Proceedings of the American Control Conference
Anchorage, AK May 8-10, 2002

Automation Procedures for Air Traffic Management: A Token-based Approach

S. Devasia [1] M. Heymann [2] G. Meyer [3]

Abstract

A *token-based* Air Traffic Management paradigm is presented that enables the de-coupling of: (1) the Air Traffic Control System (ATCS) task to maintain safety; and (2) Airline Operational Center (AOC) goal to choose optimal flight paths. Under the token-based paradigm, each local area (sector) issues tokens that permit entry to that region of space over an assigned period of time. The number of tokens is limited by the ATCS to guarantee admission and safe passage to the destination point in the sector within a pre-specified time-interval. The token-based paradigm allows AOCs to freely trade the tokens between different aircraft and to choose desirable flight paths by choosing the sequences of sectors (as long as the AOC can procure the needed tokens). Although, the aircraft is forced to fly along established routes in each sector, the flexibility in choice of sectors may meet the AOC's needs for freedom.

1 Introduction

A critical challenge in Air Traffic Management (ATM) is to provide Airline Operational Centers (AOCs) the freedom to choose flight routes while ensuring safety. Lack of flexibility in choice of flight routes implies that AOCs are unable to adapt and optimize operations to accommodate emerging situations like changing weather patterns, missed connections, and enroute traffic congestion — this can substantially increase operational costs. To provide more freedom to AOCs, the Federal Aviation Administration (FAA) is considering a shift to the *Free Flight Paradigm* for ATM. The exact structure of the Free Flight Paradigm is currently being debated and can range from (a) complete freedom in choosing flight trajectories to (b) freedom in choosing flight segments along established route structures. However, the free-flight paradigm also increases the complexity of maintaining safety — a task performed by the Air Traffic Control System (ATCS). Therefore, automated ATM approaches are needed to handle the complex inter-dependence of safety and flexibility.

[1]Mechanical Eng. Dept., U. of Washington, Seattle, WA 98155; Email: devasia@u.washington.edu,

[2]Dept. of Computer Science, Technion, Haifa 32000, Israel; Email: heymann@cs.technion.ac.il

[3]NASA Ames Research Center, MS 210-10, Moffett Field, CA 94035; Email: gmeyer@mail.arc.nasa.gov

We present a *token-based* ATM paradigm that enables the de-coupling of the two main ATM tasks: (1) maintaining safety at the local level that can be assigned to the Air Traffic Control System (ATCS) and (2) optimal choice of the flight paths, which is the goal of Airline Operational Centers (AOCs). Under the proposed paradigm, each local area (sector) issues tokens that permit entry to that region of space over an assigned period of time. The number of tokens is limited by the ATCS to guarantee admission to the sector during the alloted time-span, and safe passage to the destination point (output point) in the sector within a pre-specified time-interval. This paradigm allows AOCs to freely trade the tokens between different aircraft and to choose desirable flight paths consisting of sequences of these regions (as long as the AOC can procure the needed tokens). Thus, the proposed paradigm decouples the ATCS's task of maintaining safety from the AOCs optimization objectives; and enables a type of free-flight that may be acceptable to the AOCs.

The article discusses two key aspects of the proposed paradigm: (1) the release of tokens by ATCS such that aircraft safety can be maintained (through conflict resolution and scheduling procedures); and (2) token-based choice of flight paths by the AOCs.

2 Token-based Approach

If AOCs are allowed to freely choose flight segments, then there is a possibility that too many aircraft could come to a specified sector leading to congestion, similar to rush hour slow-downs in freeways. Congestion can cause significant safety problems in ATM. To avoid such congestion, currently ATCS controls the overall flight plans of aircraft. Current approaches have limited flexibility; it is challenging for AOCs to negotiate with each other to choose optimal flight plans.

