Algorithmic traffic abstraction and its application to nextgen generic airspace

Citation for published version (APA):

DOI:
10.2514/6.2010-9335

Document status and date:
Published: 01/01/2010

Document Version:
Accepted manuscript including changes made at the peer-review stage

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 30. Apr. 2021
Algorithmic Traffic Abstraction and its Application to NextGen Generic Airspace

Girishkumar R. Sabhnani, Arash Yousefi, Irina Kostitsyna, Joseph S. B. Mitchell
Metron Aviation Inc., Dulles VA, 20166
Stony Brook University, NY, 11794
University of Helsinki, FI-00014

We develop traffic abstraction algorithms that, given a set of 4D Trajectories (4DTs), extract the traffic structure in terms of standard flows and critical points (conflict and merge points). We demonstrate the application of our techniques to enable the NextGen generic airspace concept. We also analyze historical demand data to evaluate the level of abstraction underlying the en-route traffic within high-altitude sectors. Finally, we compare the structure of historical traffic to user preferred, wind optimal futuristic trajectories.

Nomenclature

G gradient.
H hessian.
4DTs 4D Trajectories.
ATC Air Traffic Controller.
CIT Controller Information Tool.
DSTs Decision Support Tools.
HITL Human in the Loop.
nm Nautical Miles.
NRS National Referencing System.
ZDC Washington ARTCC.
ZFW Fort Worth ARTCC.
ZOA Oakland ARTCC.

I. Introduction

Analysis of air traffic structures for a given set of flown or projected trajectories can serve multiple purposes. In particular, such analysis can be used for designing future airspace boundaries (e.g., sectors) and analyzing NextGen generic airspace concepts. As the current system gradually migrates to a satellite-based navigation system, and the National Referencing System (NRS) replaces the existing ground-based Navigational Aids, aircraft will fly more direct and user preferred trajectories and traffic will be more scattered throughout the NAS. The existing sector boundaries are designed primarily based on the existing airways and any airspace redesign requires analysis of futuristic traffic structures.

The concept of generic airspace has been proposed within the NextGen ConOps.¹,² By definition, such airspace can be managed by any controller from any facility with supporting procedures, structure, and
automation levels. This will enable interchangeability of resources among different facilities in the NAS and can potentially enhance the utilization of human controller resources. To enable such concepts, traffic characteristics, airspace structures, referencing systems, and control procedures must be generalized so that controllers can quickly understand and comprehend the traffic situation. Decision Support Tools (DSTs) need to be developed to assist in controller training and to expedite the time required for controllers to become familiar with traffic structure and local characteristics of sectors. Mogford et al. developed the Controller Information Tool (CIT) to study these issues and evaluate the benefit of DSTs in facilitating the goals of generic airspace. It was concluded that presenting the sector and traffic information is helpful to controllers before they start controlling on the D-side. Traffic abstraction and conflicting points will help controllers to quickly comprehend the traffic structure. Tools similar to CIT require traffic abstraction and structure including the coordinates of standard flows and conflicting areas as inputs. In highly dynamic heterogeneous future airspace, automated techniques are needed to provide these inputs to generic airspace DSTs.

In this paper we present two methods for producing traffic abstractions and present examples of their application in generic airspace. The organization of paper is as follows: In Section II, we discuss the important elements of abstraction that must be considered in model development. Section III gives the details of various traffic abstraction methods. Section IV presents applications of the methods to analyze the NextGen generic airspace concept and to serve as input for the Air Traffic Controller (ATC) decision support and training tools. In Section V, we compare the structure in today’s demand with the futuristic user preferred wind optimal demand. The conclusion and future work is discussed in Section VI.

II. Elements of Traffic Abstraction

Based on over-the-shoulder monitoring of controllers and analysis of historical traffic, Hansman et al. identified the following major factors in structure-based abstraction:

- **Standard Flows**: Standard flows are major flows of aircraft within a piece of airspace, such as a sector; the majority of the overall traffic flies along these standard flows. In today’s ground-based navigation system, standard flows are mostly placed along the jet routes.

- **Critical Points**: Critical points are the intersections of standard flows. At these points there is a higher probability for loss of separation. Controllers perform more tasks when aircraft fly close to critical points. Merge points and crossing traffic points are examples of critical points.

