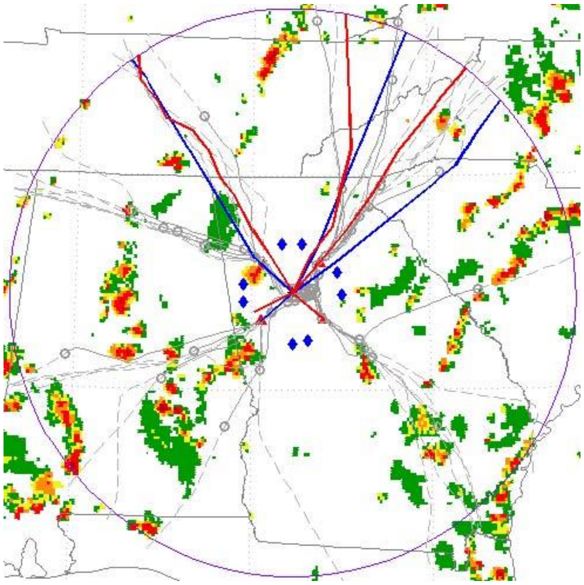


Figure 30. Historical aircraft "picking through" hazardous weather from the south.

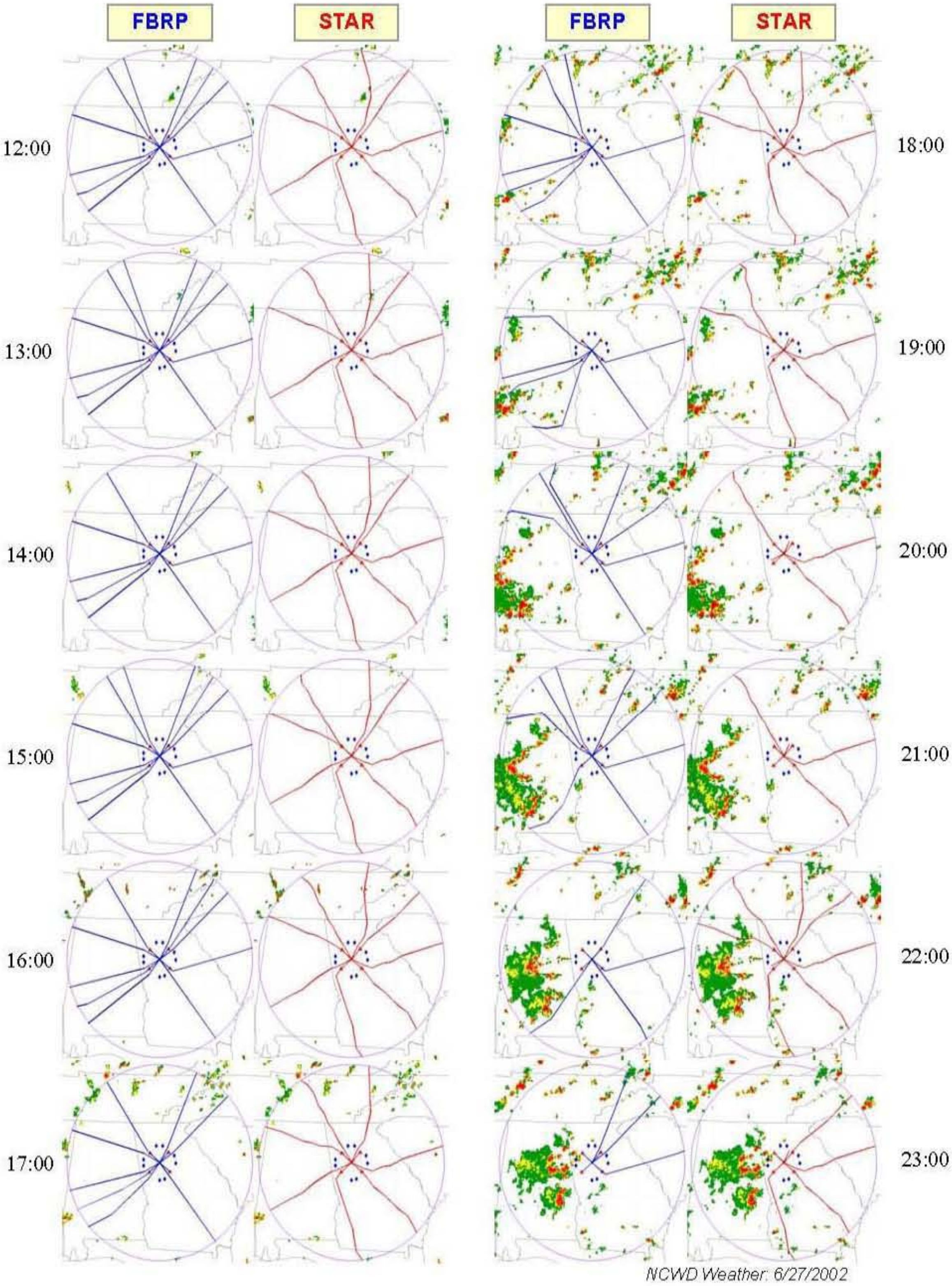


Figure 31. Comparison of Flow-Based Route Planning to variable STARs on 6/27/02.

1-nmi hazardous weather separation; 15-minute route update times.

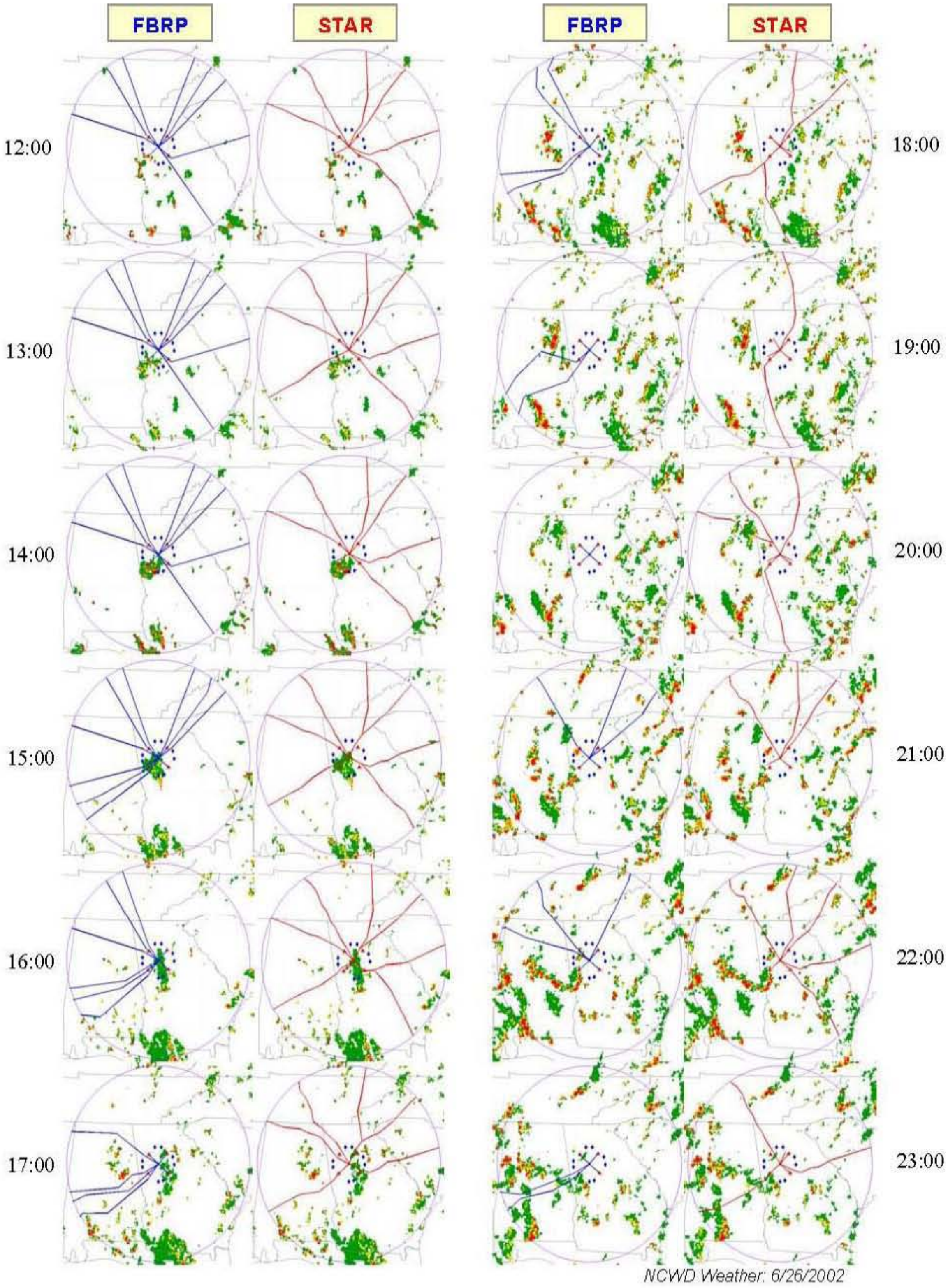
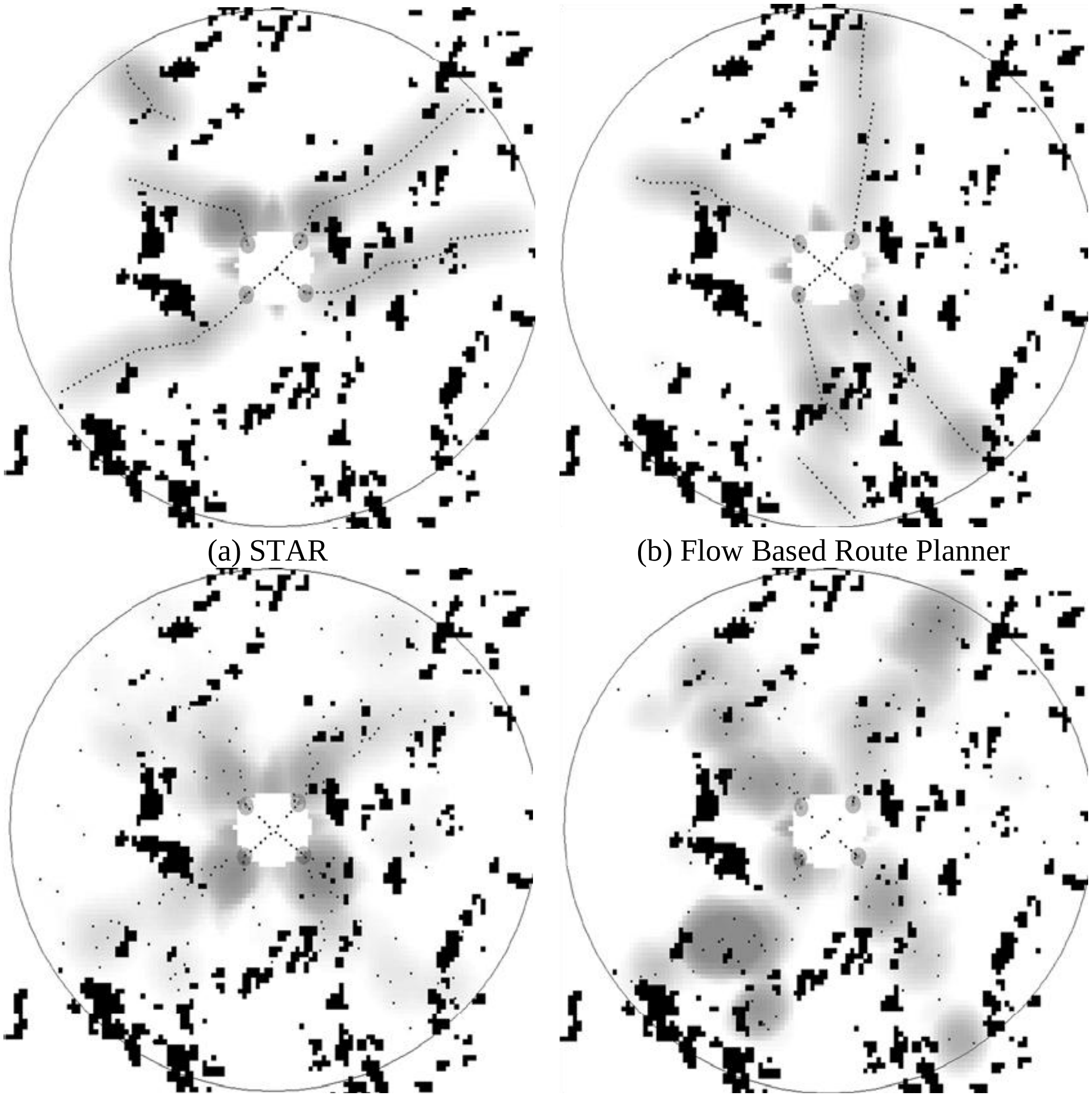


Figure 32. Comparison of Flow-Based Route Planning to variable STARs on 6/26/02.

1-nmi hazardous weather separation; 15-minute route update times.



(a) STAR

(b) Flow Based Route Planner

(c) Free Flight (RTA)

(d) Free Flight (FCFS)

Figure 33. Example complexity metric for maximum throughput rates for each method.

3.7 Conclusion

Three algorithmic routing methods have been presented. Compared to today's routing, these methods demonstrate improved throughput, increased safety, and reduced complexity during hazardous weather events in the transition airspace. While maximum throughput generally decreases with increasing weather severity, the specific location of hazardous weather cells plays an important role as well. The proximity of severe weather to key resources limits throughput, and therefore overall capacity. This is true for today's resources (primarily the airport, arrival and departure fixes, and jet routes), and is also true for conditions within all of the routing algorithms that were included in this study (entry points and metering fixes). Making these resources flexible, predictable, and consistently available is the key to maintaining throughput and increasing capacity during severe weather.

The Flow-Based Route Planner provides more flexibility than the variable STARs and therefore provides greater throughput. When hazardous weather approaches metering fixes, the use of variable metering fixes proves some advantage, as this technique allowed synthesized routes to remain feasible even when the metering fix was impacted by weather.

A tradeoff exists when going from centralized, flow-based routing to a distributed, Free Flight solution. The flow-based techniques are less computationally intensive, as a single synthesized route may apply to many aircraft. Free Flight requires a route to be generated for each aircraft that simultaneously avoids hazardous weather and conflicts with all other aircraft. This means, for example, that instead of creating one route for use by 22 aircraft in a 15-minute period, there would be 22 separate routes, each having to avoid all routes of aircraft that came before it.

To address the computational complexity, such a Free Flight solution may be performed for each aircraft in a distributed manner via a self-optimization implementation. However, there are still limitations to the Free Flight concept itself. In theory, a perfect system-optimized Free Flight solution (i.e. RTA, as opposed to the self-optimized FCFS) will always achieve throughput as good, or better, than a flow-based solution, because there are fewer constraints in Free Flight. However, computing such a system-optimized Free Flight solution requires imposing a RTA on each aircraft (thereby reducing some of the freedom afforded by Free Flight) and involves greater airspace complexity.

Chapter 4: Designing On-Demand Coded Departure Routes[2]

4.1 Introduction

Traffic Flow Management (TFM) initiatives, including the use of Coded Departure Routes (CDRs), are currently used to resolve weather constraints during severe weather events in the National Airspace System (NAS). CDRs provide a combination of coded air traffic routings and refined coordination procedures designed to reduce workload and departure delays for the FAA and NAS users during periods of severe weather or other events that impact the NAS. CDRs are typically used by the Air Traffic Service Provider (ATSP), more specifically the Air Traffic Control System Command Center (ATCSCC) and Air Route Traffic Control Centers (ARTCCs) (otherwise referred to as Centers), in order to predefine a set of possible contingency plans for use when weather introduces complexity to the situation. Furthermore, the naming convention of CDRs facilitates easy communication among the ATSP, Airline Operations Centers (AOCs) and the flight deck for tactically deciding which departure routes should be used to exit a departure airport.

From the ATSP perspective, the identification of the best CDRs to route traffic away from an airport includes the issues related to where the traffic wants to go, constraints imposed by overflights as well as departing or arriving traffic into nearby airports, and being fair to competing airlines for the limited airspace resources. From the AOC perspective, strategically, it is the airline dispatcher’s job to expedite the departure of a flight while ensuring that the flight departs on a route that avoids known hazardous weather and other safety hazards. By preparing a flight for a set of alternative departure routes (CDRs) ahead of time, and letting the ATSP select the final route (with the concurrence of the flight crew), the dispatcher is better able to achieve these objectives.

In the next two sections, we present how CDRs are used today in the NAS, and the current design process for creating CDRs.

4.2 Current Use of CDRs

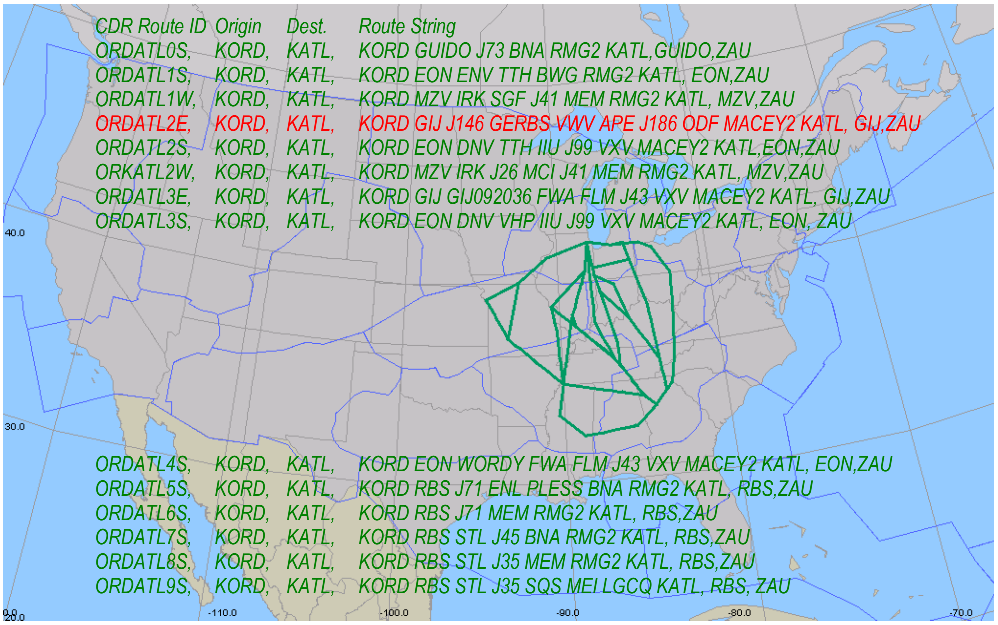
Consider the CDRs between Chicago O’Hare International Airport (ORD) and Atlanta Hartsfield International Airport (ATL), as shown in Figure 34. The FAA provides CDRs through FAA web pages. CDRs define ahead of time a set of routes between two cities that avoid known traffic constraints (such as filing flights from ORD-

[2] The material in this chapter is adapted from J. Krozel, J. Prete, J.S.B. Mitchell, P. Smith, A. Andre, “Designing On-Demand Coded Departure Routes”, AIAA Guidance, Navigation and Control Conference, 2006.