To allow AOCs the flexibility of choosing flight segments while limiting the rate of arriving aircraft at a sector, we propose that each sector issue tokens for aircraft that is valid over a specified time interval — the number of tokens issued depends on the capacity of the sector. Limiting the aircraft admitted to a sector to those with valid tokens allows the sector ATCS to maintain safety. On the other hand, AOCs are allowed

0-7803-7298-0/02/$17.00 © 2002 AACC

the freedom to choose (or change) flight segments provided they have sufficient tokens for the chosen routes. Such a token-based paradigm allows AOCs to freely trade the tokens between aircraft, and thereby have the flexibility to optimally handle changing operating conditions. Thus, the token-based paradigm decouples safety (ATCS task) from optimization (AOC goal).

2.1 AOC and the Token-Based Approach

As in current ATM paradigm, let the flight plan of the aircraft be composed of movements through a sequence of sectors in the airspace (see Figure 1); where each sector is controlled locally to maintain safety. AOCs should be allowed to choose (or change) a flight plan if sufficient tokens can be procured — the number of tokens needed is discussed, next.

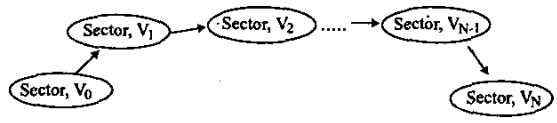

Figure 1: Flight Plan as a Series of Sectors

Let the flight plan require an aircraft to pass through sectors $V_0, V_1, V_2, \ldots V_N$ (see Figure 1). Then the arrival time interval $T(V_k)$ at any sector (V_k, $k = 1, \ldots, N$) can be determined if the time-for passage $t_p(V_k)$ and maximum anticipated delay $\delta(V_k)$ in each sector is known apriori as

$$T(V_k) = (T_k, T_k + \Delta_k) \quad (1)$$

$$\text{where } T_k := \sum_{i=0}^{k-1} t_p(V_i); \quad \Delta_k := \sum_{i=0}^{k-1} \delta(V_i). \quad (2)$$

The AOC needs to procure tokens for access into each sector V_k during the anticipated arrival time interval $T(V_K)$. The ATCS may limit the number of tokens issued based on safety considerations (discussed in Section 3); however, the AOCs should be allowed to directly negotiate and exchange tokens with each other to optimize their operations.

To enable the token-based paradigm it is necessary to quantify the following two characteristics of each sector: (1) the maximum capacity of the sector; and (2) the maximum delay accrued when passing through the sector.

3 Sector Capacity and Delay

The number of tokens issued by ATCS for a sector should be such that safety can be assured. The goal is to quantify the maximum number of aircraft that can be handled in the sector (i.e., the sector capacity) and the maximum delay (the uncertainty) that could be added when passing through the sector.

Sector capacity and delay depends on the procedures used for conflict resolution and aircraft scheduling. Several works have focused on the development of automated systems for air traffic control. For example, automation tools to schedule and resolve conflicts for aircraft arriving at an airport terminal has been developed by researchers at NASA Ames Research Center [1]. These are applicable for terminal radar approach control (TRACON) and for air route traffic control center (or Center). Researchers are also developing automation tools to resolve enroute conflicts in Centers under the free flight paradigm (see, e.g., [1, 2, 3, 4, 5, 6]). In contrast, the current article studies conflict resolution and scheduling on established routes. Simplified models and example sectors are used to quantify capacity and delay in automated ATCS; however, the proposed procedures can be extended to general sectors (see, e.g., [7]).

3.1 Sector Description

Sector V We begin by spatially discretizing a given sector and its routes. Let the routes in a sector be considered as a directed graph (digraph [8]) of order N_V with vertex set

$$V = \{v_1, v_2, \ldots v_{N_V}\}$$

and set E of directed edges of the from $\alpha = (v_i, v_j)$ where v_i is called the initial vertex and v_j is called the terminal vertex of the edge α.

Assumption 1 *It is assumed that there is at most one edge between any two vertices.*

Output and Input Vertices The set V_O of vertices with zero *outdegree* (where outdegree is the number of edges issuing from a vertex [8]) is called the output set V_O which is enumerated as $V_O = \{v_{o,1}, v_{o,2}, \ldots v_{o,N_o}\}$. Similarly, a set V_I of vertices with zero *indegree* (where indegree is the number of edges entering a vertex) is called the input set V_I.