- **Groupings**: A grouping is a sequence of aircraft flying the same or similar heading and speed. To control a grouping, controllers only need to assign a certain heading and speed and require the aircraft to maintain their speed and heading until approaching a critical point, where additional instruction and conflict resolution actions may be needed. Therefore, controlling \( n \) aircraft within a grouping requires less effort than controlling \( n \) individual aircraft that are scattered across a sector.

Fig. 1 gives an example illustrating these abstractions. Our algorithms are designed to compute such an abstraction quickly and automatically in response to dynamic demand and resectorization.

Identifying standard flows can be thought of as a trajectory clustering problem. There is an excellent web survey covering different kinds of trajectory clustering problems. The most relevant kind of trajectory clustering of interest here is classified as repetition in Andrienko et al.’s book chapter also gives a good overview of the topic, along with some key applications beyond air traffic management. Recently, Zelinski et. al. studied the same problem of extracting elements of abstraction (critical points and major flows) from flight tracks. They first identified individual track merge and intersection points by analyzing the separation between pairs of flights. These merge and intersection points are clustered into nodes of a graph. Links are then placed between nodes to represent major traffic flows. This is a bottom up method that first

![Figure 1. Four Types of Structure-Based Abstractions (graphics taken from Hansman et al.)](image-url)
identifies the critical points and then uses those to define the dominant flows. In contrast, techniques discussed in this paper are top down, by which we directly identify the standard flows from tracks and then use them to define the critical points.

III. Traffic Abstraction Methods

We describe two methods to identify standard flows. Dykes et al.\(^9\) suggest that the traffic density heat map approach used by both of our methods dates back to the work of Hägerstrand in 1970.

Traffic Density Map. We first describe the construction of a traffic density heat map in the 2D (lat,lon) plane. We discretize the space of the bounding rectangle of (the 2D projection of) the trajectory set \( T \) with a uniform square grid \( G \). The grid spacing in \( G \) is defined by a parameter \( \text{incr} \) (in Nautical Miles (nm)). For each grid point \( g \), the weight \( w(g) \) is defined as the number of tracks that intersect a disk of radius \( \epsilon \) (in nm and \( \text{incr} \)), centered at \( g \). See Fig. 2(b) for an example of a traffic density map. The grid points are color-coded, with traffic density increasing from green to red. It should be easy to see that this construction extends to 3D (lat, lon, and altitude) and to 4D (lat, lon, altitude, and time).

1. Greedy Trajectory Clustering

This method addresses the following problem: Given a set \( T \) of \( n \) aircraft trajectories (polygonal paths in 2D), determine if there exists a set of standard flows \( D \subset T, |D| \leq k \), such that at least \( c\% \) (measured as the sum of length) of the trajectories in \( T \) lie “close” (within \( \epsilon \)) to at least one of the standard flows. A trajectory \( t \) is “close” to a standard flow \( d \) if a certain fraction \( f \), of its length \( l(t) \), lies within the \( \epsilon \)-fattening of \( d \) (see Fig. 2(a)). One may want to fix \( c \) and minimize \( k \), or fix \( k \) and maximize \( c \).

![Figure 2. Greedy Clustering results for ZDC.](image)

The greedy trajectory clustering method (see Algorithm 1) is inspired by the greedy approximation algorithm for set cover. We greedily select trajectories that “cover” many tracks and designate them as standard flows. The main assumption of this method is that there is at least one trajectory in the input set that exactly follows a standard flow, and can act as a representative for that flow.

To identify the “best” track in Algorithm 1, we use three heuristic described below.

- **GreedyT** considers a “best” track, \( t^* \), to be one with maximum coverage: \( t^* \leftarrow \arg\max_{t \in T, t \notin C} \text{cov}(t) \), where \( \text{cov}(t) = \sum_{\{t' \in T, t' \notin C, \text{and } t' \text{ “close” to } t\}} l(t') \).