ATL through CVG departure airspace), and assign an abbreviation for identifying such a route The flight deck can type in the code, for instance, ORDATL2E for a route between ORD and ATL, and the definition of the route is known, as it has been pre-coded into the Flight Management System (FMS), and can be discussed between pilot, controller, and AOC with a common short hand notation. For instance, “You are cleared to depart ORDATL2E” is easily verbally communicated from the controller to the pilot.

The Route Management Tool (RMT) was developed to manage and display CDRs and National Playbook Plays, which are similar in concept and use to CDRs, but are applied to deal with larger storm systems. The National Playbook is a traffic management tool developed to give the ATCSCC, other FAA facilities, and system users a common product for various route scenarios. The purpose of the National Playbook is to aid in expediting route coordination during those periods of constraint on the NAS. The National Playbook contains the most common scenarios that occur during each severe weather season and each includes the resource or flow impacted, facilities included, and specific routes for each facility involved [FAA Order 7210.3]. Users of RMT can view the database information and tables of preferred routes, location identifications, and airway intersections. RMT is used by ATCSCC Severe Weather Specialists, all 20 ARTCCs, and the CDM participating airlines.

RMT centralizes the CDR database and allows global changes on a route description element (e.g., SID/STAR numbers and jet route numbers.). Miscommunication on routes caused by database discrepancy is avoided by having a centralized database. In addition, RMT provides route validation capabilities as routes are modified or new routes are defined. RMT users are able to view, search/query, and download the database. In addition, the upcoming 56-day cycle CDR changes are available to users so they can prepare for changes in advance.



CDR Route ID	Origin	Dest.	Route String
ORDATL0S,	KORD,	KATL,	KORD GUIDO J73 BNA RMG2 KATL,GUIDO,ZAU
ORDATL1S,	KORD,	KATL,	KORD EON ENV TTH BWG RMG2 KATL, EON,ZAU
ORDATL1W,	KORD,	KATL,	KORD MZV IRK SGF J41 MEM RMG2 KATL, MZV,ZAU
ORDATL2E,	KORD,	KATL,	KORD GIJ J146 GERBS VWV APE J186 ODF MACEY2 KATL, GIJ,ZAU
ORDATL2S,	KORD,	KATL,	KORD EON DNV TTH IIU J99 VXV MACEY2 KATL,EON,ZAU
ORKATL2W,	KORD,	KATL,	KORD MZV IRK J26 MCI J41 MEM RMG2 KATL, MZV,ZAU
ORDATL3E,	KORD,	KATL,	KORD GIJ GIJ092036 FWA FLM J43 VXV MACEY2 KATL, GIJ,ZAU
ORDATL3S,	KORD,	KATL,	KORD EON DNV VHP IIU J99 VXV MACEY2 KATL, EON, ZAU
ORDATL4S,	KORD,	KATL,	KORD EON WORDY FWA FLM J43 VXV MACEY2 KATL, EON,ZAU
ORDATL5S,	KORD,	KATL,	KORD RBS J71 ENL PLESS BNA RMG2 KATL, RBS,ZAU
ORDATL6S,	KORD,	KATL,	KORD RBS J71 MEM RMG2 KATL, RBS,ZAU
ORDATL7S,	KORD,	KATL,	KORD RBS STL J45 BNA RMG2 KATL, RBS,ZAU
ORDATL8S,	KORD,	KATL,	KORD RBS STL J35 MEM RMG2 KATL, RBS,ZAU
ORDATL9S,	KORD,	KATL,	KORD RBS STL J35 SQS MEI LGCQ KATL, RBS, ZAU

Figure 34. The FAA distributes CDR data which indicate the set of routes that TFM plans on using for weather avoidance out of a particular airport.

To use CDRs a memorandum of understanding (MOU) between the facility and user must exist. If there is a predictable weather constraint, the ATCSCC sends out a reroute advisory specifying that flights for the affected city pair should file a specific CDR. In this case, the AOC files the assigned reroute, which then shows up as the filed route on the flight release that the pilots receive. If this CDR is unacceptable to the flight because of safety or efficiency concerns, the AOC can work with ATCSCC to find an alternative.

If the weather constraint is unpredictable, then the relevant Center or ATCSCC may coordinate with the affected users to identify a set of alternative CDRs that may be used, depending upon how the weather develops. In this case, the AOC prepares the flight ahead of time for this set of alternative CDRs, and includes information on these alternative routes on the pilots' flight release. The relevant Center then makes the determination regarding which CDR to use for a given flight as its departure time approaches (with the concurrence of the flight crew).

4.3 Current Design Process for CDRs

CDRs are developed and updated by the Centers. At the destination airport, the routes must tie into normal arrival routings. CDRs are updated on a 56-day cycle to coincide with the normal chart update. To this end, there are two segments to the CDR database. The first segment is a read only record of all the current CDRs. The other segment is a staging database, which is read-only to the users, but amendable by FAA facilities. On each chart date, the operational database is replaced by the staging database. The new staging database is then opened for changes. The changes must be entered into the staging database at least 36 days prior to the chart data. Thirty-five days prior to the chart date, the staging database is closed. There is then a five day review process for errors in the staging database. The errors are forwarded to the POC at the facility for correction. The route will be deleted from the database if the error cannot be corrected immediately. Thirty days prior to release, the FAA and users are able to view the staging database, allowing them to update their files.

An ARTCC must provide a POC for the ATCSCC to contact regarding CDRs. In addition, it is responsible for creating and validating CDRs prior to inclusion in the database. The validation process is considered complete when all facilities affected by the CDR have been provided a 30-day opportunity to reply to the proposed route, or each impacted facility has been contacted and has approved the route. [FAA Order 7120.3]. Also, the center reviews CDRs created by other centers. It will either validate the originating facility's CDR or provide an alternative routing when appropriate. In addition, the Center is responsible for reporting unusable, inaccurate, or unsatisfactory CDRs to the ATCSCC POC. The report must include the CDR designator, affected sectors, and specific descriptions of the impact, and, if appropriate, suggestion for modification.

4.4 Shortcomings of CDRs

CDRs sometimes offer weather avoidance solutions that work well, and at other times, the CDRs take aircraft far out of the way around a weather constraint. The intent of the research in this chapter is to create CDRs that meet the constraints without

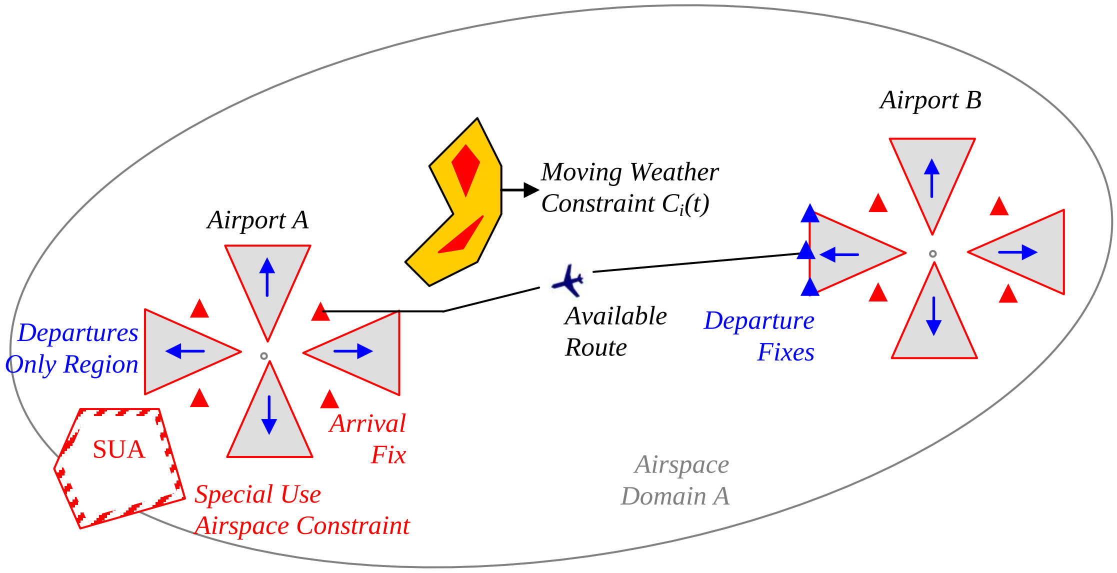


Figure 35. **The problem is defined for two airports and the surrounding airspace defined by an ellipse.**

excessive deviations (costs). Furthermore, we investigate techniques for generating CDRs which will not require the use of Navaids to define the CDRs. In the future as aircraft with Global Positioning System (GPS) navigation and advanced FMS capabilities dominate the fleets, routing solutions can be defined using routing structures that are not constrained by the physical location of Navaids. The solution approach of this chapter is designed to provide solutions under these future conditions.

4.5 Modeling and Problem Statement

The problem can be described as follows. While the geometry is generally described by 3D space, we model the problem in a 2D horizontal plane primarily due to the fact that we plan routes near the airports using standard descent and arrival rates. Furthermore in en route airspace, we do not track the altitude of the aircraft nor control the altitude, so only the 2D horizontal dimension is considered.

Let *A* be the *airspace domain*, a subset of the real plane. Domain *A* consists of the union of two airport regions and the airspace between them as defined by an ellipse (Error! Reference source not found.) with foci at Airport A located at point $a_0 \in A$ and foci at Airport B located at point $b_0 \in A$. At *time t* there exists a set of *constraints, C(t) =* $\{C_1(t),C_2(t),\ldots,C_N(t)\}$, with each connected component $C_i(t) \subset A$, a region of airspace through which aircraft are not safe to fly. Airspace constraints are of various types, including:

- No-fly zones and Special Use Airspace (SUA) (typically stationary, not varying with *t*);
- Departure-only constraints or arrival-only constraints, which specify airspace regions that apply only to departing or to arriving aircraft (constraining departures to avoid airspace designated primarily for arrivals and vice versa);

- Weather constraints, typically growing or decaying weather cells (varying with *t*)
- Weather constraints arise from convective weather severe enough (above an intensity threshold) as to pose a safety hazard for air travel. While the criteria for weather avoidance depends on the pilot and airline guidelines, research shows that pilots generally avoid National Weather Service (NWS) Level 3 and higher weather cells (these are yellow, red, or more hazardous weather cells on the standard weather map).

A *horizontal separation requirement* δ (5 nmi) is a constraint that requires two aircraft to be horizontally separated by a distance δ at all times. A *vertical separation requirement h* requires two aircraft to be vertically separated by a distance *h* (1000 ft) at all times. Because of the horizontal separation requirement, CDRs are designed to be at least 2δ away from each other at all times, and CDRs are never designed to cross over each other.

Two types of terminal conditions are considered:

- CDRs from airport departure fixes to airport arrival fixes
- CDRs from pitch points to catch points

Arrival and departure fixes are located in the terminal area of the airport, roughly 40 nmi from the runways. These fixes define the initial and final conditions of the CDRs. Alternatively, we define pitch and catch points for the terminal conditions. The pitch point is the entry into the set of CDRs, and the catch point defines the exit out of the set of CDRs. A terminal area routing tool may be used to route traffic to and from the pitch and catch points. In our work, the pitch and catch points are represented by line segments rather than points in space.

4.6 Algorithm

The algorithm to compute CDR routes is based on the FBRP algorithm, which is described in the first chapter. The order in which CDR routes are computed depends on the priorities assigned to the source/destination pairs. As each route is computed, it becomes a constraint for routing subsequent routes, which collectively become a family of alternative CDR routes.

4.7 Results

Examples below show cases that span from no weather, medium weather, to hazardous weather. The first set of results show algorithmic routing solutions from a departure fix to an arrival fix, first for one pair, then for each of the others, in order of increasing total path length. The horizontal separation standard between CDRs is 10 nmi. The time window is 2 hours. The weather is shown thirty minutes into the 2-hour window.

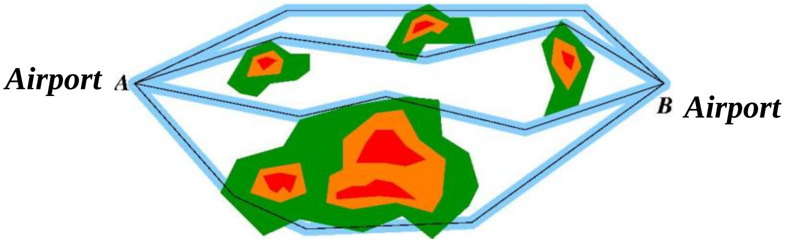
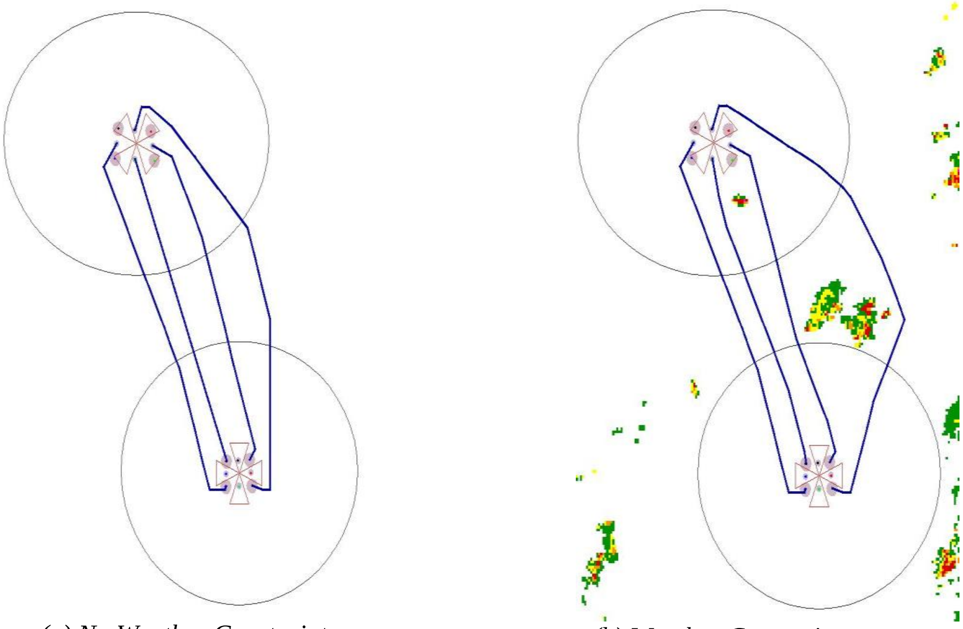


Figure 36. Weather avoidance routes between Airport A and Airport B.

4.8 Human Factors Issues

In the current NAS, when the weather constraint is predictable, the use of CDRs as reroutes reduces the workload for ARTCC traffic managers, as the dispatchers at the different AOCs file the correct route rather than relying on a traffic manager to make a route amendment. It also reduces computer entry and communication workload, as a given CDR's abbreviation can be used to access it in the computer and to simplify voice communications between pilots and controllers.

If the weather constraint is less predictable, then a set of alternative CDRs are identified as contingency plans to mitigate the various weather scenarios that are most likely to materialize. Traffic managers and AOCs communicate ahead of time to identify these contingencies (alternative CDRs), sharing their knowledge and data. For each flight the responsible dispatcher then determines whether it can be preplanned to accept all of these CDRs. This is done by the dispatcher 60-75 minutes before departure as part of the normal flight planning process. The approved CDRs are then included on the flight release for the pilots. As the aircraft approaches its departure time, a traffic



(a) No Weather Constraints

(b) Weather Constraints present.

Figure 37. On-demand CDRs between Airport A and Airport B.

manager at the responsible ARTCC evaluates the situation and assigns the final departure route. This is transmitted to airport Tower ATC, which uses it for the flight's departure clearance. Assuming the offered departure route (CDR) was listed as approved on the flight release, the pilots can then accept this clearance if they judge it to be safe at that time.

The benefit of the above process for dealing with unpredictable weather is that it keeps everyone in the loop (traffic managers, dispatchers, controllers and pilots). It also allows them to ensure safe, efficient operations, letting them plan ahead and develop acceptable contingencies, while also providing the flexibility to deal with the uncertainties in the weather. However, there are costs as well. For example, manually determining the suitability of various CDRs for a given, unpredictable weather situation is a difficult and potentially time-consuming task, and one often based on qualitative heuristics or memory. So is the task of prioritizing the potentially-relevant CDRs or determining the optimal CDR. Finally, while communication and coordination take place between the various NAS users, it does so without the aid of modern collaborative tools and technology, perhaps leading to less-than-equal levels of situation awareness among the personnel involved.