Subgraphs of V The set $\mathcal{V}_O$ is defined as the set of all subgraphs of the above directed graph with the same vertex set V, such that (1) the outdegree of each vertex is at most one and (2) the set of vertices with zero outdegree (i.e., the output set of the subgraph) is the same as the output set V_O of the original graph V. It is noted that the edge set $\bar{E}$ of any subgraph $\bar{V}$ in $\mathcal{V}_O$ is a subset of the edge set E of the original directed graph V, i.e., $\bar{E} \subseteq E$.

State-Transition Map The state-transition map $A(\bar{V})$ of a subgraph $\bar{V} \in \mathcal{V}_O$ is defined as the matrix $A(\bar{V}) = a_{i,j}, (i, j = 1, 2, \ldots, N_V)$ such that $a_{i,j} = 1$ if

(v_j, v_i) is an edge of $\bar{V}$; otherwise $a_{i,j} = 0$. The set of state-transition maps of subgraphs in $\mathcal{V}_O$ is defined as $\mathcal{A}_O$. (It is noted that the state-transition map is the transpose of the adjacency matrix of a graph [8].)

Server State At any time step K, the state of the server is denoted by a column $(N_V \times 1)$ vector

$$X(K) = [x_1(K)\ x_2(K)\ \ldots\ x_{N_V}(K)]^T \quad (3)$$

where $x_j(K)$ represents the number of aircraft occupying vertex V_j.

Assumption 2 *An aircraft occupies a vertex of the graph V at each time step K and moves through one edge during each time step. Furthermore, it is assumed that there are no conflicts if the value of any row of the server state $X(K)$ is less than or equal to one, i.e.,*

$$\| X(K) \|_\infty = \max_{j=1,2,\ldots,N_V} |x_j(K)| \leq 1.$$

Remark 1 *This assumption can be changed to aircraft occupying a set of vertices at every time step, for example, to account for aircraft with different speeds [9]*

State Equation Given the server state $X(K)$ at time step K, and a subgraph $\bar{V} \in \mathcal{V}_O$ the server state is uniquely mapped into the next time step as

$$X(K+1) = A(\bar{V})X(K) \quad (4)$$

where $A(\bar{V}) \in \mathcal{A}_O$ is the state-transition map of $\bar{V}$

3.2 Aircraft Flow Through the Sector

Agent Types Aircraft entering the sector is assigned an agent type based on the input vertex and output vertex. Each agent type $AT(k), k = 1, 2, \ldots, N_{AT}$ is of the form $AT(k) = (v_{in,k}, v_{out,k})$ with the input vertex $v_{in,k} \in V_I$ and the output vertex $v_{out,k} \in V_O$. The set of agent-types is denoted by $\mathcal{AT}$.

Input Map The input-map B_k for agent type $AT(k)$ represents the vertex at which the aircraft arrives into the sector. The input-map B_k is the server state X with a one at the row corresponding to the input vertex $v_{in,k}$ and zero elsewhere.

Assumption 3 *Each agent type in the set $\mathcal{AT}$ has a distinct input vertex. The set V_I of input vertices is enumerated as $V_I = \{v_{i,1}, v_{i,2}, \ldots v_{i,N_I}\}$.*

Remark 2 *This can be achieved by splitting the input stream upon entry into the sector such that aircraft seeking different output vertices are in different queues.*

Agent-types for a given Output Vertex Given an output vertex, $v_{o,k}$, let the index of Agent-types with output vertex $v_{o,k}$ be enumerated as $l(v_{o,k}, m), m = 1, \ldots, N_{v_{o,k}}$. Here $N_{v_{o,k}}$ is the number of Agent-types with output vertex $v_{o,k}$. In the above example, $N_{v_{n+1}} = n$