Since GreedyT is computationally slow, we devise two approximations GreedyD and GreedyW which use the traffic density map to identify the “best” track. For each track \( t \), we define the weight \( w(t) \) as the sum of weights of all grid points that are within distance \( \epsilon \) (in the \( \epsilon \)-band) of \( t \). Note that this \( \epsilon \) is the
same as the disk radius used in the construction of the traffic density map. The density of \( t \) is defined as \( d(t) = w(t)/n(t) \), where \( n(t) \) is the number of grid points in the \( e \)-band of \( t \). Intuitively, the density is weight per unit length of the track and is a good quantifier for the dominance of \( t \). A track \( t' \) with a high \( d(t') \) goes through (or lies completely within) a region of high traffic density. Thus, \( t' \) is a good candidate for standard flow.

- **GreedyD** selects a track \( t^* \) with the maximum density. Since a track \( t \) with high density may have a very low weight, \( w(t) \), it may not be a good choice because it may cover very few tracks. Thus, **GreedyD** utilizes a constraint that the weight of the selected track be at least a specified fraction, \( \omega \), of the maximum track weight. Thus, \( t^* \leftarrow \text{argmax}_{t \in T \text{ and } t \notin C} \{w(t) > \omega \cdot w(t')\} \). Here, \( t' \) is the track with maximum density. Since a track \( * \) with high density may have a very low weight, \( w(t) \), it may not be a good choice because it may cover very few tracks.

- **GreedyW** selects a track \( t^* \) with maximum weight, under the constraint that \( d(t^*) \) is at least a specified fraction, \( \delta \), of the maximum track density. Thus, \( t^* \leftarrow \text{argmax}_{t \in T \text{ and } t \notin C} \{d(t) > \delta \cdot d(t')\} \). Here, \( t' \) is the track with maximum weight, under the constraint that \( d(t^*) \) is at least a specified fraction, \( \delta \), of the maximum track density.

Note that the traffic density map is updated in each iteration of Algorithm 1, for **GreedyD** and **GreedyW**, corresponding to the tracks in \( T \setminus C \). An instance of **GreedyW** output for 24hr historical demand for the Washington ARTCC (ZDC) is shown in Fig. 2(c)). The parameters used for this experiment are as follows: \( incr = 1 \text{ nm} \), \( \epsilon = 2.5 \text{ nm} \), \( f = 70\% \), \( c = 60\% \) and \( dc = 0.6 \). Refer to Sabhmani\(^{10}\) for more information about how these parameters affect the results of the method.

### Algorithm 1 Greedy Trajectory Clustering

**Input:** Set \( T \) of tracks, Coverage threshold \( c \).  
**Output:** Set \( D \) of standard flows. 
\[
D \leftarrow \phi \quad /\!* Covered tracks */\!
C \leftarrow \phi \quad /\!* Coverage */\!
\text{Coverage} \leftarrow 0 \\
\text{while} \ \text{Coverage} < c \% \ \text{do} \\
\quad t^* \leftarrow \text{"best" track} \\
\quad D \leftarrow D \cup t^* \\
\quad \text{for all} \ T \in T \ \text{do} \\
\quad \quad \text{if} \ t \notin C \ \text{and} \ t \text{ "close" to} \ t^* \ \text{then} \\
\quad \quad \quad C \leftarrow t \\
\quad \quad \text{end if} \\
\quad \text{end for} \\
\quad \text{Coverage} \leftarrow \sum_{t \in C} l(t)/\sum_{t \in T} l(t) \\
\text{end while} \\
\]

### 2. Ridge Top Detection

This method primarily works in 2D. It starts by building the 2D traffic density heat map, and considers the traffic density heat map’s values as the third dimension to create the ridge domain, as shown in Fig. 3(a)). Let a twice-differentiable function \( f : \mathbb{R}^2 \rightarrow \mathbb{R} \) represent the density heat map after passing through a Gaussian filter. The gradient \( \mathbf{G} \) of \( f \) is

\[
\mathbf{G} = \nabla f = \begin{pmatrix}
\frac{\partial f}{\partial x} \\
\frac{\partial f}{\partial y}
\end{pmatrix}.
\]

**G** points in the direction of maximum increase in \( f \). The directional derivative of \( f \) in (unit vector) direction \( \mathbf{U} \) is \( \mathbf{G}^T \mathbf{U} \). The hessian \( \mathbf{H} \) of \( f \) is

\[
\mathbf{H} = \nabla^2 f = \begin{pmatrix}
\frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\
\frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2}
\end{pmatrix}.
\]