In our operational concept, we propose to continue the currently successful and collaborative process for contingency planning and coordination in the context of CDRs, but assume that this collaboration will be accomplished with the aid of advanced technology for sharing voice, video, and text among disparate team members (akin to Web conferencing and application sharing). Additionally, our concept assumes use of recent and projected advances in DST and user interface technologies that should mitigate some of the known human factors costs of the current process in the following ways:

- Advanced weather avoidance algorithms, such as those presented in this chapter, would provide greater flexibility in defining a unique and optimized

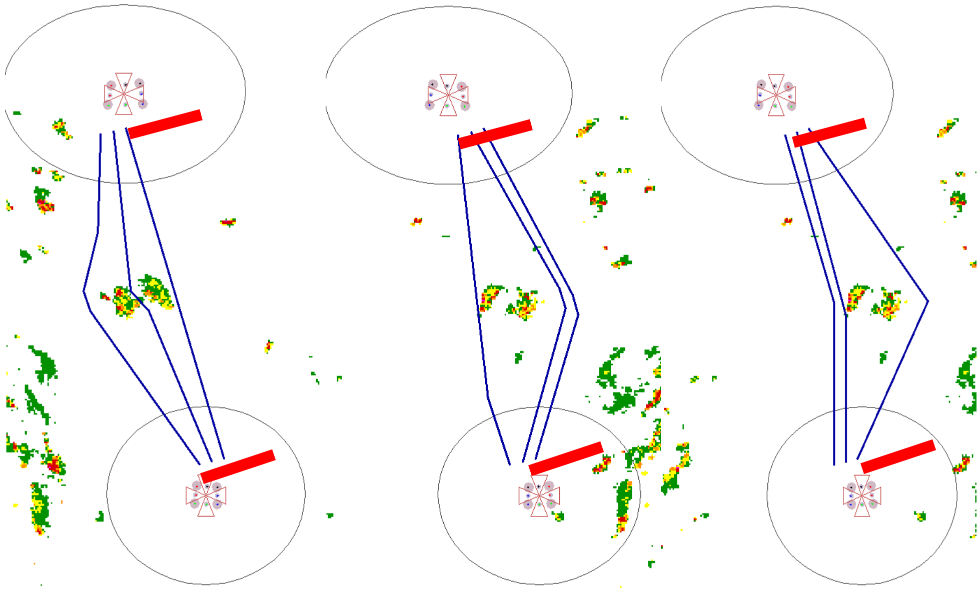


Figure 38. On-demand CDRs between pitch points outside Airport A and catch points outside Airport B.

set of CDRs to deal with a given weather situation while taking into consideration known traffic constraints. This dynamically-generated set would then be used for consideration instead of the current static set of CDRs.

- Use of a computer automation system, such as the aforementioned DST, would provide the traffic manager with the ability to place constraints on the generation of CDRs, evaluate (play out) the resultant CDR or CDR set, to compare CDRs, to view historical data on the prior success and failure of specific CDRs for a given weather constraint, and to modify CDRs as needed.
- A similar tool would be used by airline dispatchers to evaluate CDRs in terms of flight priority/scheduling, safety/comfort and equitable treatment.
- Access to various information about the current and predicted weather and traffic, the weather avoidance algorithms, the CDRs dynamically generated by the algorithms, and guidance on selecting the best CDR for a given situation would be provided through a single, integrated DST, negating the need for these activities to be performed manually or through separate systems.

While we are optimistic about the role of our weather avoidance algorithms in the context of future traffic planning DSTs, there are some significant human factors considerations that arise with this concept. The first consideration arises because it is not sufficient for flexibly defined CDRs to merely avoid the predicted weather. Like current CDRs, they must also be designed to avoid traffic bottlenecks. Since the knowledge necessary to introduce such traffic constraints into the algorithm is quite complex and extensive, it will be very important to treat the generated routes as recommendations, which must be evaluated by the appropriate traffic managers for acceptability. Initially, this may introduce significant extra workload. Note also that there is still a coordination issue (and therefore an associated coordination time issue) associated with this process. For a given algorithmically generated CDR, traffic managers from several Centers may have to complete this evaluation. (Clearly the traffic managers will need to be able to manually edit or replace these algorithmically generated CDRs as well.)

We assume that these tasks will be aided by the functionality of the DST and through designing the DST with high levels of interface transparency, allowing the user to access, understand and perhaps edit the logic and intentions of the automated CDR generation and recommendation process. Through incorporation of human factors principles to the design of these DSTs, we would expect that the traffic managers will, over time, begin to recognize patterns in the flexible CDRs that are generated and may therefore be able to quickly look at an algorithmically generated CDR and accept or reject it. Ultimately, they may become sufficiently comfortable with the rules or constraints underlying the generation of these CDRs (that the system design provides them access to), and be willing to let a single traffic manager assess the acceptability of the generated routes at a more cursory level in terms of traffic considerations.

The second human factors consideration has to do with air traffic controller performance. More flexibility in the definition of CDRs means more variability in the traffic patterns confronting controllers, potentially limiting their ability to easily monitor for typical confliction points in their assigned sector. Two possible solutions can deal with this issue. First, the constraints introduced to limit CDR generation should ensure that the complexities introduced by flexible CDRs are not too great for controllers. To this end, future DSTs should incorporate controller workload as a constraint in the CDR generation process through incorporation of workload estimation algorithms. Second, we anticipate that similar technological and automation-based advances to air traffic controller tools will make it easier for controllers to deal with greater variability in traffic patterns.

Finally, from an AOC perspective, an important consideration arises if the algorithms for generating flexible CDRs consider only weather and traffic constraints. AOCs are also looking for economical routes that are safe for their aircraft. These latter considerations require an assessment of routes using knowledge specific to each NAS user. In addition, the greater variability in the CDRs generated will increase the workload associated with evaluating and comparing them. Thus, AOC flight planning systems will need to be able to input the set of recommended flexible CDRs for a given situation and provide the dispatcher with the data and displays necessary to quickly and easily evaluate that set in terms of safety and efficiency.

4.9 Conclusion

This chapter investigates on-demand Coded Departure Routes (CDRs) where the CDRs are defined 1-2 hours ahead of a severe weather event and provide routing that avoids the weather constraint without excessive rerouting. An algorithmic solution is provided that identifies a set of CDRs that avoid severe weather constraints for the next 1-2 hours, and do not intersect each other. Human factors issues are taken into consideration in our design, but require further study in future development of this operational concept.

Chapter 5: Capacity Estimation for Level Flight with Convective Weather Constraints[3]

5.1 Introduction

A fundamental problem in Air Traffic Management (ATM) is to estimate the capacity of an airspace given a weather forecast indicating convective weather constraints. If the demand of an airspace exceeds its capacity, then a traffic flow management control strategy for that airspace is necessary. The demand of an airspace is determined by the number and type of aircraft that desire to fly through the airspace within a particular time window. The capacity of an airspace is defined as the maximum number of aircraft per unit time that can be safely accommodated by the airspace, given controller and pilot workload constraints and airspace constraints (e.g., Special Use Airspace, convective weather constraints, etc.).

The primary focus of this chapter is to investigate the tradeoffs between the airspace capacity and ATM control laws in the presence of weather constraints. We compare a set of ATM control laws that span decentralized techniques, such as Free Flight [RTCA], to centralized control techniques, such as Flow-Based Route Planning (FBRP) [PM, KPPM1]. Furthermore, we examine the effects of platooning of aircraft within these ATM control laws, ranging from platoon sizes of 1 (individual aircraft) through very large platoon sizes that approach continuous flows of aircraft following the same waypoints across the airspace. At constant flight level, a platoon is a set of aircraft, all of which are flying the same route (waypoints), while separated by a longitudinal Miles-In-Trail (MIT) requirement. For the purpose of establishing the maximum capacity, we set the MIT requirement to be the en route separation requirement of 5 nmi; however, if our algorithms are to be used for current ATM applications, a MIT requirement of 7, 10, 15, or 20 nmi may be appropriate.

This research has been conducted to support the Next Generation Air Transportation System (NGATS) [SBL]. Tradeoff studies in this chapter compare fundamental ATM control laws that are not dependent on today's jet routes or current ATM practices. The study will help NGATS policy decision makers to choose between candidate ATM designs, and helps researchers understand the relationships between ATM control laws, capacity, and traffic complexity.

In today's National Airspace System (NAS), there is a need to establish the estimation of capacity of an en route airspace to support Airspace Flow Programs (AFPs) [KJP, B] and to establish Flow Constrained Areas (FCAs). While the techniques presented in this chapter do not depend on jet route structures that affect today's AFPs

[3] The material in this chapter is adapted from Krozel, J., Mitchell, J.S.B., Polishchuk, V., Prete, J., "Capacity Estimation for Level Flight with Convective Weather Constraints", submitted to *Air Traffic Control Quarterly* (est. July 2007).

and FCAs, our results do represent upper bounds that are useful for understanding the effect of hazardous weather constraints on AFPs and FCAs in the NAS. Today, there is no standard theoretical approach for establishing the capacity of a particular en route airspace in the NAS, particularly when impacted by severe weather; existing approaches are empirical. To this end, the theory and algorithms presented in this chapter break new ground.

5.2 Modeling

5.2.1 Model of the Airspace

We consider a region of airspace specified by a two-dimensional (2D) polygonal domain *P*. Our experiments use a square airspace, however, all of our techniques apply immediately to general polygonal en route airspaces (e.g., sectors, centers, or FCAs). For synthetic-weather experiments the size of the square that represents the airspace is 60-by-60 nmi; for real weather experiments the size is 200-by-200 nmi, in order for the airspace to be large enough to exhibit intricate patterns of weather. One portion of the airspace boundary serves as the *source* while another portion serves as the *sink*. We restrict all flights to enter at the source and exit the airspace at the sink. We do not allow flow to originate or to terminate within the airspace (i.e., no aircraft exit or enter the flight level within the interior of *P*).

Several assumptions are made. All aircraft are assumed to have a constant speed of 420 kn in our experiments. (However, the software allows any speed, and the speeds may vary by aircraft.) The aircraft horizontal separation requirement is 5 nmi for the en route airspace. Thus, at peak throughput, a single lane of traffic can carry 84 aircraft/h past a particular point in space. We assume that aircraft can come arbitrarily close to hazardous weather as long as they do not enter it; our algorithms readily permit safety margins to be added to the hazardous weather regions, but our experiments used a margin of zero. We ignore the earth's curvature, since it is not a significant factor over relatively small experimental areas.

5.2.2 Convective Weather Constraints

We consider constraints that arise from convective weather. Convective weather severe enough to pose a safety hazard for aircraft is often characterized by specifying an intensity threshold in the National Weather Service (NWS) scale. While the criteria for weather avoidance depend on pilot preferences and airline guidelines, research [RP] shows that pilots generally avoid NWS Level 3 and higher weather cells (greater than 13.3 mm/hr rainfall or reflectivity greater than 41 dBZ). While the altitude of cloud tops in severe storms is also an important factor [DE] that pilots consider in determining which storm cells to avoid, cloud tops data are not included in our algorithmic experiments.

The Weather Severity Index (WSI) for en route airspace is defined as the percentage of the airspace that is occupied by convective weather with NWS level 3 or greater. We acknowledge that the structure of weather cells, not just their number, can have a significant impact on the throughput [MPK]. When generating synthetic weather datasets, we generate weather cells according to a common distribution, across all severity levels, so that the structure of cells can be expected to be similar even as severity is varied. All weather used in our experiments is static. (Note, however, that the FBRP routing methods we employ do apply to the dynamic weather constraint environment, as shown in [PM].)

5.2.2.1 Synthetic Weather Generation

Our experiments use synthetic weather, randomly generated in order to have a wide variety of coverage (WSI). To generate random "popcorn" weather, we employ a simple model in which random circular weather cells are generated within a larger airspace (specifically, within an 80-by-80 nmi square centered on the 60-by-60 nmi airspace of interest). In our simple model, each formation's center is uniformly (independently) distributed in the region; the sizes vary between 2 to 6 nmi in radius, according to a symmetric triangle distribution with a peak at 4 nmi. Weather cells are randomly added or removed until the measured WSI is within a small percentage of the desired value. For each WSI value from 0% to 70% in 5% increments, ten weather samples are generated. No experiments were run for WSI over 70%.

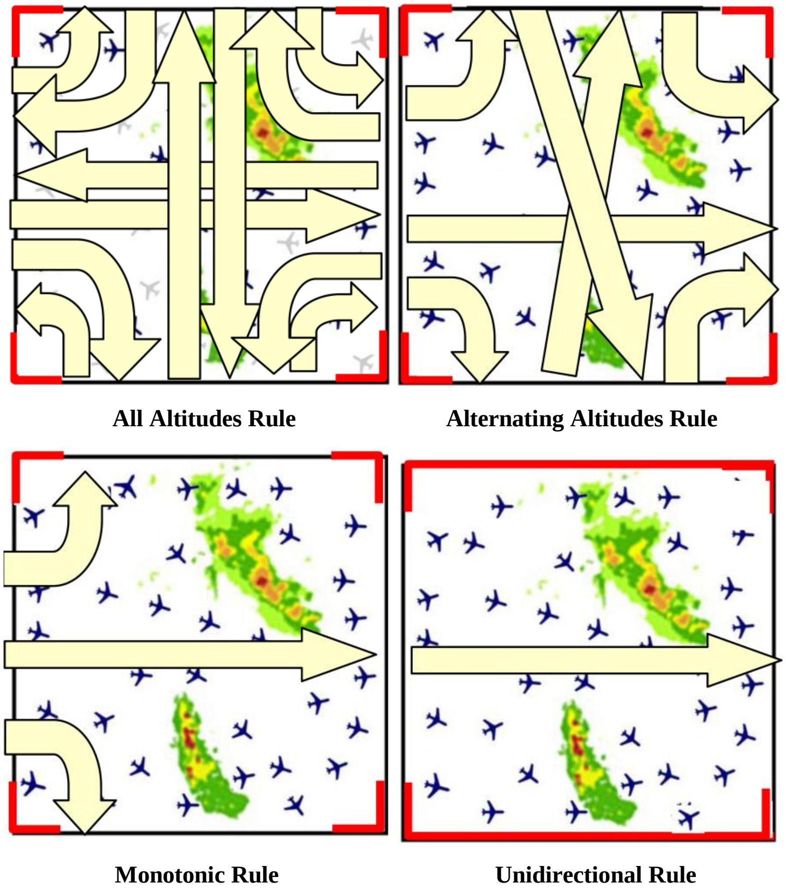
5.2.2.2 Real Weather Extraction

We also conducted experiments on real-world weather data using our capacity estimation methods. We use samples that are roughly 200-by-200 minutes (of latitude/longitude). (In order to approximate more closely a square region, we slightly stretch the samples east-west to cover a full 200 nmi). The data samples used in the experiments come from NWS data for certain time slices on severe weather days: June 26, 2002 and June 27, 2002. These time slices have been subjectively selected as the most hazardous weather instances from those particular days. In order to extract multiple samples of weather data, we shift a 200-by-200-minute sampling region across the US by 100-minute steps. The WSI is measured for each sample, allowing us to create a table of many instances of weather at various WSI values.

Real weather at 200-nmi scale does not normally have very high WSI, even though it may represent a significant navigational constraint. In order to obtain a wide variety of weather samples with high WSI values, we examine the weather data twice, once using a NWS level 3 threshold, and once using a NWS level 2 threshold. In this way, we are able to extend our results to realistic weather samples with WSI values as high as 34%.

5.2.3 ATM Control Laws

We examine four different ATM control laws in order to determine how capacity and complexity are affected by the level and type of control. The rules are illustrated in Figure 1. Each rule represents a different tradeoff in capacity, freedom of routing, and human factors complexity for air traffic controllers. In order to avoid corner-clipping cases, in our simulations we require flights to enter and exit the airspace within the central 75% of each side of the airspace; this constraint is indicated with thick red lines in Figure 1. In order from least restricted to most restricted, the ATM rules are:



All Altitudes Rule

Alternating Altitudes Rule

Monotonic Rule

Unidirectional Rule

Figure 39. Four ATM control laws.

- *All Altitudes (AA).* Flights cross the airspace in any direction, entering on any side and exiting on any side as long as the entry and exit sides are different.
- *Alternating Altitude Rule (AAR).* Flights follow the standard rule that east-to-west aircraft are altitude separated from west-to-east aircraft according to “east is odd, west is even”. Our experiments simulate flights only according to the west-to-east rule; the east-to-west flights behave similarly and independently. Any flight passing through the airspace must have its exit point to the east of its entrance point, guaranteeing a (generally) west-to-east traversal, and must enter and exit distinct sides of the airspace.
- *Monotonic Rule (MR).* Flights must enter the airspace on one side (the west) and may exit at any point on the other three sides; thus, as with AAR, flights are required to be monotonically eastward, but, in the MR case, they must enter the airspace on one specific side (the west side).
- *Unidirectional Rule (UR).* Flights must enter the airspace on the west side and exit the airspace on the east side.

5.2.4 Decentralized Free Flight versus Centralized Packed Demand

In our experiments we investigate demand sets for both decentralized Free Flight scenarios and centralized Packed scenarios. In Free Flight scenarios, flights may enter and exit the airspace at arbitrary points on the boundary of the airspace, as long as the ATM control law of the experiment is obeyed. In Packed scenarios, flights must enter and exit the airspace at predetermined points, specially spaced along the boundary of the airspace, based on a calculated route packing designed to maximize throughput. By properly spacing the entry/exit points, one can avoid having many wasteful gaps between air lanes.

5.2.5 Metrics of Comparison

Airspace capacity and the traffic produced by our algorithms are analyzed with two quantitative measures: throughput and airspace complexity. The throughput and complexity metrics are defined in chapter one. The capacity estimation research produces lanes which, in actual use, may or may not be full; since we are estimating maximum capacity we assume that all lanes are in fact full of aircraft. In current practice, aircraft would never be packed so tightly for reasons of safety and maneuverability.

5.3 Theory of Capacity Estimation

In this section we investigate the theoretical capacity of an airspace in which there are given deterministic weather constraints.

5.3.1 Flows in Discrete Networks

First, we review some basic definitions and facts about network flows. A *network* is a directed graph *G=(N,A)*, where *N* is the set of *nodes* and *A* is the set of (directed) *arcs* connecting certain pairs of nodes; each arc *e* has a *capacity*, *c(e)*. Two nodes, *s* and *t*, in *N* are designated as the *source* and *sink*, respectively; all other nodes of *N* are *internal nodes*. A *flow* in *G* is an assignment of a *flow value*, *f(e) ≤ c(e)*, to each arc *e* in *A*, such that the total flow into each internal node is equal to the total flow out of it. The *value* of the flow is the total flow out of *s*; by flow conservation, this is also the total flow into *t*.