Expected Arrival Time (ETA) The expected time of arrival (ETA) of aircraft is used to define an ETA-input $ETA(K) = eta_m(K),\ m = 1, 2, \ldots, N_{AT}$ which defines the arrival of different agent types at the sector boundaries

$$\begin{aligned} eta_m(K) &= 1 \quad \text{if an aircraft of agent type } AT(m) \\ &\quad\ \ \text{has an expected time of arrival } K \\ &= 0 \quad \text{otherwise} \end{aligned}$$

N-Density for an Output Vertex For a positive integer N, the density δ of the ETA input for output vertex $v_{o,k}$ is defined as the maximum number of aircraft wanting to exit at $v_{o,k}$ with expected time of arrival in any N-time interval, i.e.,

$$\delta(N, v_{o,k}) = \max_{K=1,2,\ldots} \left(\sum_{m=1}^{N_{v_k}} \sum_{j=0}^{N-1} eta_{l(v_{o,k},m)}(K+j) \right)$$

It is noted that the N-density is always less than or equal to N times the number N_{v_k} of agent-types with v_k as the exit vertex, i.e., $\delta(N, v_k) \leq N N_{v_k}$.

M-modified Scheduled Arrival Times (STA) Given an ETA-input $(ETA(\cdot))$, a M-shifted STA input is defined as a rearrangement of the ETA-input with the arrival time of each aircraft delayed between 0 and M time steps (with only one aircraft of each agent type arriving at any time instant). The rearranged input stream is denoted by $STA_M(\cdot) = sta_m(K),\ m = 1, 2, \ldots, N_{AT}$ with

$$\begin{aligned} sta_m(K) &= 1 \quad \text{if an aircraft of agent type } AT(m) \\ &\quad\ \ \text{has a scheduled time of arrival } K \\ &= 0 \quad \text{otherwise} \end{aligned}$$

3.3 The Conflict Free Scheduling Problem

Given an ETA-input $(ETA(\cdot))$ the M-delay conflict free scheduling problem is defined as finding (1) a M-modified STA-input $STA_M(\cdot)$, and (2) a sequence from the subgraphs $\bar{V}(\cdot)$ from the set $\mathcal{V}_O$ such that the following three conditions are satisfied. In the following, the movement of aircraft from each agent-type through the sector is described by (for each $m = 1, 2, \ldots, N_{AT}$) for all $K = 1, 2, \ldots$

$$\begin{aligned} X_m(0) &= 0 \\ X_m(K+1) &= A[\bar{V}(K)]X_m(K) + B_m sta_m(K) \end{aligned}$$

1. The aircraft move through the sector without conflicts

$$\| \textstyle\sum_{m=1}^{m=N_{AT}} X_m(K) \|_\infty \leq 1 \text{ for all } K = 1, 2, \ldots$$

2. An agent-type may not exit the sector from the wrong exit vertex,

$$y_m(K) = C_m X_m(K) = 0 \text{ for all } K = 1, 2, \ldots;$$

where C_k is the wrong-output-vertex map C_k for agent type $AT(k)$; it is the transpose of the server state X with zeros everywhere except for ones at the row corresponding to the all the output vertices in V_O which are different from the agent-type's output vertex $v_{out,k}$.

3. There exists a number N^* such that any aircraft will pass through the sector within N^* time-steps.

3.4 Examples

We consider two examples to illustrate capacity and delay associated with (1) intersecting flows; and (2) merging flows.

Example 1: Intersecting Flows

Consider the following example sector shown in Figure 2. The vertex-set for this sector is $V = \{v_1, v_2, \ldots, v_8\}$, its edge set is

$$\begin{aligned} E = \{&\{v_1, v_3\}, \{v_1, v_4\}, \{v_2, v_3\}, \{v_2, v_5\}, \\ &\{v_3, v_5\}, \{v_3, v_5\}, \{v_4, v_6\}, \{v_4, v_7\}, \\ &\{v_5, v_6\}, \{v_5, v_8\}, \{v_6, v_7\}, \{v_6, v_8\}\}, \end{aligned} \quad (5)$$

and with output set $V_O = \{v_7, v_8\}$.