The second derivative of \( f \) in direction \( \mathbf{U} \) is \( \mathbf{U}^T \mathbf{H} \mathbf{U} \). \( \mathbf{H} \) is symmetric and, thus, has two real eigenvalues, \( e_1 \) and \( e_2 \), with corresponding eigenvectors, \( \mathbf{V}_1 \) and \( \mathbf{V}_2 \), that are orthogonal.
At a ridge, $e_1 < 0$ and $V_1$ is orthogonal to the ridge. Also $G^T V_1 = 0$ and $V_2$ points along the ridge. Near a ridge, $G$ has a component in the direction of the ridge and $V_1$ points roughly towards the ridge or away from it. We identify the location of ridges by checking for these conditions in $f$.

The output of this process is illustrated in Fig. 3(b). Once the location of the ridges is identified we identify the standard flows based on the average heat value along the ridges. Basically, we filter out the ridges with low average heat values; see Fig. 3(c). Both Figs. 2(c) and 3(c) show the identified flows for ZDC using two different methods; the similarity of the results is apparent.

A main advantage of the ridge method is that it does not rely on individual tracks; if multiple tracks together define a standard flow, but none of them individually follows the standard flow exactly, this method is likely to identify that flow.

![3D heat map showing the ridges.](image1)

![All identified ridge tops.](image2)

![Cleaned up by average heat-value along the flows.](image3)

**Figure 3. Ridge top detection results for ZDC.**

3. **Identifying Critical Points**

A point where standard flows intersect can be one of the following: a) a merge point where two (or more) flows merge, b) a conflict point where two flows cross each other, or c) a diverge point where flows diverge. We consider all flow intersections as critical points in the traffic abstraction, even though a diverge point may be less critical for quantifying the ATC workload.

We follow a top-down method to identify the critical points: We first identify the standard flows, and then we identify the intersection points of two or more standard flows. This is in contrast with the bottom-up method, where intersections of individual tracks are clustered to identify critical points. For generic airspace purposes, we believe the top-down method is more relevant. A critical point, if present, should be translated back to the intersection of some standard flows. This is important because, a) it is unlikely that few aircraft approach a point in space-time from different directions without themselves belonging to some standard flow, and b) even if that is the case, it is not really a part of the underlying traffic structure, to which the ATC would want to fall back in case of high traffic scenarios. Note that the top-down method also considerably reduces the search space for finding the critical points, since the points of interest are only the ones at the intersections of identified standard flows.

For a critical point, it is important that the flows must intersect in 4D, i.e. aircraft flying along the intersecting standard flows should interleave in space-time for the point to be considered critical. For experiments in this paper, since we use the 2D implementation of above methods, it is important to post-analyze the flow intersection points to check for interleaving in altitude-time.

Consider an intersection point of two standard flows in 2D. Fig. 4(a) shows the altitude-time profile of all aircraft that fly in the vicinity (2.5nm) of this intersection point (in the 2D projection). The red points correspond to aircraft that fly along the first standard flow, the blue points correspond to the second flow, and the gray points correspond to aircraft that do not follow along either of the flows. Using an altitude threshold of 2000 feet and a time threshold of 15 minutes, we identify conflict points in the altitude-time graph as follows. For any pair of points corresponding to different flows (i.e., different color points), if both
the altitude and the time difference between the points satisfy the thresholds, we mark them as conflict points (shown as black points in Fig. 4(b)). Now, we can just count the number of these conflicts and weight the corresponding critical points. An alternate way to weight the critical point is to cluster these conflict points and identify a critical range for altitude-time (as shown by rectangles in Fig. 4(b)). The size of these rectangles (sum or max) can be considered as weights of critical points. We are currently investigating the use of the standard $k$-means clustering method to identify these critical intervals.

![Image](a) The altitude-time graph of all flights in the vicinity of an intersection point. 
(b) Clustering the conflict points (points along different flows, but close in altitude and time) to define the critical range (rectangles).

Figure 4. Analyzing the altitude time profile of flights in the vicinity of a flow intersection point in 2D.