The *maxflow* problem is to find a flow with maximum value. A *cut* in *G* is a partition of the nodes into two sets *S* and *T*, such that *s* is in *S*, and *t* is in *T*. An edge *e* is said to *cross* the cut if one of its endpoints is in *S*, and the other is in *T*. The *capacity* of a cut is the sum of the capacities of the edges that cross it; no flow in *G* can possibly have a larger value than the capacity of any cut. A *mincut* is a cut of minimum capacity, and therefore its capacity is an upper bound on the value of any flow. It is a basic fact in optimization that the value of a mincut equals the value of a maxflow; this “maxflow/mincut” theorem is a consequence of *duality* in linear programming [AMO]. Furthermore, efficient (polynomial-time) algorithms are known for computing maximum flows and minimum cuts.

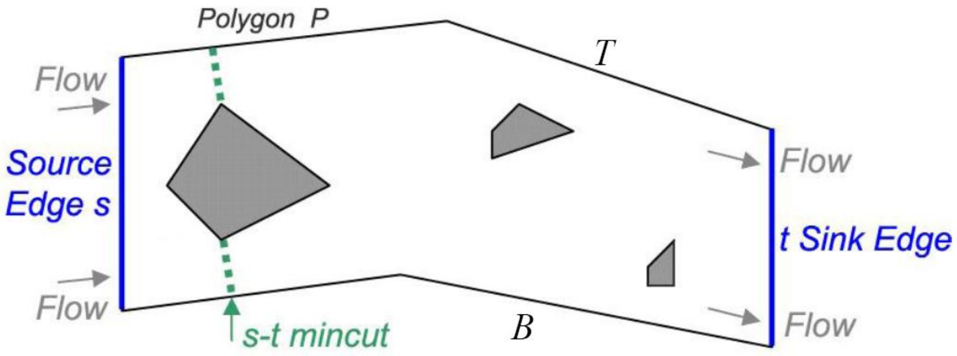


Figure 40. Theoretical capacity of a continuous flow field is determined by the *s-t* mincut.

5.3.2 Continuous Flows with Deterministic Constraints

5.3.2.1 Notation and Structure of Continuous Flows

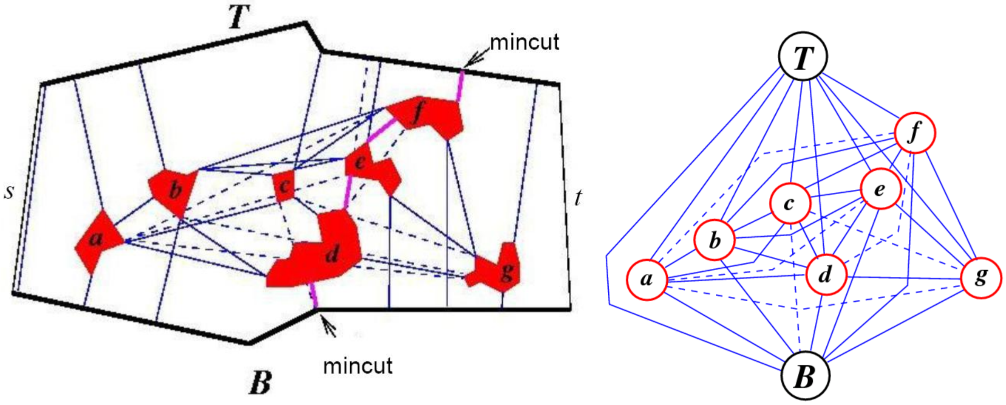
The notions pertinent to the discrete network flows can be extended naturally to flows in 2D domains. Instead of a discrete network, a continuous domain, such as a simple polygon *P*, is considered (Figure 2). Two boundary edges, *s* and *t*, of the polygon are designated as the source and the sink. A *flow f* in *P* is a vector field. The *constraints* $H_1 \ldots H_K$ are pairwise-disjoint simple hazard polygons that lie fully inside *P*; the flow is not allowed to pass through any of the constraints, i.e., for any point *x* within a constraint, $f(x)=0$.

The polygon *P* is assumed to be uniformly capacitated, i.e., the length of the flow vector must nowhere exceed 1. The *value* of the flow is defined as an integral of the normal component, $f \cdot n$, over *t*. There are no sources or sinks inside *P*; i.e., for any *x* in *P*, div $f(x) =0$.

The maxflow problem is to find an s-t flow of maximum value. A cut in P is a partitioning of the polygon into two parts so that s is in one of the parts, and t is in the other. The capacity of a cut is the length of the boundary between the parts, where only the part of the boundary that is interior to P (and not on the boundary of P or within a constraint) is included in the length. We use the term “mincut” to refer to the path(s) within P that comprise the boundary of the cut (shown as the dashed paths in Figure 2), as well as to refer to the capacity (length) of the mincut.

The maxflow/mincut theorem holds for polygonal domains [S2], as it does for discrete networks. [M2] developed geometric shortest path techniques to compute the maxflow and the mincut efficiently in 2D polygonal domains, even if there are multiple source and sink edges on the boundary of the domain. In this chapter, we are concerned with computing a mincut in a polygonal domain, since, as we show, it represents the *maximum* theoretical capacity of an airspace with respect to a given set of weather constraints.

5.3.2.2 An Algorithm to Compute a Mincut



(a) Continuous domain **(b) Graph representation**

Figure 41. A shortest *B*-*T* path in the critical graph defines the capacity (mincut).

The source and sink edges, *s* and *t*, split the boundary of *P* into two polygonal chains, which we denote by *B* and *T* (Figure 65). If *s* and *t* represent the west and east boundaries, then *B* and *T* are the bottom and top (including the small corner exclusion zones).

The *critical graph* of the airspace has a vertex for each constraint, for *B*, and for *T*. The critical graph joins a pair of vertices with an edge whose length is equal to the minimum (Euclidean) distance between the constraints corresponding to the vertices. An example is shown in Figure 3. Dashed edges in Figure 3 correspond to pairs of constraints for which the minimum distance is achieved by a line segment that passes through other constraints; such segments are not necessary in the critical graph, as they will never be part of a mincut (as seen by a simple application of the triangle inequality). The mincut corresponds to a shortest *B*-*T* path in the critical graph. [M2] and [GMMN] showed how to use computational geometry techniques to compute a mincut (and a corresponding maxflow) more efficiently than naively constructing the critical graph and searching it; however, those techniques require a more complex implementation, which we do not do here.

5.3.2.3 Flows with RNP Requirements

In our ATM model, the polygon *P* represents the airspace, *s* and *t* represent the edges through which the aircraft may enter/exit *P*, and the constraints correspond to hazardous weather. The modeling of weather constraints may be in any form (polygons, grid cells, or circles).

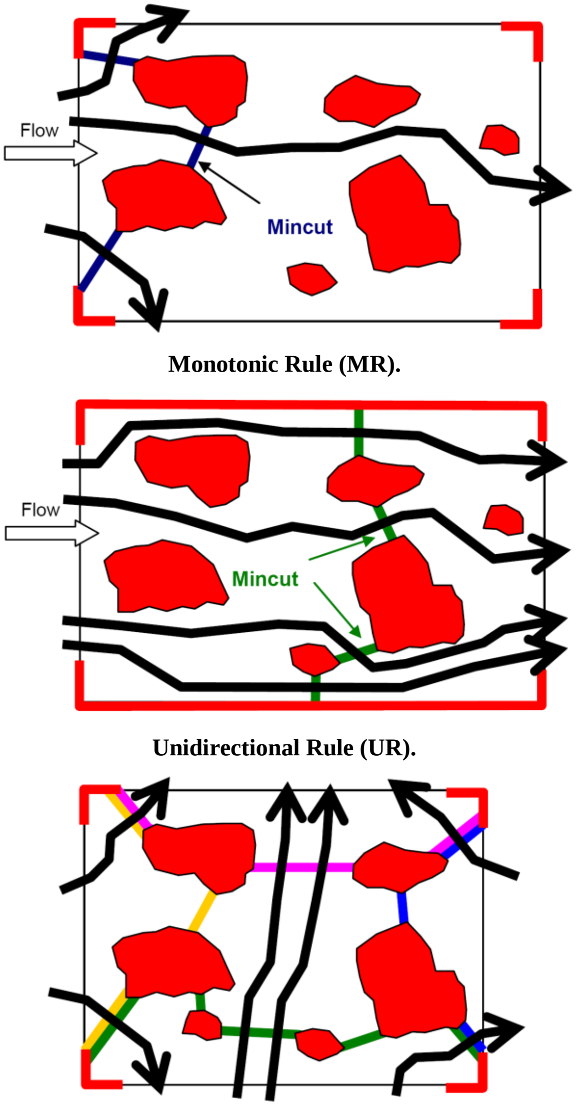
Each air lane is thought of as a *thick* path, where the thickness of a path equals the RNP. The problem of computing the maximum number of air lanes from *s* to *t* through *P* is that of searching for the maximum number of *thick* paths that can be threaded though the airspace from *s* to *t*. This problem is closely related to the continuous maxflow/mincut problem; however, there is an important distinction due to the discrete nature of routing an *integral* number of air lanes. As with the continuous maxflow/mincut computation, we can write the problem as a shortest path problem in the critical graph defined previously; however, before computing the shortest path in the critical graph, the length of each edge is rounded to $\lceil l_{ij}/RNP \rceil$, where l_{ij} is the distance between constraints i and j, and the upper brackets denote rounding up to the nearest integer. Rounding edge lengths to a multiple of RNP reflects the fact that only the *integer* number of air lanes, equal to the rounded length of an edge, may be routed through the edge while meeting the RNP requirement.

5.3.3 Capacity as a Function of the ATM Control Law

In the maxflow/mincut discussion above the traffic is allowed to enter the airspace only through the west side, and exit only through the east. This corresponds to computing the mincut appropriate for the Unidirectional Rule. Refer to Figure 4 (a).

For the other ATM control laws, the mincut is defined slightly differently. For the Monotonic Rule, the bottom B and the top T used in defining the critical graph correspond to the L-shaped (artificial) constraints placed in the northwest and southwest corners of the airspace in order to restrict aircraft from entry/exit very near the corners. See Figure 4 (b). The corresponding mincut represents a theoretical upper bound on the number of air lanes that can be routed according to the Monotonic Rule, entering the airspace between the L-shaped corner constraints.

For All Altitudes and for the Alternating Altitudes Rule we must proceed differently in obtaining a theoretical upper bound, since flights may enter or leave the airspace through any of the four sides. First, we compute the mincuts associated with each of the four sides of the airspace, between each pair of consecutive corners. In Figure 4 (c), the corners are labeled A, B, C, and D, and mincuts are shown connecting pairs of consecutive corners; we let x_{AB}, x_{BC}, x_{CD}, and x_{DA} denote the respective lengths of these cuts. (Note that two distinct mincuts may partially coincide.) Under our assumption that flights are not allowed to enter and exit through the same side, each flight must cross *two* of the mincuts. Thus, an upper bound on the number



Monotonic Rule (MR).

Unidirectional Rule (UR).

All Altitudes or Alternating Altitudes Rule (AA or AAR).

Figure 42. Mincuts and flows under each ATM control law.

of air lanes is given by ½(x_{AB} + x_{BC} + x_{CD} + x_{DA}). This value is the theoretical maximum throughput for both the AA and the AAR rules.

5.4 Algorithmic Solution Approaches

An important feature of the mincuts that we compute is that they give *tight* estimates of the maximum capacity under each ATM control law. This means that, from maxflow/mincut theory, we know that the mincut is equal to the number of air lanes that can actually be routed through the airspace. Our experiments compute the mincut values (exactly); from the theory, we know that we could implement an algorithm that achieves a routing of the mincut number of air lanes.

While mincut algorithms provide a hard theoretical upper bound on the capacity of a region of airspace, this bound is tight only if several assumptions hold: all aircraft are routed in non-crossing lanes, all constraints are *static* (not changing in time), and all aircraft are centrally controlled to avoid conflicts. In some cases, one wants the option to use crossing lanes of traffic in order to satisfy demand. For example, during the passing of a densely formed group ("platoon") of north-south flights, east-west traffic may be blocked; however, after the group passes, there is an opportunity for east-west demand to be met. In high-density situations, flight paths can also block each other from making optimal use of the bottlenecks in an airspace. For this reason, we sought an experimental method for estimating the practical capacity of a region of airspace.

In our experiments, we chose to use the FBRP routing algorithm of the first chapter to determine routes, realizing that, due to its heuristic method of incrementally adding routes (described below), it is not guaranteed to achieve the theoretical maximum throughput. We chose to use the FBRP, though, because it is a more general routing tool than is known theoretically for routing maxflows. In particular, FBRP is capable of routing flows in the presence of *moving* weather constraints and of imposing certain turn and heading constraints on routes. It is also capable of routing small groups of aircraft as "flows", or single aircraft (as in Free Flight). These features of FBRP make it more applicable to the ATM domain than the relatively limited theoretical algorithms ([M2]) that achieve maximum throughput, since those algorithms apply only to *static* weather constraints and do not apply to flows with heading constraints.

5.4.1 Estimating Capacity with the FBRP Algorithm

The FBRP algorithm is an incremental algorithm that computes *flows* (routes that are available for a specified window of time) between specified start and end points. It computes each route in succession while avoiding hazardous weather and all previously-routed aircraft. Technical details of the algorithm are given in the first chapter. For our experiments, we added an elliptical constraint to avoid excessive rerouting: Specifically, we require each route to stay within an ellipse whose foci are the start and end points and whose summed distance is equal to the Euclidean distance between the foci, plus a parameter E (E= 10 nmi for synthetic weather scenarios and E= 30 nmi for real weather scenarios).

The demand across the airspace is determined by entry/exit points on the boundary of P. For both the Free Flight and the Packed cases, we first calculate a set of unblocked points along the boundary, at 0.1 nmi intervals; we then eliminate points in order to achieve the desired spacing of points, usually a minimum of 5 nmi between points.

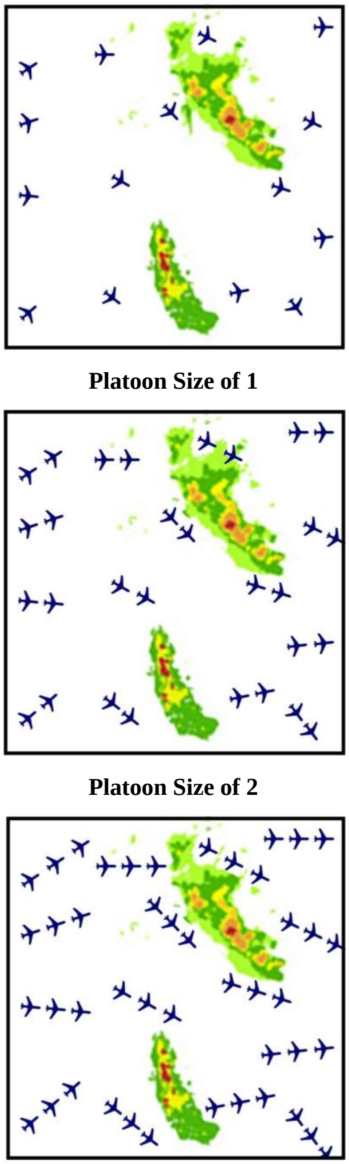
For decentralized Free Flight scenarios, we experimentally measure airspace capacity by randomly generating very large amounts of demand until it is no longer possible to route more aircraft across the airspace. We start at the beginning of the time window for the scenario, iterating through possible timestamps until we reach the end. (The number and frequency of chosen timestamps are based on the platoon size.) For each timestamp, we generate random demand according to the ATM control law in use, trying to route it using the remaining free airspace, until 40 demand requests in a row have failed. All unblocked points in that time slice have an equal chance to be chosen as entry or exit points, depending on the ATM control law in effect. For any given timestamp, the same entry point is never used twice.

For centralized Packed scenarios, we use the set of unblocked candidate entry/exit points somewhat differently. For the All Altitude and Alternating Altitude Rule, we pair off points starting from the corners (all four simultaneously) and working our way towards the center; all of these routes cut across a corner, and routes closer to the corners have priority over those further away. Once we can no longer pair off corner points, we pair off the remaining candidate entry/exit points directly across the sector. For Unidirectional and Monotonic Rules, we calculate the set of points along the appropriate entrance and exit sides, such that we have an equal number of entry and exit points, the entry points are all equally spaced, and the exit points are all equally spaced (these can be different spacings, and usually are). The points are then paired off from the center outwards, giving the center routes priority over the border routes. While there is no guarantee that the produced routes will achieve the theoretical maximum, they come close, as will be seen in the results, and the FBRP is capable of maintaining certain operationally useful constraints on the generated routes, such as the number of turns, the monotonicity with respect to prescribed directions, etc. We examine the resulting routes computed and directly measure throughput and complexity of the airspace.

5.4.2 Platooning

A *platoon* is a set of aircraft, all of which are flying the same waypoints, separated by the MIT requirement. This means that a single solution to the routing problem is reused by multiple aircraft. Platoon sizes in our experiments (e.g., as illustrated in Figure 5) range from one aircraft (no platoon) to 100 (which closely resembles a continuous flow). Between these two sizes, platooning represents a compromise between letting flights go anywhere they want and keeping them highly organized. With platooning, a small number of aircraft are grouped together, and since they can be treated as one unit they are simpler to route and to monitor. The disadvantage is that platooning aircraft must have the same arrival and destination point within any given airspace.

With platooning, demand is generated at time intervals of half the time it takes for a platoon to enter the airspace. For example, a platoon capable of carrying of 10 aircraft is 50 nmi long (given an MIT of 5 nmi), and therefore takes 7.14 min (50/420 h) to enter the airspace; thus, demand is generated at 3.07 min intervals. Our experiments did not include mixed platoon sizes; within a single experiment, all platoons were the same size.



Platoon Size of 1

Platoon Size of 2

Platoon Size of 3

Figure 43. Different levels of platooning.

5.5 Experiments

5.5.1 Experiments with Synthetic Weather

Figure 6 shows plots of throughput as a function of WSI. The plots include mincut values, which give the theoretical upper bound on the throughput if air lanes are perfectly packed, as given by the maximum flow. For Free Flight, the actual achieved throughput was typically 15% to 50% of the theoretical upper bound. Monotonic and Unidirectional Rules typically achieved 30% to 50% of theoretical capacity, while the less-constrained All Altitude and Alternating Altitude Rules achieved 15% to 30% of theoretical capacity. Packed routing methods on Monotonic and Unidirectional cases, however, achieved 50% to 90% of the theoretical capacity of the airspace, occasionally (at 0% WSI) achieving the actual theoretical capacity.