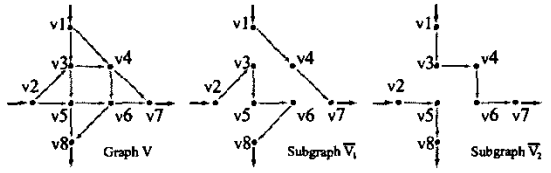

Figure 2: Intersecting Flows

Two subgraphs $\bar{V}_1, \bar{V}_2$ are also shown in Figure 2, which have the following state-transition matrices.

$$A(\bar{V}_1) = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0 \end{bmatrix} \quad (6)$$

$$A(\bar{V}_2) = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0 \end{bmatrix} \quad (7)$$

Solution to Example 1

The scheduling problem can be solved for any ETA (without rescheduling) with each aircraft passing through the sector in four time-steps as follows.

$$\begin{aligned} X_m(0) &= 0 \\ X_m(K+1) &= A(K)X_m(K) + B_m eta_m(K) \\ &\quad \text{for all } K = 1, 2, \ldots \end{aligned} \quad (8)$$

where

$$\begin{aligned} A(K) &= A(\bar{V}_1) \quad \text{if } K \text{ is an odd integer} \\ &= A(\bar{V}_2) \quad \text{if } K \text{ is an even integer} \end{aligned} \quad (9)$$

It is noted that the intersection requires additional space around the intersection point, and also requires changes in speed (because distance between vertices v_1 and v_3 may not equal the distance between vertices v_1 and v_4.

Solution to General Intersections

The intersection of multiple flows can be solved, for example, by rearranging them such that only two flows intersect at any point in space provided sufficient space is available for such re-arrangements (i.e., the density of intersections allows such re-arrangements). Then each intersection can be solved using the scheme in Example 1. Thereby, the solution to the two intersecting flows example can be extended to generalized intersections. Thus, general intersections can be rearranged to allow aircrafts pass through the sector with zero-delay. The maximum capacity of each agent-type in the sector is the rate at which they can exit the server, i.e., one per time step.

Example 2: Merging Flows

Consider the example sector shown in Figure 3. The vertex-set for this sector is $V = \{v_1, v_2, \ldots v_{n+1}\}$, its edge set is

$$E = \{\{v_j, v_j\}, \{v_j, v_n + 1\},\ j = 1, 2, \ldots, n\},$$

with output set $V_O = \{v_{o,1} = v_{n+1}\}$.

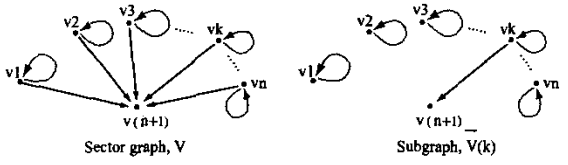

Figure 3: Merging Flows

We define n subgraphs with each subgraph $\bar{V}_k$ allowing aircraft in vertex v_k to go through while others are held at the current location as shown in Figure 3; the general subgraph $\bar{V}_k$ has the following state-transition matrix (for all $k = 1, 2, \ldots, n$)

$$
\begin{aligned}
A(\bar{V}_k) &= a_{i,j}(k), (i,j = 1,2,\ldots,n+1) \quad (10)\\
a_{i,j}(k) &= 1 \text{ if } i = j,\ j \neq k, \text{ and } j \neq n+1\\
&= 1 \text{ if } i = n+1,\ j = k\\
&= 0 \text{ otherwise}
\end{aligned}
$$

Solution to Example 2
The central idea is to hold all aircraft and let the aircraft with the lowest expected time of arrival pass through the merge vertex v_{n+1} (if there are multiple such aircraft then the one on the vertex with the lowest index is allowed to pass first).