IV. Application to Generic Airspace

NextGen ConOps\textsuperscript{1} introduces changes in airspace usage and design by dynamically resectorizing the airspace in response to changes in traffic that may arise from convective weather activities or changes in demand profile. As a result, controllers may constantly need to adjust to the new airspace and possibly may be assigned to a different position in the NAS. The concept of generic airspace\textsuperscript{12} is proposed in NextGen to facilitate this shifting of positions.

Even though certain local and site-specific procedural and referencing knowledge is not transferable, it is possible to abstract some features that make a piece of airspace a good candidate for generic airspace. Presence of structure enables the controllers to develop abstract mental models of traffic, and this enhances situational awareness. A high level of situational awareness results in lower airspace complexity and controller workload, which in turn reduces the required training time for the controllers. Hence, a piece of airspace is a good candidate for generic airspace if the majority of its traffic is flying along a highly structured pattern. Here, we use our flow abstraction methodologies to identify sectors with highly structured traffic patterns.

A. Analysis of Traffic Abstraction for Generic Airspace

**Experimental Setup**

We analyze 36 high-altitude sectors (with at least 10,000 feet overlapping within altitude range 24,000 – 99,900 feet) in the ZDC, the Oakland ARTCC (ZOA), and the Fort Worth ARTCC (ZFW). We use historical flight data from May 1, 2008, a day with no severe weather disruption.

The day is divided into 12 two-hour time intervals, and the demand in each individual sector-time-interval pair is analyzed to be a candidate for generic airspace. For the greedy trajectory clustering method, standard flows are restricted to have at least 6 tracks in its vicinity. The choice of 6 tracks following the path of any standard flow translates to having 1 flight (on average) every 20 minutes. In addition, all flows with length less than 10 nautical miles were eliminated as outliers. The same experiment was repeated with 6 four-hour time intervals and the threshold for number of tracks in the vicinity of a standard flow was set to 12. We consider all intersection points of the standard flows (projected to 2D) to be critical points.

**Results**

For each sector, we discard the time intervals that have less than 15% of demand (total tz-hit counts) as
compared to the peak demand time interval (for the same sector). This is done because controllers, regardless of unfamiliarity of the new airspace, should be able to manage such sparse traffic demand.

Fig. 5 shows histograms of the number of sector-time-intervals with respect to the number of standard flows (left), with respect to the number of critical points (middle), and with respect to the ratio of traffic along the standard flows (right). (The ratio of traffic is defined as the sum of track lengths along the flows over the total length of all tracks.)

It is interesting to see that for most sector-time instances (for both two-hour and four-hour intervals) there are $1 - 4$ standard flows, $0 - 5$ critical points, and for almost half of the instances, $50\%$ or more of the traffic followed the standard flows. This strongly suggests that there is enough structure in the demand that can be extracted and presented to the ATC. This is not surprising, given that the historical (and present day) demand is already structured along the jet routes. Later, in Section V, we compare the underlying structure in current demand with the underlying structure in projected futuristic demand.

Also, for the two-hour time-interval experiment, $31.15\%$ of the instances (apart from the $95$ sparse traffic instances) have $60\%$ or more of the traffic along the identified standard flows, including $20.77\%$ that have $6$ or fewer critical points. These numbers suggest that a significant chunk of airspace (in 4D) has very structured traffic and can be considered as suitable for being regarded as generic airspace.

Fig. 6 shows the variation in these abstraction elements for different two-hour time intervals for two specific sectors: ZDC19 and ZFW49, over the entire day. For both sectors, we also include a plot of the ratio of demand (total tz-hit counts) in each individual time interval to the peak demand time interval for that sector. This explains why certain time intervals had no standard flows and critical points. Both the number of standard flows and the number of critical points is less in ZDC19 as compared to ZFW49. Also the ratio of traffic along the standard flows is much high for ZDC19. These results indicate that traffic in ZDC19 is more structured as compared to ZFW49. This information can be used in identifying candidates for generic airspace in NextGen.

**B. Input for ATC Decision Support and Training Tools**

Traffic abstractions can assist the ATC decision support and training tools by presenting a high-level structure in the projected demand. Auto-traffic abstraction methods are important in highly dynamic demand environments, like in case of weather events etc., since the demand structure changes significantly from the current operations. Such methods can quickly set up the visualization tools that assist ATC to develop the
Figure 6. Traffic abstraction for two sectors for different 2hr time windows for the entire day.