There are several evident reasons for the difference. Free Flight demand was random and inflexible, reflecting that in a Free Flight scenario aircraft have specific routing demands and will not want to be shifted from their desired route. However, this also means that flights tend to block each other, as the airspace fills up with somewhat randomly-oriented routes. The discrepancy disappears at higher WSIs (40% and above), reflecting that, at such high weather coverage, there are few candidate air lanes to begin with, and the first aircraft to arrive that is willing to travel in the available direction will take it.

Monotonic and Unidirectional Rules generally do not achieve the capacity of All Altitude or Alternating Altitude Rules. The reduced capacity is caused by the entry constraint imposed by the rules: flights following the MR/UR can only enter from one side, while flights following the less constrained AA/AAR can enter and exit from virtually any pair of sides. Based on this property, we estimated that AA/AAR capacity should be about double that of MR/UR, and this seems to be the case the majority of the time.

The mincut dependence on the WSI is different for each ATM control law. As mentioned above, the weather constraints in our experiments were generated at random uniformly over the square, thus modeling popcorn convection. It is known that in the presence of popcorn convection, the mincut drops roughly according to $1-\sqrt{WSI}$ [MPK]. This is evident in the plots for the unidirectional flows. For the other rules, the drop was closer to linear. This is due to the fact that in these rules the mincut line was always going close to the “corners” of the square (the bottom B and top T, as in Figure 4). Thus, although the weather cells were generated over the whole square, only a relatively thin region close to the corners impacted the mincut value. This is a feature of squall line weather organization. As confirmed in our earlier work [MPK], squall lines lead to a linear decrease of the mincut with WSI.

There is a small non-monotonicity of the MR-mincut, due to the random weather generation, but this is just due to statistical fluctuation in randomly generated inputs.

Figure 7 shows the complexity of the airspace in these experiments. An increase in flow organization (AA to AAR to MR to UR) produces a decrease in complexity, as would be expected. An increase in platoon size also produces a decrease in complexity, because the larger platoon size guarantees that flight velocity vectors are very highly correlated for long periods of time, since all flights in a platoon follow the same path. The only time when the complexity increased when going to higher platoon sizes is when the platoon size goes from 1 up to 2 or 3. The reason is that the complexity has two parts to it: the proximity of aircraft and the variability of aircraft velocity vectors. When there is no platooning (platoon size is 1), the first term is negligible in comparison to the same term in the presence of platooning. The absence of the first term in the complexity explains the increase in complexity when going from platoon size 1 to platoon size 2 to 3.

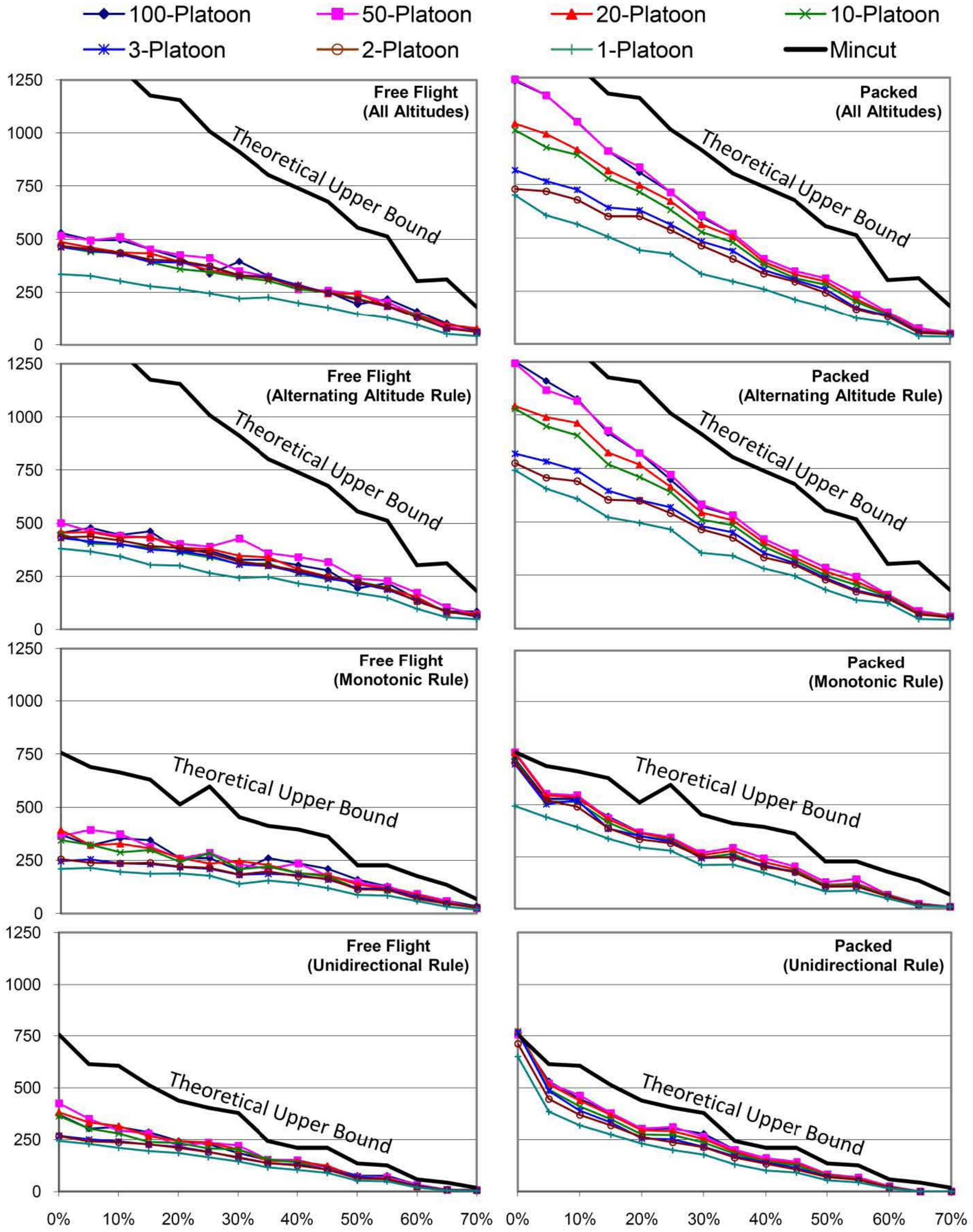


Figure 44. Throughput (aircraft/h) vs WSI (unitless) for synthesized weather.

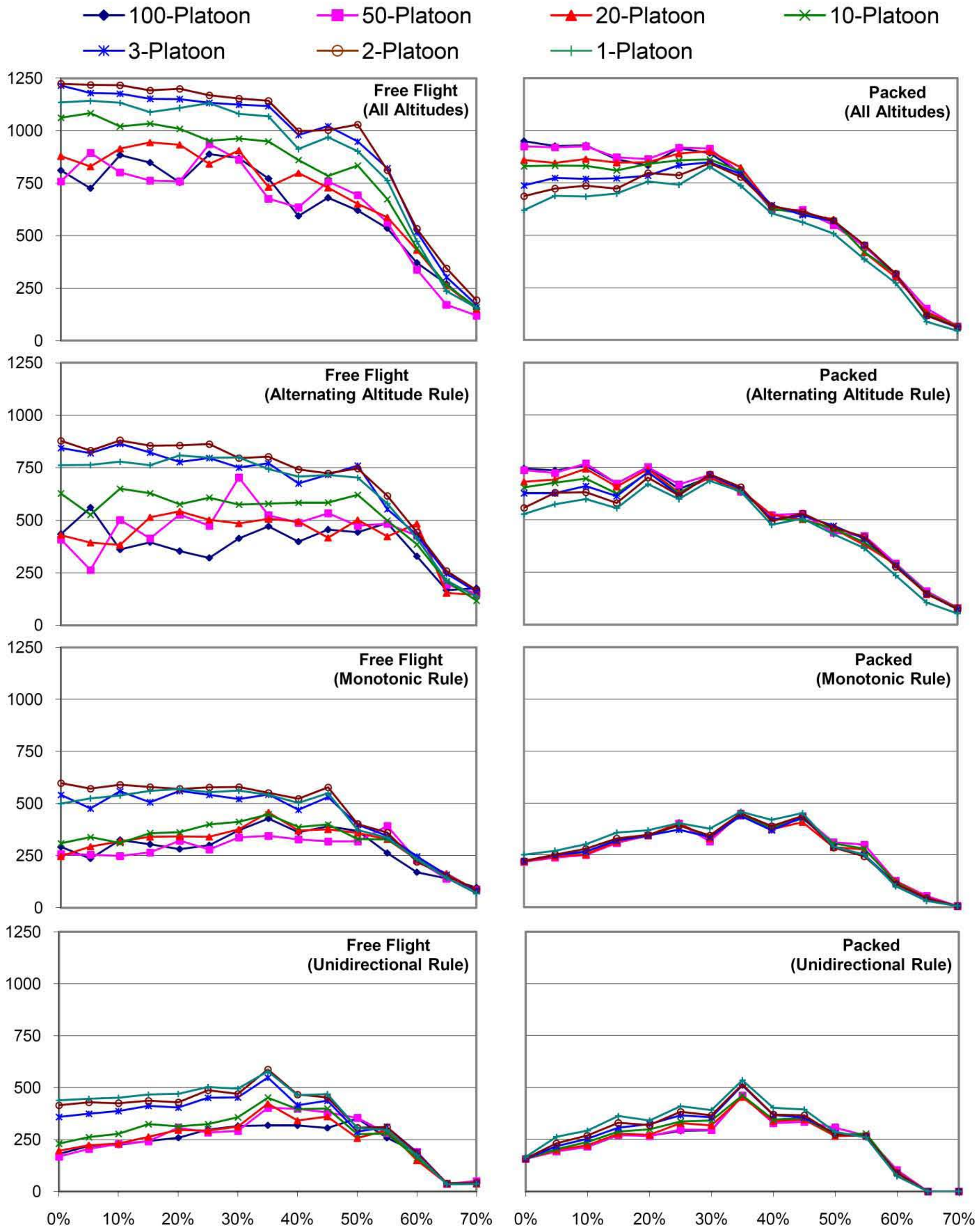


Figure 45. Complexity (unitless) vs WSI (unitless) for synthesized weather.

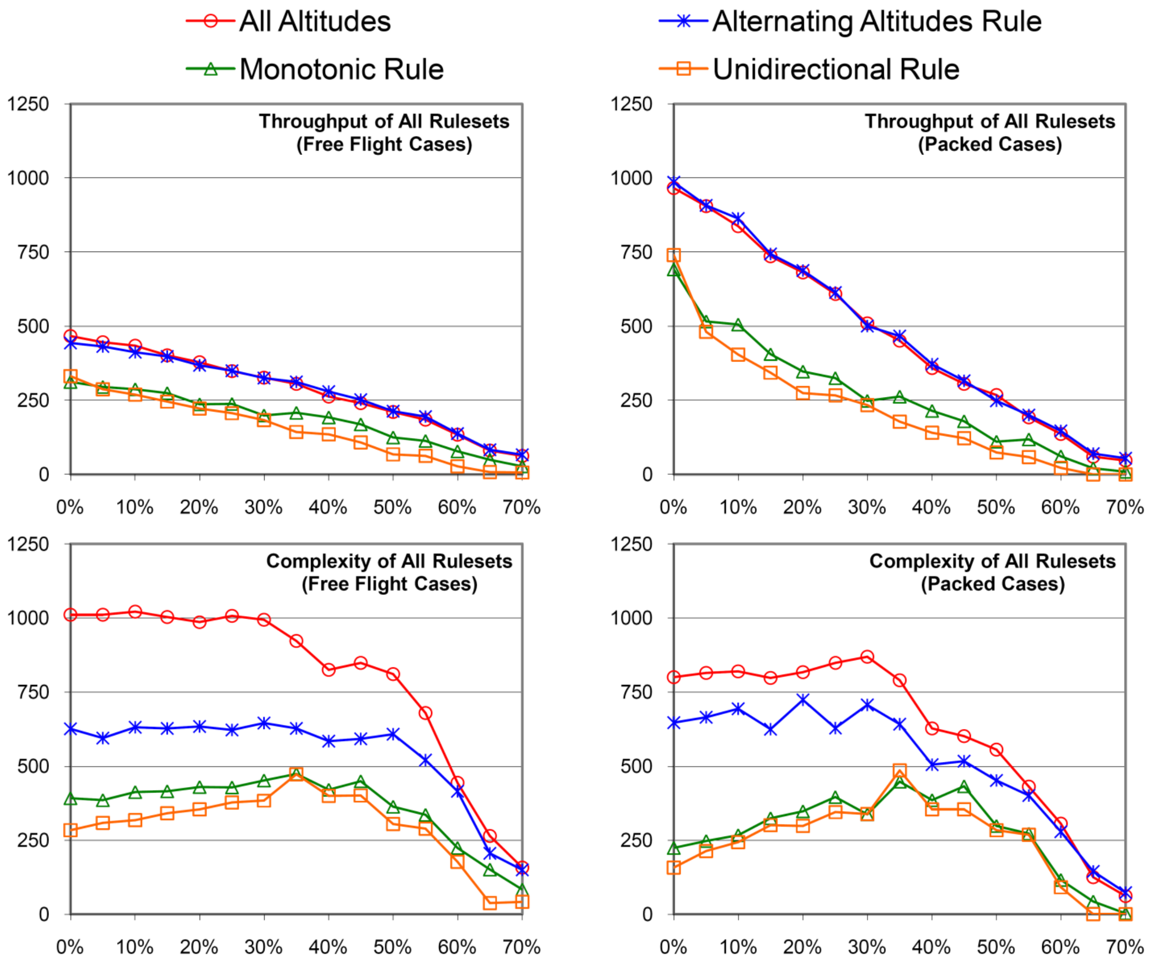


Figure 46. Comparison of all methods for synthesized weather.

Note that platoon size has little to no effect on complexity in Packed scenarios, especially with more controlled flight rules and with higher WSIs. Platoons tend to follow the same routes in Packed scenarios because there is little reason to deviate from the existing shortest path, so in practice small platoons tend to produce as little complexity as large ones.

We examined the dependence of the complexity of the solutions on the ATM control laws in order to determine whether the choice of a particular ATM control law would have a significant impact on flight controller workload. We found that, as a general rule, complexity increased substantially when the directions of flights were permitted a larger degree of freedom. When flights can cross in any direction, the resultant situation is theoretically harder for a controller to monitor than when only east-to-west flights are permitted. Complexity generally decreased with larger platoon sizes because larger platoons enforce more order on aircraft routing.

When plotted against weather severity, we find that complexity peaks at a WSI of approximately 10-15%. Below that value, there is little enough weather that most platoons will generally take a fairly straight route to its destination, with few turns, which is easier to track. Above that value, reduced throughput dominates, and complexity falls simply because fewer aircraft are actually present.

5.5.2 Experiments with Real Weather

The real-weather experiments were done using roughly 200-by-200 nmi regions of airspace and a platoon size of 40. The maximum theoretical capacity of a 200-nmi-square region of the airspace, under the Unidirectional or Monotonic Rules, is 2604 aircraft or 31 lanes, assuming that all aircraft are traveling directly west-to-east with no weather constraints and maximally packed. Under the All-Altitude or Alternating Altitude Rules, the maximum is 5208 aircraft or 62 lanes, under equivalent assumptions of perfect packing into the airspace.

When routing aircraft in parallel flows from west to east, it is found (Figure 9) that even small fractions of weather (WSI = 10%) could cause serious disturbances in the ability to route the maximum possible number of aircraft. Typically 40-60% of the clear-weather capacity of an airspace is actually reached in such cases.

This is not the case, however, for more flexible flight rules. Both the mincuts and the experimental data suggest a more linear falloff in capacity as the airspace increases in weather. Since real weather at the 200-nmi scale tends to clump into large walls of hazardous weather, these walls tend to significantly block either the east or west borders, making unidirectional flight impossible, but if flights are allowed in all directions, two or three out of the four border lines will typically be mostly unblocked, and flights can still be routed.

The characteristics of the synthetic “popcorn” weather differ markedly from that of real weather. In particular, the real weather used in our experiments is non-uniformly distributed across the airspace. The typical result (Figure 10) of this in the real-weather experiments is that there are only a few gaps in the airspace hazardous weather constraints, allowing through only a few lanes of traffic. However, the real weather in our experiments is not considered a squall line. When the area is relatively clear, there is little difficulty in routing many aircraft. Partially this is the result of the greedy nature of FBRP routing; one platoon is routed at a time, along the shortest possible path, and this tends to block passages between neighboring hazardous weather constraints. As above, this tends not to affect All Altitude and Alternating Altitude Rule scenarios very much, because there are still many options for flights to enter and exit the airspace.

Real weather experiments show the same complexity trends as do the synthetic weather experiments, for the same reasons.