If the $N_{v_{o,k}}$-density of the ETA input is less than or equal to $N_{v_{o,k}}$ then there exists a $(N_{v_{o,k}} - 1)$-delayed STA input such that each aircraft pass through the sector with at most $(N_{v_{o,k}} - 1)$ delay. The nominal time through the sector is 3 (for this particular example). The $(N_{v_{o,k}} - 1)$-delayed STA input is found from the ETA input by delaying an aircraft in agent-type i (which seeks to enter the sector at vertex v_i and leave the sector at vertex v_{n+1}) till vertex v_i becomes empty. If vertex v_i is empty then the aircraft is allowed to enter the sector at vertex v_i. At any time K let the S_{ETA} denote the set of vertices between 1 and n which have aircraft with the smallest arrival time (K). Then, the flow in the sector is chosen as

$$
\begin{aligned}
X_m(0) &= 0\\
X_m(K+1) &= A(K)X_m(K) + B_m sta_m(K)\\
&\quad \text{for all } K = 1,2,\ldots \quad (11)
\end{aligned}
$$

where

$A(K) = A(\bar{V}_k)$ where $1 \leq k \leq n$ is the lowest index of vertices in S_{ETA}; *and* k = 1 if S_{ETA} is empty

Remark 3 *The merge solution can be generalized, as long as the number of vertices between the input and the output vertices $v_{o,k}$ is the same for all agent-types with the same output vertex. This can be accomplished, for example, by changing speeds, by modifying the routes, or by re-defining the entry point to the sector at which the ETA to the sector is defined.*

Remark 4 *The above example illustrates a fundamental limitation of the merge — it always introduces uncertainty in the time-needed to pass through a sector. Such uncertainty is unavoidable even when speed changes are allowed, and is inherent to a merge.*

Capacity and Delay in a General Sector
Given input and output vertices and a set of agent-types, let routes be established in the sector such that

1. Each agent-type has a different input vertex.
2. Agent-types with different output vertices do not merge inside the sector.
3. Agent-types with the same output vertex merge at a single vertex in the sector such that the number of vertices between the merge point and the input vertex is the same for all Agent-types arriving at that merge-point.
4. All intersections are of the form in Example 1.

Then, the sector capacity is determined by the number of aircraft attempting to pass through a given output-vertex $(N_{v_{o,k}})$ as

$$\delta(N_{v_{o,k}}, v_{o,k}) \leq N_{v_{o,k}} \quad (12)$$

If the capacity constraint is not violated then the maximum delay $D(v_{o,k})$ for an aircraft with output vertex $v_{o,k}$ is determined also by the number of aircraft attempting to pass through a given output-vertex $(N_{v_{o,k}})$ as

$$D(v_{o,k}) = N_{v_{o,k}} - 1 \quad (13)$$

which is the uncertainty in passing through the sector. It is noted that $D(v_{o,k}) = 0$ if there are no flows merging into the output vertex $v_{o,k}$.

Remark 5 *The solution to the merge uses an M-modified STA-input that is obtained by rescheduling the expected time of arrivals — each expected arrival time could be delayed by upto M-steps. Such rescheduling implies that each input stream into the sector has M buffers before entry into the server. Such buffers can be developed using loops, speed changes, or by increasing the flight paths.*

Maximum Number of Tokens Issued If a sector satisfies the above four conditions, then the tokens issued by the sector must be such that the number of arriving aircraft satisfies Eq. 12. The maximum uncertainty in the passage time through the sector (i.e., delay in each sector) can be quantified in terms of the number of aircraft arriving at the same time (Eq. 13).

4 Discussion

Decoupling Safety and Flexibility The major advantage of using the proposed token-based approach is the decoupling of safety and optimization. The number of tokens issued can be used by the ATCS to limit the number of aircraft arriving at a sector in a given period of time to enable guaranteed safety. However, the ability to trade tokens between aircraft allows AOCs to optimally adapt to changing operational conditions.