Mogford et al.\textsuperscript{3} developed the CIT to facilitate controller training, and to enable rapid realization of important attributes of sectors before controllers actually start controlling the traffic in an unfamiliar sector. CIT provides easily accessible sector information, and enables the controllers to display critical information while hiding less important items. The usefulness and usability of the CIT was evaluated by Human in the Loop (HITL) experiments.\textsuperscript{3}

Fig. 7(a) shows the CIT’s 3D view (as used in HITL) with major flows and critical points that were identified by interviewing the controllers for a specific sector. Fig. 7(b) illustrates the same sector with algorithmically generated traffic abstraction. Standard flows and critical points are marked with ribbons and dots so that controllers can easily grasp the overall picture of the traffic situation. For HITL simulations, the location of critical points and standard flows was solicited from the controllers during interviews and manually plotted on the CIT screen. We algorithmically extracted the standard flows and critical points using a historical trajectory data for a clear weather day. It is apparent from the figure that our automated
algorithm was able to generate an image very similar to that generated though controller interview and manual plotting. (An exact correspondence was not possible because we did not know the precise time period for which the CIT image was generated.) Such an automation may facilitate rapid preparation of input data for future tools like CIT.

V. Historical vs Futuristic Demand Structure

Next, we compare the historical demand structure where the trajectories more or less comply to the underlying jet routes, with the predicted futuristic demand. The predicted futuristic demand represents 2.0X (approximately twice the number of flights) wind-optimal user-preferred trajectories. We restrict the comparison to only the 18 high-altitude ZFW sectors with the demand split up into 12 two-hour intervals. Keeping the other parameters of methods the same (as in Section IVA), we just double the threshold (12 flights) for the standard flow identification methods, in order to account for 2.0X traffic.

Fig. 8 shows different metrics for the comparison. Out of 18 × 12 = 216 sector-interval pairs, 167 such pairs for the historical track data and 154 pairs for the futuristic data were considered interesting, as they had more than 15% of the peak traffic within each sector. Furthermore, the number of standard flows increased whereas the number of critical points decreased significantly for the futuristic data and the percent traffic along the flows remained similar. These observations are consistent with the intuition that by removing the constraint of ground-based navigation systems, aircraft will be more scattered throughout the NAS when they fly their preferred wind-optimal trajectories. Note that, a significant proportion of demand complies
with the dominant flows in both historical and futuristic demand.

VI. Conclusion and Future Work

Algorithmic traffic abstraction methods are needed to enable dynamic sectorization and support the NextGen generic airspace concept. We developed two methods for extracting elements of abstraction (standard flows and critical points) from 4D flight trajectories. We examined the application of these methods to identify sectors with highly structured traffic and rank sectors based on their level of traffic abstraction. The abstraction methods can be applied to any piece of airspace; thus, they are also useful as the sector boundaries change dynamically within NextGen. Furthermore, we applied the abstraction methods to both historical and futuristic track data and compared their underlying structure.

Future work may include comparison of our methods to other traffic abstraction methods including bottom-up methods where critical points are identified first and major flows are created by connecting critical points.

Elements of abstraction extracted using our methods can be used within sector design algorithms. The critical points and dominant flows can be weighted based on their traffic density. Sectorization methods can then use these weights to balance the overall critical point weights instead of only their number. Other factors such as the direction and average speed of flights along the major flows could also be useful to include constraints within sectorization algorithms to enforce placement of the critical points far enough from sector boundaries.

Acknowledgment

This research was sponsored by NextGen Airspace Program at NASA Ames Research Center under order number NNA08BA50D. We wish to thank Ms. Shannon Zelinski at NASA Ames Research Center for her support and contribution to this research. Joseph Mitchell is partially supported by the National Science Foundation (CCF-0729019 and CCF-1018388). Valentin Polishchuk is partially supported by the Academy of Finland grant 118653 (ALGODAN). Authors also wish to thank Dr. David P. Kierstead of Daniel H. Wagner, Associates for his contribution to this research.

References