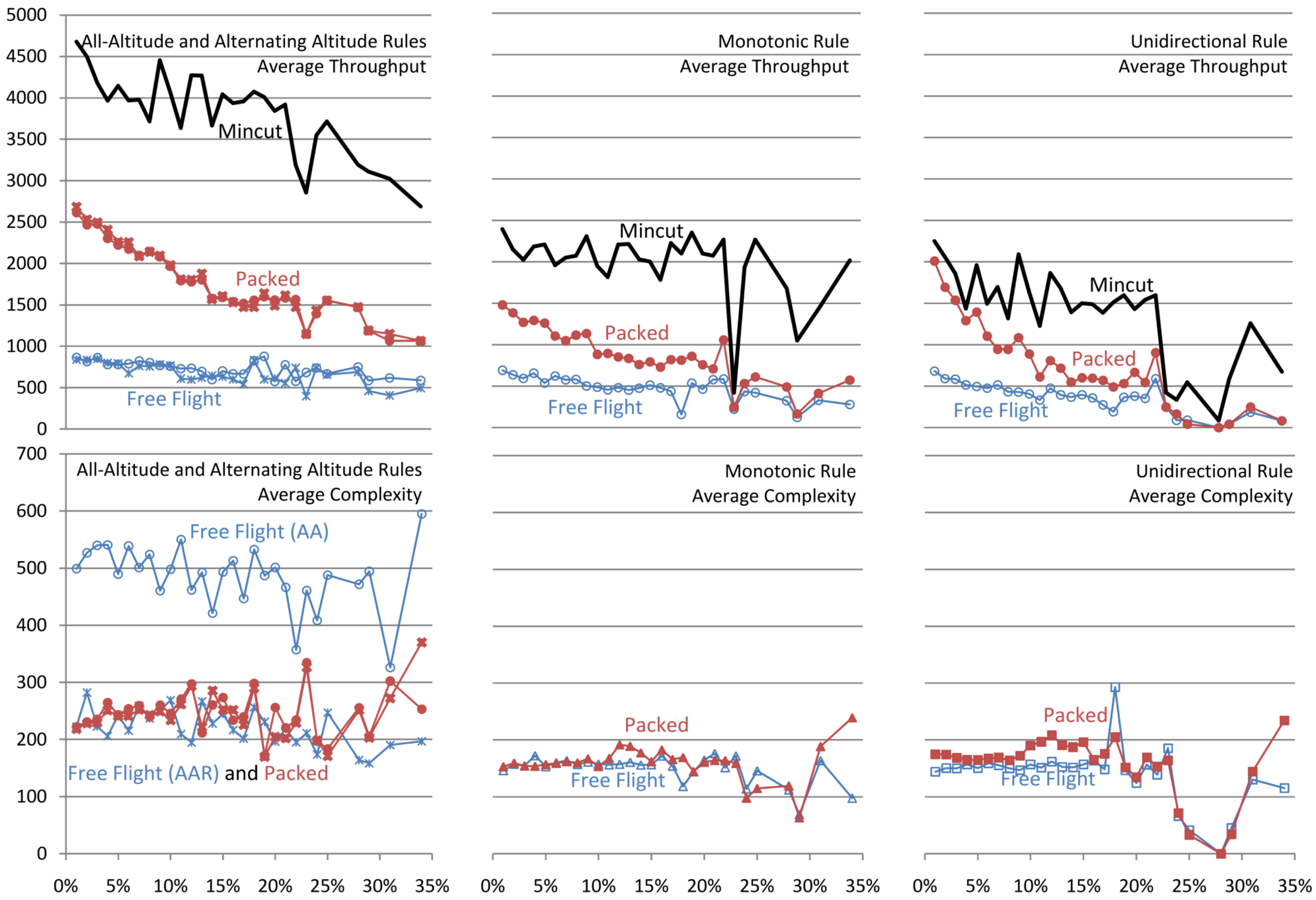


Figure 47. Throughput (aircraft/h) and complexity (unitless) versus weather severity (unitless) for real-weather samples.

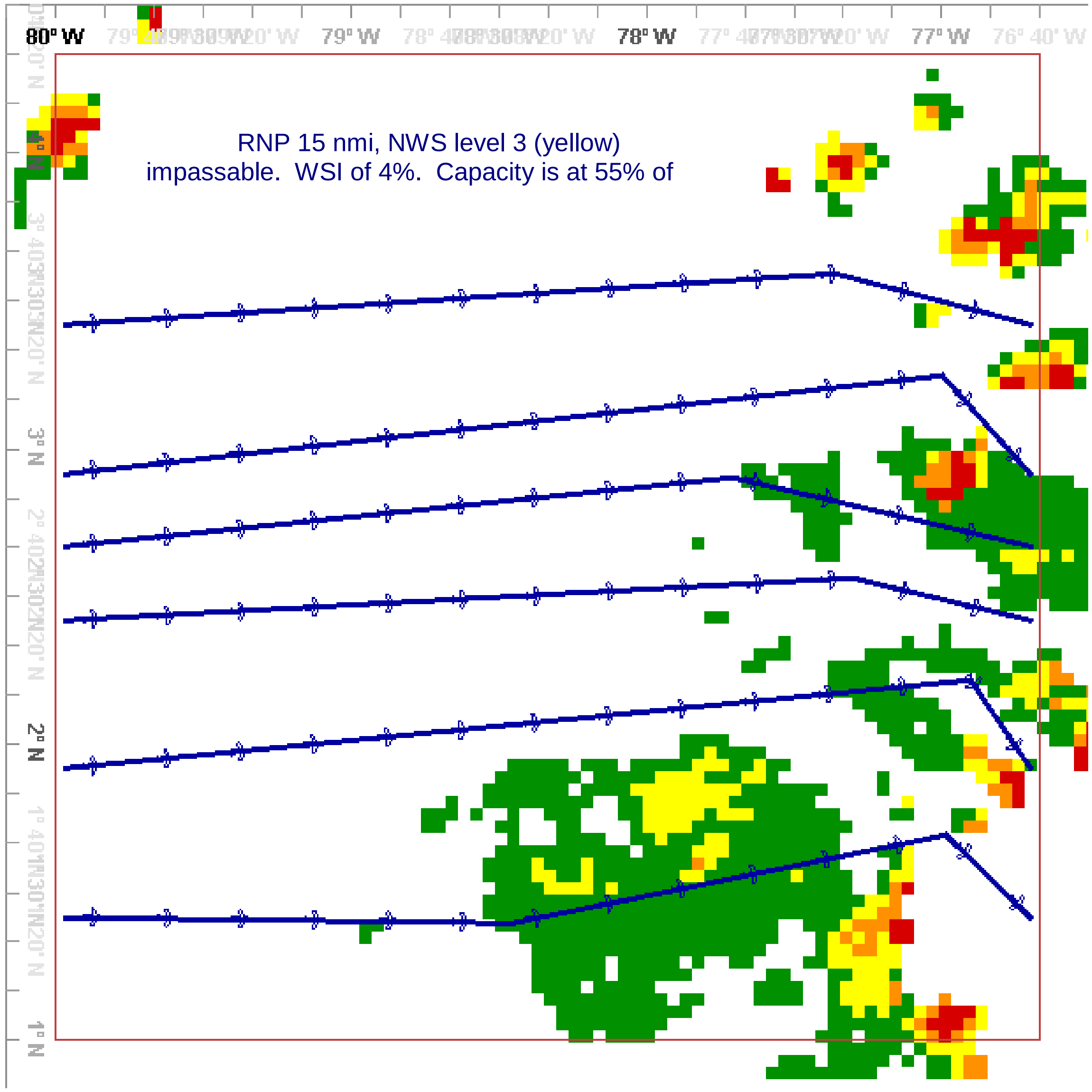

80° W
79° W
78° W
77° W
76° 40' W
4° N
3° N
2° N
1° N
RNP 15 nmi, NWS level 3 (yellow)
impassable. WSI of 4%. Capacity is at 55% of

Figure 48. Typical solution for real weather data over 200-by-200 nmi of airspace.

5.6 Conclusions

This chapter presents both a theory and a practical method for airspace capacity estimation. This study investigates the tradeoffs between: (1) capacity estimations under air traffic management control laws that range from decentralized Free Flight to highly-centralized control – spanning aircraft flight in all directions to aircraft following an alternating altitude rule to aircraft flying only in one direction (e.g., west-to-east), (2) capacity estimation as a function of weather severity ranging from no weather constraints to severe weather constraints that make the airspace impassible, (3) flows that range from platoon sizes of one (individual aircraft) to very large platoon sizes that approximate continuous flows, and (4) the complexity of the resulting traffic flows.

We find that the theoretical capacity of an airspace is quite high compared to the estimated capacity of Free Flight methods of crossing the airspace, and that organization of flows utilizes the airspace much more thoroughly, at a lower complexity, but at the cost of enforcing heavy constraints on all flights within the airspace. In general, the use of platooning increases the capacity for each ATM control law, however, with diminishing returns after platoon sizes of two or three aircraft in a platoon. This is not true for low-weather coverage cases of All-Altitude and Alternating Altitude Rule scenarios, however; platooning continues to provide moderate benefits when increased past a size of three. Maximum capacity is most accurately measured via theoretical mincut methods, when applicable, but our theoretical bounds are tight only for static weather and strict flows from one side of an airspace to the other. Experimental methods using algorithms to maximize capacity (as with centralized Packed scenarios) are capable of closely approximating the mincut measure. Our experiments demonstrate that some planning component is a requirement for efficient utilization of airspace, but that this becomes less important as the severe weather coverage rises and the amount of free space available reduces the improvement that planned routing can achieve.

5.6.1 Future Work

The theory described here was developed, and the experiments run, under the assumption that all aircraft arriving at the entry to the airspace have the same RNP and MIT requirements. In a real-world scenario, aircraft with different RNPs are likely to be routed through an airspace. Future work will address the computational complexity of computing routes for flows of aircraft with different RNPs. We expect that, while the general problem is likely to be computationally intractable, special cases (e.g., having only a few different RNPs) may be efficiently solvable.

In our model we assumed that the position and size of the constraints in the airspace were deterministic, i.e., known in advance rather than being random variables. It is more realistic to assume that weather is given to us as an uncertain forecast, and preferable that in the future it be given as an ensemble set of forecasts. With an ensemble set of forecasts, scenario forecasts that currently have higher skill are assigned higher probability. The ensembles can be generated by perturbed initial conditions, multiple models applied to the same initial conditions, using bred vectors, or through a data assimilation process, as a few examples. The mean of the ensemble is typically chosen as the ideal forecast, while a covariance is computed to signify the spread of the ensemble. In [MPK], capacity estimation is considered in a simple, scenario-based stochastic weather model; namely, for each of the forecasts, the mincut is computed using techniques described here for the deterministic case, and then the probability distribution of the mincut value can be determined explicitly, along with the mean, variance, etc. In future work, we will extend our results here to more sophisticated stochastic weather models, e.g., based on probabilistic weather maps, together with an explicit modeling of the spatial correlation between nearby points.

Chapter 6: FBRP Software Architecture

6.1 Introduction

The Flow-Based Route Planner is a software system as well as a technique, developed over four years with numerous refinements and a few partial reconstructions to adapt to new experimental requirements. The major part of the system is the graph construction package, used to build complex implicit graphs and keep track of obstructions that must be checked to determine the legality of edges. The route management and airspace view systems are closely related. A special file format was also adapted from existing formats in order to provide a simple, compact, and human-readable format for storing scenarios and solved routes.

6.2 Graph Construction

The graph construction system was developed explicitly as an object-oriented filter-style system for building and manipulating graphs.

6.2.1 Graph interface

The graph interface provides basic services needed by any graph. From a programmatic standpoint, broken down as purely as possible, a graph is a container of arbitrary objects that have links to each other, determined by the structure of the graph. So the Graph interface is very sparse, providing methods for getting the number of vertices in a graph, iterating through the objects in the graph, and iterating through all objects that have an inbound edge from a particular vertex.

As a point of efficiency, an edge cost is not calculated until it is actually needed, at which point it must be requested via an edge cost calculation method. These edge costs may be infinite, signifying that the edge does not actually meaningfully exist – if a given graph traversal will not be tolerant of this, then the edge cost needs to be checked before the edge is actually used. However, for the FBRP it is more efficient to calculate edge costs as needed because of the relative high cost of collision detection on edges. This also makes it easier to change graph structures with subclassing and Decorator-style relationships where one class uses another [GoF].

Additionally, since A* search depends on heuristic estimation of the distance between a node and the goal, the interface provides a method specifically for this to be calculated. The value is guaranteed to be equal to or less than the actual cost to get between the two points on the graph. (Since this value is not always easily calculated for arbitrary graphs, it is permissible for an implementing class to simply return zero.)

6.2.2 Graph Classes

A number of classes are implemented for graph usage. A few of them (GridGraph, CompleteGraph) stand on their own. The objects that these two graphs use as vertices are a special type of point class. The others all take existing Graphs as parameters and alter them by changing the way particular cases are handled. Normally this means providing a new version of the cost function that examines any point pairs that are checked for cost and. If the examination reveals an illegal segment, the returned cost of the segment is infinite.

- GridGraph – The basic graph that provides for a defined grid of points. It implements the grid structure and the connectivity constant for the initial route search graph. It also calls the Obstruction classes, described below.
- CompleteGraph – A complete graph over a provided set of points. It provides the basic structure of the refinement-stage search graph. It also calls the Obstruction classes, described below.
- MultiSourceSinkGraph – Augments an existing graph so that two more Objects are involved: a supersource and a supersink. The supersource and supersink are connected to any number of existing vertices with connections of cost zero.
- SupplementedGridGraph – Augments an existing graph with a number of arbitrarily-placed points that are directly connected to nearby points in the grid. This allows a grid graph to connect to precisely placed points along, for example, the circular boundary of the transitional airspace of an airport (which are typically then used as source points in a MultiSourceSinkGraph).
- AngleLimitedGraph – Restricts an existing graph so that any two Point objects are only considered connected if the vector between them lies in a particular range of angles.
- MonotoneRadiusGraph – Restricts an existing graph so that any two Point objects are only considered connected if the origin point is further from a centerpoint than is the destination point. Any legal path on the graph must go closer and closer to the defined centerpoint. In actual usage the centerpoint is normally an airport.
- OvalLimitedGraph – Restricts an existing graph so that any point which lies outside a specified ellipse cannot be connected to any point whatsoever.

6.2.3 Graph Assembly

Each of the above-defined graphs (except for GridGraph and CompleteGraph) can take the other graphs as parameters, leading to a plug-in style of construction for the graphs actually used. For example, the capacity estimation searches use graphs constructed as follows:

- A GridGraph provides the basic structure of the search graph.
- A SupplementedGridGraph contains the grid graph, and augments it with one each of source and sink points, neither of which are normally on the grid.
- An optional AngleLimitedGraph contains the SupplementedGridGraph but throws away any connection that fails the direction test.
- An optional OvalLimitedGraph contains the above graphs, throwing away any connection that strays outside of the designated area.

- A MultiSourceSinkGraph is constructed to contain these, albeit redundantly since there is only one source or sink. However it's simpler for the code to assume a MultiSourceSinkGraph is the outermost class involved.

In this way the entire graph is constructed from smaller pieces, which means that all of the code to deal with any given constraint is contained entirely within one class, and can be debugged independently of the other constraints. (This is not strictly true of Obstructions, but the basic code is very simple and the details are all handled in the Obstruction hierarchy.)

6.3 Obstruction Constraints

The Graph classes directly handle constraints relating to where flights would be allowed to go in a clear, unobstructed sky. However, in the scenarios contemplated in this thesis, the sky is not clear at all. Hazardous weather storms move across the sky, making passage dangerous for aircraft; other aircraft routes, already laid out, have safety margins that may not be crossed; no-fly zones around military and other sensitive installations force aircraft to find alternate routes; existing airport and sectorization structures simplify air traffic control but restrict the movement of aircraft. All of these represent obstructions in space-time that aircraft must avoid.

6.3.1 Obstruction interface

The only significant member of the Obstruction interface is named isObstructed. This method takes as input a Segment, which contains two 3d points, a duration, and a desired conflict radius, and can use *k*-dops for prefiltering potential obstruction hits. Each Obstruction has the option of using the Segment's methods to check whether obstruction is even possible; if not, then the rest of the cost of collision detection is not necessary.

Many obstruction types store bounding volume information to make *k*-dop prefiltering easy and fast. The bounding volume information must always be there, but it is permissible for the volume to be unbounded in some or all directions, in any desired combination.

6.3.2 Obstruction classes

All obstructions have the same basic structure: they each contain enough data to implement isObstructed for the real-world object that they represent.

Some of the obstructions in the current codebase are obsolete, but not yet removed because of their use in old experiments. Those are not listed.

- ObstructionSet – Stores a list of other Obstructions, and the union of their bounding volumes. This Obstruction allows sets of obstructions to be treated as one, simplifying the code. ObstructionSets can therefore also be nested like a tree of arbitrary-degree nodes.
- TimedObstruction – A container for another Obstruction, such that no part of the Obstruction that falls outside of a specified time interval is considered as part of collision detection. Prefilters the incoming segment so that only the

part of the segment that falls in the time interval is actually used for collision detection. The code, however, does assume that the Obstruction in question is static; this class is used to change an obstruction from static to dynamic.

- CircleObstruction – Defines a particular circle of radius zero or higher that a route may not intersect. The circle has no time limits and is always an obstruction regardless of simulation time. This is used to implement popcorn weather, among other things.
- ImageWeather – Stores the grid of weather data that is used to check for hazardous weather intersection. The original grid is stored as well as any recently-used convolutions of the original grid. (Recall that the weather data is specially convoluted with a circle of radius equal to the weather avoidance distance.)
- PathObstruction – Does collision detection against a single pre-existing route that has already been routed in the scenario under consideration. Stores *k*-dop and triangle-mesh information for the route stored in the PathObstruction.
- PolygonObstruction – Stores a polygon that routes may not cross over. Only the boundaries are considered, so this obstruction may be used to contain routes as well as restrict them.

6.4 Routing Requests

Each routing request that the system can handle has slightly different constraints. Each of these kinds of requests is therefore encapsulated into a separate class of type Routable, which allows each request to be programmatically handled correctly.

6.4.1 Routable interface

The Routable interface provides a number of important services. The first used is generally the method named resolve(); this takes one parameter, a RouteManager that contains the scenario being examined. The Routable object then calculates a suitable route according to the constraints of the request and the constraints already present in the current scenario. Once resolved, the Routable represents a number of aircraft that will use the route in question, so getObstruction() and getAircraft() allow retrieval of that information for the appropriate purpose.

6.4.2 Routable classes

Each Routable class represents a different kind of routing request.

- FlightPath – Our original experiments called for a number of flows to be routed between two different arbitrary points. FlightPath implements this condition.
- OneOffFlightPath – Occasionally only one single flight or flow between two points is required. OneOffFlightPath routes this kind of request.

- CrossSectorRoute – Several of our experiments demand the ability to send a series of parallel routes between two boundary edges of a sector. CrossSectorRoute implements this kind of request.
- FanIn – Our transitional airspace experiments required that three different points on the airspace boundary be connected to one metering fix, without interfering. FanIn implements this request.

6.5 Route Manager System

The central organizational class in the Flow Based Route Planner is the class RouteManager, containing pointers to the scenario bounds, the obstacles and parameters of the scenario, and the routes that have been requested.

- When a routing request is made, RouteManager redirects the request to the route itself; each type of Routable object knows how to handle its own routing.
- When global information is required – for example, to dynamically set the GUI limits of time selection – RouteManager calculates it.
- A RouteManager represents a single scenario. When a scenario is read or written to disk as a whole unit, a RouteManager is constructed or examined.
- All painting requests from the GUI pass through the RouteManager; it has control over delegation of painting requests to subsystems.

6.6 File I/O

Late in development it was discovered that file I/O was becoming a relevant factor in the time and complexity of experiments, although not their results. In order to improve this a new file format was adapted from existing public formats. A sample is below:

```
Airspace = {
  Bounds = { -7639.0 1246.0 720.0 -3918.0 3155.0 1440.0 }
  Airport = {
    Name = "New York"
    Location = { -4440.384 2442.852 }
    Radius = 200.0
    MeteringFix = {
      Name = NYA
      Type = Arrival
      Location = { -4482.384 2412.852 }
      AvgVelocity = 420.0
    }
  }
  Weather = {
    ObstructionSet = {
      TimedObstruction = {
        BeginTime = 711.0
        EndTime = 721.0
        ImageWeather = {
          BoundingRect = { -7219.74 1592.04 -4617.94703125
3001.95015625 }
          CellsPerMile = 0.06879871002418704
          Data = {
                Size = { 97 179 }
                Singleton = { 35 0 165 93 41 95 29 95 31 96 31 }
                Singleton = { 30 56 96 61 0 61 93 61 97 63 93 71 171
72 171 72 178 74 55 75 101 76 101 76 103 79 53 79 55 80 55 81 55 82 101
83 44 83 101 86 34 87 31 87 35 87 82 }
                Singleton = { 45 59 97 }
                Run = { 55 95 35 35 30 }
                Run = { 57 95 30 45 30 }
<etc>
                Run = { 94 29 35 30 30 }
                Run = { 94 33 30 35 35 45 -3 55 }
          }
        }
      }
    }
  }
}
```

Singleton entries start with a weather value, and then list a series of X-Y coordinate values that the weather value should be placed at. Each of these weather values is isolated in the original data and cannot be part of a run.

Run entries start with a coordinate pair where the run of data begins. The values afterwards fill in the values of the original array. Each positive value represents a weather value; negative values mean that the following weather value should be repeated a number of times.

Figure 49. Example of file format.

It was found that weather data, compressed into a run-length-en coded format and embedded in a simple text-based file format as above, was substantially faster to load and run, making the thousands of experiments and tests required by our research much more efficient.

Chapter 7: Performance Characteristics of the FBRP[4]

7.1 Introduction

We have conducted numerous experiments with the FBRP system to study its performance and results. We have also examined the results of the FBRP under various choices of the parameters to study their effect on the algorithm. The primary parameters believed to affect the speed and effectiveness of routing were the density of the search grids, the connectivity constant chosen, and for the refinement phase, the number of Steiner points used per original point. Our dependent variables are the running time, in seconds, and the waypoint count of the resultant routes.

These experiments were performed on the transition airspace scenario, as described in the chapter "Weather Avoidance in Transition Airspace".

7.2 Studied Scenarios

The scenarios used for testing were from three representative days of Atlanta airspace. In broad terms, May 22, 2002 was a mostly clear-weather day, while June 26 and 27, 2002 included clear, moderate and severe weather systems. In all cases, flows were calculated for each 30-minute interval from 12:00 to 23:30 (the last flow being valid until just before midnight).

7.3 Studied Parameters

We examined the effect of the following parameters on the output:

- Density of the search grid used in Phase 1 of the algorithm – 32, 48, 64, 96, 128. These correspond to grids of resolution (grid cell size) 12.5 nmi, 8.3 nmi, 6.3 nmi, 4.2 nmi, and 3.1 nmi, respectively.
- Connectivity of the search grid – see above for a definition of the connectivity constant, *K*. This varied between one and four.
- *M*, the number of Steiner points used (on each side) for each input point in Phase 2 of the algorithm. This varied between zero and three. (The size of τ was adjusted to produce the same total width of Steiner points, however.)

[4] The material in this chapter was adapted from Prete, J., and Mitchell, J.S.B., "Safe Routing of Multiple Aircraft Flows in the Presence of Time-Varying Weather Data," *AIAA Guidance, Navigation, and Control Conf.*, Providence, RI, Aug., 2004.

7.4 Successful Routing

One of the primary objectives of the FBRP is to increase airspace capacity, which in turn means that a successful FBRP should be able to route successfully a large fraction of the requested routes, thus producing a higher capacity airspace. [KPPM] compares the FBRP to historical data, but it is also of direct interest to examine what parameters result in the highest proportion of successful routings.

Counter-intuitively, increasing the density of the Phase 1 search grid produced a similar or lower success rate to lower-density grids, while increasing its connectivity produced the expected increase in success rate. (This observation applies, of course, only to the small ranges of values of grid densities used in the experiments.) Using more Steiner points in Phase 2 refinement produced a marginal increase in routing successes, most likely because it improved the algorithm’s ability to recover routes with the required complexity.

The fact that the success rate was better than 50% in all scenarios suggests that we could achieve an average throughput of 1.5 air lanes to each metering fix, or 6 lanes of inbound aircraft to the airport.

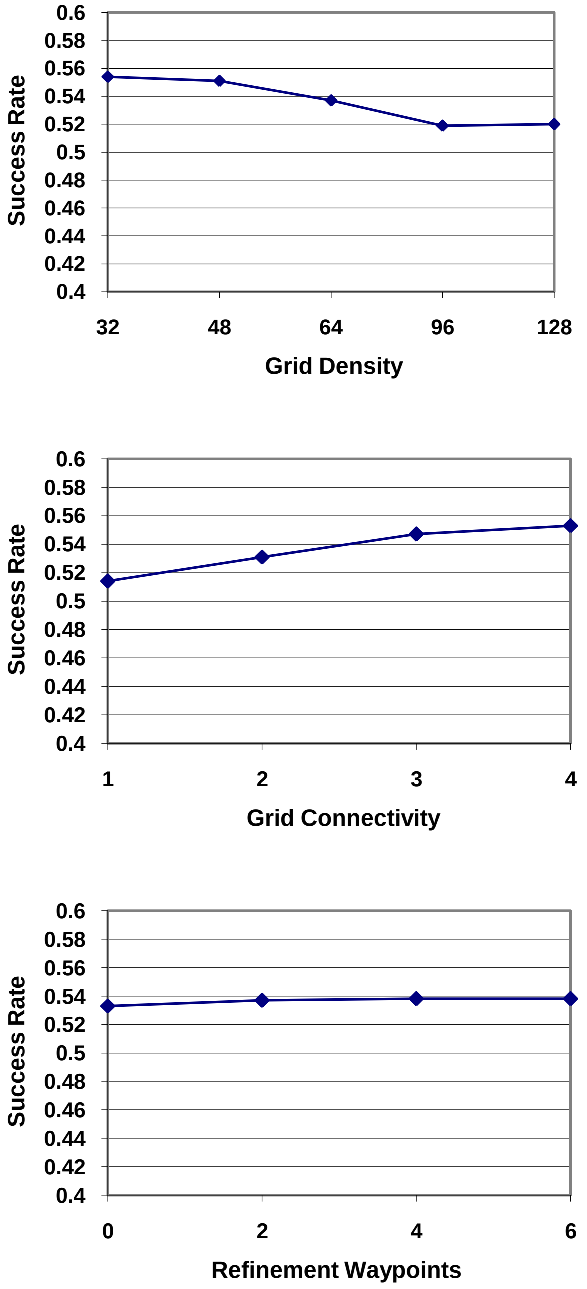


Figure 50. Success rate vs. search properties.

7.5 Low-Complexity Routing

One of the primary purposes of the algorithm is to produce routes of low complexity so as to insure that the routes themselves do not unnecessarily increase the workload of air traffic controllers or pilots. It is important to study how the parameters of the algorithm affect this aspect of the output.

The output necessarily contains a large number of trivial routes, in which a direct connection between the start and goal waypoints can be immediately established. These routes have been ignored when examining complexity since they provide no information with which to compare the parameters of the algorithm.

The experiments showed no significant effects on average complexity of nontrivial routes for any of the parameters. Mid-sized (6.3 nmi) grids have lower average complexity than large or small grids, but the effect is moderate. Low-connectivity grid searches show lower average complexity than high-connectivity grid searches, but this is most likely a result of the higher rate of routing success, which would in general produce more of the complex routes that are missed or unsuccessful in a low-connectivity grid search.

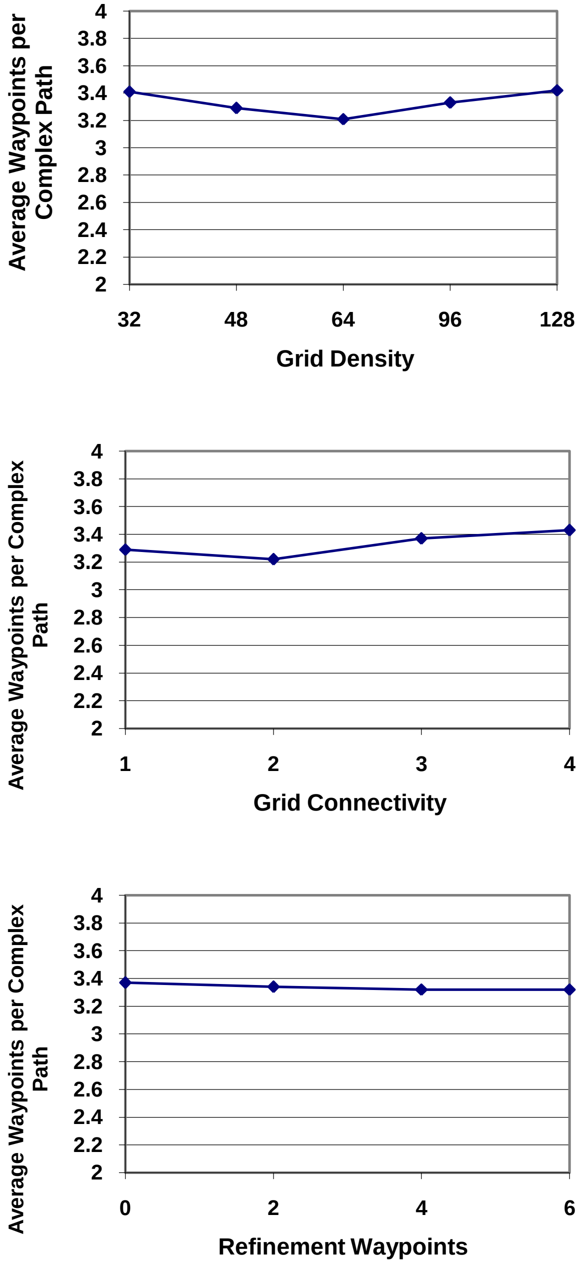


Figure 51. Route complexity vs. search properties.

7.6 Running Time

It is of particular interest to determine expected running time of the algorithm, because of the real-time nature of air traffic control. Investigating running time as influenced by the parameters of the algorithm produced the expected results.

In general, increasing the grid resolution directly increases running time as the square of the grid resolution. This was an expected result; the vast majority of time is spent searching the grid of points, and the region that must be searched increases proportionately to the number of nodes in the grid. Similarly, increasing the number of Steiner points used in Phase 2 refinement produces an expected increase in running time. Interestingly, increasing the number of Steiner points has a more pronounced effect on running time when the grid is denser.

The most interesting effect here is that increasing the connectivity constant decreases running time to a point – a connectivity constant of three – and then running time increases. This appears to be the case for almost every combination of algorithm parameters.

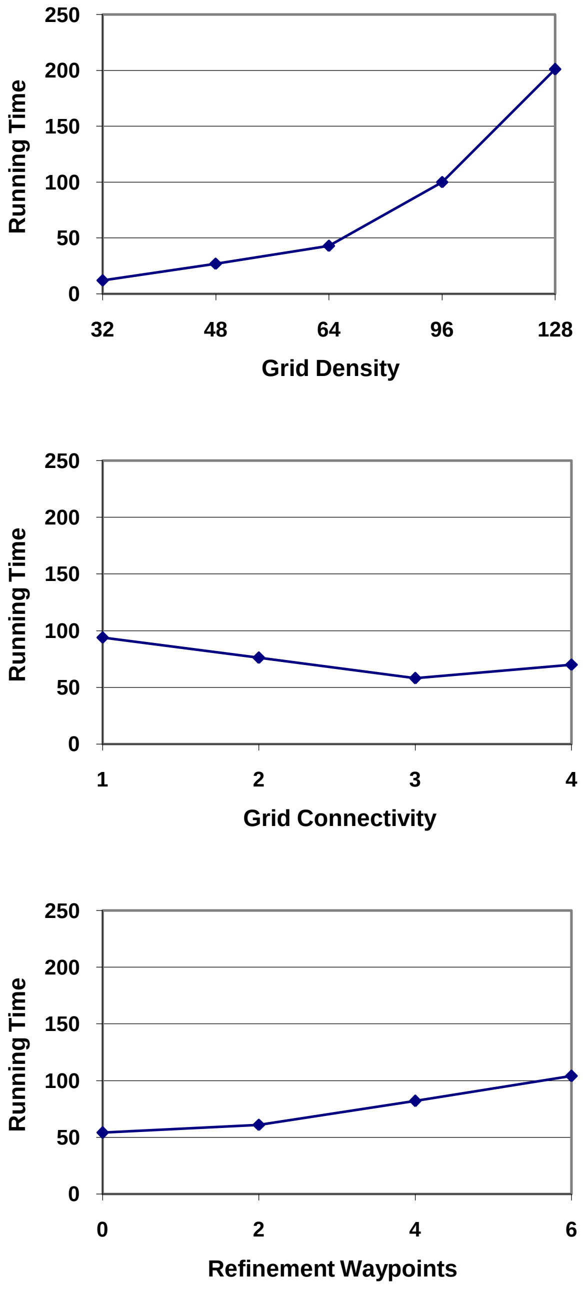


Figure 52. Running time vs. search properties.

Chapter 8: Related Research

8.1 Optimal Route Computation

There is a large literature on optimal route computation, particularly in the field of computational geometry. See the survey articles ([Mi1,Mi2]) for extensive background on computing optimal routes in geometric data. Related to our approach, [CKLM] studied multiple path routing in space-time in performing conflict resolution. Several approaches apply an "optimal" path algorithm based on grid search methods ([DW], [KWH], [LMS], [MM], [MP], [BV]). The problem can be formalized as a "weighted regions problem" in which routes obey Snell's Law of refraction ([KWH], [RRZM], [RR]) as a local optimality criterion. Algorithms can exploit the fact that optimal routes bend at boundaries between regions of varying weather severity in analogy with light rays that refract as they pass through regions of varying refractive index ([MP], [Mi3]). Another related approach ([KLM]) allows one to search for paths having at most *k* turns (waypoints), while avoiding hazardous weather, thereby bounding the workload of the pilot and controller required to track the solution. In general, these approaches mostly address the routing of a single aircraft rather than designing routes for flows of aircraft or for system optimization, as addressed in this thesis.

In the context of air traffic management, the computation of optimal routes has also been studied extensively (see, e.g., [DW,KLM,KWH] for weather avoidance optimal routes and [JB] for wind-optimized routes).

8.2 Coded Departure Routes

While there have been quite a few algorithmic approaches to routing aircraft around weather constraints, the current decision support tools (DSTs) used in Air Traffic Management (ATM) do not automate the process of weather avoidance routing or selecting among multiple weather avoidance routing options. The Expedite Departure Tool (EDT), for instance, efficiently routes aircraft out of the terminal area, but does not currently include considerations for convective weather avoidance ([JI]). Using predicted Estimated Times of Arrival (ETAs) at fixes and waypoints for both arriving and departing aircraft, a controller can use EDT to determine the best departure route and climb for an aircraft to safely traverse the TRACON area. The Route Availability Planning Tool (RAPT) ([DA], [ADMCG]) exploits Integrated Terminal Weather System (ITWS) data to assist with the planning of departure route closures during convective weather events, but it does not reroute around weather constraints. RAPT combines weather forecasts with determined departure paths to predict the availability of specific departures along a specific fixed route in the future. RAPT provides a timeline showing the departure route status (open/closed) as a function of departure time. In terms of evaluating alternative routing choices, recent work ([KWSSMS]) in the Collaborative Decision Making (CDM) program has begun to study processes to allow AOCs to file multiple flight plans around weather and Flow Constrained Areas (FCAs), to allow for the ATSP the flexibility to evaluate multiple flight plans against routing options, for instance, CDR choices. These processes have not yet been implemented.

The components of an on-demand CDR system are close to being in place. Based on the literature, terminal area tools like EDT and RAPT show promise for incorporating weather avoidance into the terminal area. The RMT tool is already in place to dynamically distribute and manage CDR information, but would have to distribute the information at a much faster rate than today. And processes for submitting and evaluating multiple flight plans (in this case, alternative CDR flight plans), are currently being investigated.

8.3 Capacity Estimation

Early work [S1] investigated the traffic variables, routes, sector geometry, and control procedures that contribute to a control difficulty index that empirically quantifies the workload required on the part of the air traffic control team to manage a sector. The sector capacity is limited by total workload as measured by the control difficulty index. More recent work [M1] presents probabilistic methods for air traffic demand forecasting, including demand count probabilities for sectors. [WSZGM] investigated probabilistic congestion management, including the prediction of traffic levels and airspace capacity. Their work warns that Monitor Alert Parameter (MAP) values should not be considered a measure of airspace capacity. Each sector is assigned a constant MAP, independent of the level of weather present in the sector, which identifies the peak number of aircraft that can be safely handled for a given 15-min time interval. The statistics of sector peak count prediction uncertainty was also studied in previous work [WCGM]. The work of [SWG] studies the problem of predicting sector capacity for sectors in today's NAS through a pattern recognition technique – recognizing the traffic flow pattern is included in their technique.

Generally, these previous methods of studying sector capacity stem from the empirical analysis of how controllers work in the NAS today, including the use of jet routes and controller workload limitations. In contrast to this, we describe in this chapter a theoretical analysis of the airspace capacity as a function of hazardous weather constraints, independent of workload considerations and independent of today's jet routes. The analysis is based on maximum flow concepts in geometric domains, based on the theory given by [M2], [MP2], and [S2]. Our work is in support of the design of new roles for controllers and pilots in the NGATS, and addresses the maximum throughput of an airspace, assuming that workload is not a constraint.

Bibliography

[AMO] Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, NJ.

[Anon] Anonymous, *An Evaluation of Air Traffic Control Complexity*, Final Report, NASA Contract No. NAS2-14284, Wyndemere Corp, Boulder, CA, Oct., 1996.

[ADMCG] Allan, S., R. DeLaura, B. Marin, D. Clark and C Gross, “Advanced terminal weather products demonstration in New York,” *11th AMS Conf. on Aviation, Range & Aerospace Meteorology*, Hyannis, MA, 2004.

[AMP] Allsopp, T., Mason, A., Philpott, A., “Optimal Sailing Routes with Uncertain Weather”, *Proceedings of the 35th Annual Conference of the Operational Research Society of New Zealand*, Victoria University of Wellington, New Zealand, 2000, pp. 65-74.

[B] Brennan, M. (2007), “Airspace Flow Programs – A Fast Path to Deployment,” Journal of Air Traffic Control, 49 (1), pp. 51-55.

[BDSV] Briggs, A., Detweiler, C., Scharstein, D., Vandenberg-Rodes, A., “Expected Shortest Paths for Landmark-Based Robot Navigation”, *The International Journal of Robotics Research*, Vol. 23, No. 7-8, pp. 717-728.

[BV] Bokadioa, S., Valasek, J. “Severe Weather Avoidance Using Informed Heuristic Search”, *AIAA Guidance, Navigation, and Control Conf.*, Montreal, Canada, Aug., 2001.

[CKLM] Chiang, Y.-J. Klosowski, J.T., Lee, C., and Mitchell, J.S.B., “Geometric Algorithms for Conflict Detection/Resolution in Air Traffic Management”, *36th IEEE Conf. on Decision and Control*, San Diego, CA, Dec., 1997.

[DA] DeLaura, R., and Allan, S., “Route Selection Decision Support in Convective Weather: A Case Study of the Effects of Weather and Operational Assumptions on Departure Throughput” *5th USA/Europe ATM R&D Seminar*, Budapest, Hungary, June, 2003.

[DE] DeLaura, R. and Evans, J. (2006) “An Exploratory Study of Modeling En Route Pilot Convective Storm Flight Deviation Behavior," *12th American Meteorological Society Conf. on Aviation, Range, and Aerospace Meteorology*, Atlanta, GA, Jan./Feb., Paper P12.6.

[Di] Dijkstra, E.W., “A Note on Two Problems in Connection with Graph Theory,” *Numerische Mathematik,* Vol. 1, pp. 269-271, 1959.

[DW] Dixon, M. and Weiner, G, “Automated Aircraft Routing Through Weather-Impacted Airspace”, *Fifth International Conference on Aviation Weather Systems,* Vienna, VA, 1993, pp. 295-298.

[FAA] Federal Aviation Administration, *1997 Aviation Capacity Enhancement Plan*, Federal Aviation Administration Office of System Capacity, Washington, DC, 1997.

[GoF] Gamma, E., Helm, R., Johnson, R., Vlissides, J., *Design Patterns,* 2nd Ed., Addison-Wesley, New York, NY, 2001.

[GMMN] Gewali, L., Meng, A., Mitchell, J. S. B., and Ntafos, S. (1990), “Path planning in 0/*1/*∞ weighted regions with applications," *ORSA Journal of Computing*, 2 (3), pp. 253-272.

[HKJ] Hoffman, B., Krozel, J., and Jakobavits, R., “Potential Benefits of Fix-Based Ground Delay Programs to Address Weather Constraints,” *AIAA Guidance, Navigation, and Control Conf.*, Providence, RI, Aug., 2004.

[JB] Jardin, M.R., and Bryson, A.E., “Neighboring Optimal Aircraft Guidance in Winds”, *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 4, pp. 710 – 715, July-Aug., 2001.

[JI] Juang, Y.C. Isaacson, D.R., “Design Concept and Development Plan of the Expedite Departure Path (EDP)”, *Aircraft Technology, Integration, and Operations Conf.,* Los Angeles, CA, Oct., 2002.

[KHLR] Kindel, R., Hsu, D., Latombe, J.C., Rock, S., “Kinodynamic Motion Planning Amidst Moving Obstacles”, *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 537-543.

[KHMSZ] Klosowski, J., Held, M., Mitchell, J.S.B., Sowizral, H., and Zikan, K., “Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs,” *IEEE Transactions on Visualization and Computer Graphics*, Vol. 4, No. 1, 1998, pp. 21-36.

[KJP] Krozel, J., Jakobovits, R., and Penny, S. (2006) “An Algorithmic Approach for Airspace Flow Programs”, *Air Traffic Control Quarterly*, 14 (3).

[KLM] Krozel, J., Lee, C., and Mitchell, J.S.B., “Estimating Time of Arrival in Heavy Weather Conditions,” *AIAA Guidance, Navigation, and Control Conf.*, Portland, OR, Aug., 1999.

[KMPP] Krozel, J., Mitchell, J.S.B., Polishchuk, V., Prete, J., “Capacity Estimation for Level Flight with Convective Weather Constraints”, submitted to *Air Traffic Control Quarterly* (est. July 2007).

[KPPM1] Krozel, J. Penny, S., Prete, J., Mitchell, J.S.B. (2007), "Automated Route Generation for Avoiding Deterministic Weather in Transition Airspace," *Journal of Guidance, Control, and Dynamics,* 30 (1), Jan./Feb.

[KPPM2] Krozel, J., Penny, S., Prete, J., and Mitchell, J.S.B., “Comparison of Algorithms for Synthesizing Weather Avoidance Routes in Transition Airspace,” *AIAA Guidance, Navigation, and Control Conf.,* Providence, RI, Aug., 2004.

[KPMSA] Krozel, J., Prete, J., Mitchell, J.S.B., Smith, P., Andre, A., “Designing On-Demand Coded Departure Routes”, *AIAA Guidance, Navigation and Control Conference,* 2006.

[KWH] Krozel, J., Weidner, T., and Hunter, G., “Terminal Area Guidance Incorporating Heavy Weather,” AIAA Paper 97-3541, Aug., 1997.

[KWSSMS] Klopfenstein, M., Wilmouth, G., Smith, P., Spencer, A., Mintzer, M., and Sud, V., “Congestion Management via Interactive Dynamic Flight Lists and Customer Submitted Multiple Routing Options”, *AIAA 5th Aviation, Technology, and Operations Conf.,* Arlington, VA, Sept., 2005.

[LMS] Lanthier, M., Maheshwari, A., and Sack, J.-R., “Approximating Weighted Shortest Paths on Polyhedral Surfaces”, *Proceedings of the 13th Annual ACM Symposium on Computational Geometry,* Nice, France, 1997, pp. 274-283.

[LSBB] Laudeman, I., Shelden, S., Branstrom, R., and Brasil, C., *Dynamic Density: An Air Traffic Management Metric,* Tech. Memorandum TM-1998-112226, NASA Ames Research Center, Moffett Field, CA, April, 1998.

[MFB] Mao, Z., Feron, E., Bilimoria, K., “Stability of Intersecting Aircraft Flows Under Decentralized Conflict Avoidance Rules”, *Proceedings of the AIAA Guidance, Navigation, and Control Conference,* August 2000.

[MFB] Mao, Z., Feron, E., and Bilimoria, K., "Stability of Inter- secting Aircraft Flows Under Decentralized Conflict Avoidance Rules," Proceedings of the AIAA Guidance, Navigation, and Control Conference, Denver,CO, August 2000

[MM] Mata, C., and Mitchell, J. S. B., “A new algorithm for computing shortest paths in weighted planar subdivisions,” *Proceedings of the 13th Annual ACM Symposium on Computational Geometry,* Nice, France, 1997, pp. 264-273.

[M1] Meyn, L. (2002), “Probabilistic Methods for Air Traffic Demand Forecasting," *AIAA Guidance, Navigation, and Control Conf.*, Monterey, CA, Aug.

[Mi1] Mitchell, J.S.B., “Shortest Paths and Networks,” Chapter 24 (pp. 445-466) in the CRC Handbook of Discrete and Computational Geometry, CRC Press LLC, (Jacob E. Goodman and Joseph O’Rourke, eds.), 1997.

[Mi2] Mitchell, J.S.B., “Geometric Shortest Paths and Network Optimization,” Chapter15 (pp. 633-701) in the Handbook of Computational Geometry, Elsevier Science (J. Sack and J. Urrutia, eds.), 2000.

[Mi3] Mitchell, J.S.B., “An Algorithmic Approach to Some Problems in Terrain Navigation,” *Artificial Intelligence*, Vol. 37, 1988, pp. 171-201.

[M2] Mitchell, J.S.B. (1990), “On maximum Flows in Polyhedral Domains,” *Journal of Computer and System Sciences*, Vol. 40, pp. 88-123.

[MP] Mitchell, J.S.B., and Papadimitriou, C.H., “The Weighted Region Problem: Finding Shortest Paths Through a Weighted Planar Subdivision”, *Journal of the ACM*, Vol. 38: pp. 18-73, 1991.

[MP] Mitchell, J.S.B. and Polishchuk, V. (2007), “Thick Non-Crossing Paths and Minimum-Cost Flows in Polygonal Domains,” *Proc. 23rd Annual Symposium on Computational Geometry*, ACM Press, to appear, 2007.

[MPK] Mitchell, J.S.B., Polishchuk, V., and Krozel, J. (2006) “Airspace Throughput Analysis Considering Stochastic Weather,” *AIAA Guidance, Navigation, and Control Conf.*, Keystone, CO, Aug.

[MGMK] Mogford, R., Guttman, J., Morrow, S., and Kopardekar, P., *The Complexity Construct in Air Traffic Control: A Review and Synthesis of the Literature*, Report DOT/FAA/CT-TN95/22, US Dept. of Transportation, Federal Aviation Administration, Atlantic City, NJ, July, 1995.

[Ni] Nilsson, N.J., *Principles of Artificial Intelligence*, Tioga Pub. Co., Palo Alto, CA, 1980, Chapter 2.

[PM] Prete, J., and Mitchell, J.S.B., “Safe Routing of Multiple Aircraft Flows in the Presence of Time-Varying Weather Data,” *AIAA Guidance, Navigation, and Control Conf.*, Providence, RI, Aug., 2004.

[RMM] Rodgers, M.D., Mogford, R.H., and Mogford, L.S., “The Relationship of Sector Characteristics to Operational Errors,” *Air Traffic Control Quarterly*, Vol. 5, No. 4, pp. 241-263, 1997.

[RP] Rhoda, D.A., and Pawlak, M.L., “The Thunderstorm Penetration/Deviation Decision in the Terminal Area”, *American Meteorological Society's 8th Conference on Aviation, Range, and Aerospace Meteorology*, Dallas, TX, 1999, pp. 308-312.

[RP] Rhoda, D.A., and Pawlak, M.L. (1999) *An Assessment of Thunderstorm Penetrations and Deviations by Commercial Aircraft in the Terminal Area*, M.I.T. Lincoln Laboratory Tech. Report NAS/A-2, Lexington, MA, June.

[RMM] Rodgers, M.D., Mogford, R.H., and Mogford, L.S. (1997), “The Relationship of Sector Characteristics to Operational Errors,” *Air Traffic Control Quarterly*, 5 (4).

[RR] Rowe, N.C., and Richbourg, R.F., “An Efficient Snell’s Law Method for Optimal-Path Planning across Multiple Two-Dimensional, Irregular, Homogeneous-Cost Regions,” *International Journal of Robotics Research*, Vol. 9, No. 6, 1990, pp. 48-66.

[RRZM] Richbourg, R.F., Rowe, N.C., Zyda, M.J., and McGhee, R.B., “Solving Global, Two-Dimensional Routing Problems using Snell’s Law and A* Search,” *IEEE International Conference on Robotics and Automation*, Raleigh, NC, 1987, pp. 1631-1636.

[RTCA] RTCA, *Report of the RTCA Board of Directors’ Select Committee on Free Flight*, RTCA, Inc., Washington, DC, Jan., 1995.

[S1] Schmidt, D. K. (1975), “On Modeling ATC Work Load and Sector Capacity,” *Journal of Aircraft*, 13 (7), pp. 531-537.

[SBSB] Smith, P.J., Beatty, R., Spencer, A. and Billings, C. , “Dealing with the challenges of distributed planning in a stochastic environment: Coordinated contingency planning”, *Proc. of the 2003 Annual Conf. on Digital Avionics Systems*, Chicago, IL, 2003.

[SWG] Song, L., Wanke, C., and Greenbaum, D. (2006), “Predicting Sector Capacity for TFM Decision Support,” *6th AIAA Technology, Integration, and Operations Conf.*, Wichita, KS, Sept.

[SKG] Sridhar, B., Kapil, S., and Grabbe, S., “Airspace Complexity and its Application in Air Traffic Management,” *2nd USA/Europe Air Traffic Management R&D Seminar*, Orlando, FL, Dec., 1998.

[S2] Strang, G. (1983), “Maximal flow through a domain,” *Mathematical Programming*, Vol. 26, pp. 123–143.

[SBL] Swenson, H., Barhydt, R., and Landis, M. (2006), *Next Generation Air Transportation System (NGATS) Air Traffic Management (ATM)-Airspace*

Project, NASA Tech. Report, Version 6.0, NASA Ames Research Center, Moffett Field, CA, June.

[WCGM] Wanke, C., Callaham, M., Greenbaum, D., and Masalonis, A. (2003), “Measuring Uncertainty in Airspace Demand Predictions for Traffic Flow Management Applications,” *AIAA Guidance, Navigation, and Control Conf.*, Austin, TX, August.

[WSZGM] Wanke, C., Song., L., Zobell, S., Greenbaum, D., and Mulgund, S. (2005) “Probabilistic Congestion Management,” *6th USA/Europe Seminar on ATM R&D*, Baltimore, MD, June.

[WHG] Weidner, T., Hunter, C. G., and Guadiamos, C. (1998), *Severe Weather Avoidance Pilot Survey,* Technical Report TR-98148-4-01, Seagull Technology, Inc., Los Gatos, CA.

[W] Wyndemere (1996), “An Evaluation of Air Traffic Control Complexity”, Final Report, NASA Contract No. NAS2-14284, Wyndemere Corp, Boulder, CA, Oct.

Glossary

A* Search – A heuristic depth-first search that expands nodes in an order such that the nodes that have the lowest estimated total solution cost are expanded first.

Airspace Domain – The region of space that represents the physical boundaries of the routing problem. Routes are not normally plotted outside of this region.

Constraints – All regions that aircraft are not permitted to fly through are constraints.

Flight – One single aircraft going from one place to another.

Free-Flight – Unplanned, distributed routing. Each individual pilot is responsible for routing his own aircraft while in free-flight and is not required to hold to a flight plan.

First-Come First-Served (FCFS) – Upon arriving in the transitional airspace each aircraft plots its own most-desirable route to a metering fix that does not conflict with any other flight's arrival time at the metering fix.

Flow – An airspace-routing unit larger than an individual flight. A flow consists of a route with a defined duration. Any flight that enters the beginning of the flow within the flow's entry window, and flies at the intended rate of speed to the end of the flow, is guaranteed to avoid hazardous weather, other flows, and aircraft outside of flows.

Flow-Based Route Planner (FBRP) – The routing algorithm and system used and discussed in this document. Its normal input is a defined airspace, the hazardous weather and other constraints in that airspace, and a series of arrival-departure point pairs; its normal output is a series of flows that provide safe routing between the arrival and departure points.

Flow Demands – A set of origin and destination points. Many flights want to travel from origin to destination; in flow-based routing, the response is to generate a series of flows between the two points.

Hazardous Weather – Loosely defined as any weather that an air traffic controller does not want an aircraft to fly through.

Hazardous Weather Safety Margin – Routes are required to maintain a certain distance between the route and hazardous weather, because of the inherent uncertainty of both weather and some aircraft positional systems.

Horizontal Separation Standard – Aircraft are required to maintain a certain minimum distance between themselves and other aircraft, because of the inherent uncertainty of older aircraft positional systems. This only applies to aircraft flying at similar altitude.

Metering Fix – Reference waypoints a short distance outside of each airport. Normally there is more than one per airport. Aircraft are required to pass through the metering fix as they land; once they pass the metering fix, they begin their final approach. The boundary to the transitional airspace routing problem; beyond this point the routing problem becomes a runway allocation problem.

Planned-Route – Aircraft are bundled into coming from, and going to, a number of arrival and departure points within a particular region of airspace. Then specific, long-term routes are plotted between all desired arrival-departure pairs such as to guarantee that there is no interference between any two routes. Every aircraft is then required to follow the plotted routes.

Protected Airspace Zone – The horizontal and vertical separation standards together describe a cylindrical "hockey puck", the PAZ, around all aircraft. Other aircraft are not permitted within.

Reflectivity Index – A measure of weather severity relying on the amount of light reflected by Earth's atmosphere.

Required Time of Arrival (RTA) – We treat the metering fix as having a number of distinct slots and fit each individual flight into them, guaranteeing maximal throughput and control at the metering fix and runway, given cooperative flight plans from the aircraft.

Route – A series of waypoints which an aircraft is supposed to fly to, one by one, in straight lines, in order to reach some desired destination.

Speed Profile – A description of what range of airspeeds an aircraft should be traveling at, at any given point in the airspace. Most experiments described in this document assume a constant speed profile for any given flight, flow, or route.

Time Slice – Weather data is normally stored as snapshots of individual moments in time. Each of these is a time slice.

Transitional Airspace – The transitional airspace directly around an airport, a circle with radius 200 nautical miles, is treated differently when discussing arrivals and departures to and from that airport.

Vertical Separation Standard – Aircraft are not permitted to be within 1000 feet of another aircraft at any time unless they fulfill the horizontal separation standard.

Waypoint – A point in space that is part of a route.