Free Flight by Choosing Sectors The token-based paradigm can allow AOCs to choose a flight route by choosing the sequence of sectors the aircraft passes through (provided it can acquire the necessary tokens). Although, the aircraft is forced to fly along established routes in each sector, the freedom in choice of sectors may meet the AOC's needs for flexibility. Furthermore, if ATM is automated, then the number of routes could be increased in each sector, and the routes could be optimized to account for changing weather patterns.

Intersections Vs. Mergers The examples illustrate that intersecting flows could be resolved without addition of uncertainty in the time needed to pass through a sector (provided speed changes are allowed). In contrast, merging flows always lead to uncertainty in the time needed for passage through a sector — even if speed changes are allowed. The uncertainty depends on the number of flows merging into a single flow. Such uncertainty will exist also in continuous flows of aircraft, and is unavoidable because of simultaneous entry of multiple aircraft that want to merge into the same flow.

The uncertainty in travel time grows linearly with the number of mergers. Merging is often used because reduction in the number of flows entering a sector tends to make ATM problem easier to handle (such mergers also occur in the current hub-and-spoke approach). However, it is advantageous to avoid mergers in ATM whenever possible to reduce uncertainty.

Capacity and Tokens If the capacity of the sector is exceeded then the token-based approach will fail unless excess buffer capacity is available in each sector. Or in other words, a fraction of the capacity should be retained to handle emergencies. The capacity of a sector may also change due to changing weather conditions. In such cases, re-routing of aircraft and cancelations of flights may be unavoidable. Under the proposed token-based paradigm the ATCS could reduce the value of each token (e.g., by half) and thereby reduce the total number of tokens. The approach, however, empowers AOCs to negotiate with each other for the remaining tokens. While current ground-hold policies can be extended to accommodate such negotiations, the token-based approach provides a vehicle for such negotiations.

Future Work In this article speed changes and simplified models of sectors were used to quantify capacity and delays of sectors. Such quantification of capacity and delay is needed for the token-based approach. Such investigations should also be performed without speed changes (for protocols that only use changes in headings). Similarly, limitations in the implementation of such automation procedures need to be investigated, and robustness of such automation schemes needs to be studied.

5 Conclusions

A *token-based* ATM paradigm was presented that enables the de-coupling of the two main ATM tasks: (1) maintaining safety — an Air Traffic Control System Task; and (2) optimal choice of flight paths — an Airline Operational Center (AOC) Goal. The token-based paradigm allows AOCs to freely trade the tokens between different aircraft and to choose desirable flight paths consisting of sequences of these regions (as long as the AOC can procure the needed tokens). Although, the aircraft is forced to fly along established routes in each sector, the freedom in choice of sectors may meet the AOC's needs for flexibility.

6 Acknowledgment

Work supported by NASA Grant NAG 2-1450. Discussions with, and suggestions from D. Iamratanukal and C. Lee are gratefully acknowledged.

References

[1] H. Erzberger and W. Nedell. Design of automated system for management of arrival traffic. *NASA Technical Memorandum 102201*, June, 1989.

[2] B. Shridar and G. B. Chatterji. Computationally efficient conflict detection methods for air traffic management. *ACC, Albuquerque, New Mexico*, 1997.

[3] R. W. Schwab, A. Haraldsdottir, and A. W. Warren. A requirements-based cns/atm architecture. *AIAA Paper 985552, World Avaiation Conference, Anaheim, CA*, September 28-20, 1998.

[4] J. Kosecka, C. Tomlin, G. Papas, and S. Sastry. 2-1/2 d conflict resolution maneuvers for atms. *CDC 1998.*

[5] Y-J Chiang, J. T. Klosowski, C. Lee, and J. S. B. Mitchell. Geometric algorithms for conflict detection/resolution in air traffic management. *36th IEEE CDC San Diego*, pages 1835–1840, 1997.

[6] Z-H Mao and E. Feron. Stability and performance of intersecting aircraft flows under sequential conflict resolution. *Proceedings of the ACC, Arlington, VA, June 25-27*, 2001.

[7] S. Devasia and G. Meyer. Automated conflict resolution procedures for air traffic management. *Proceedings of the CDC, Vol. 3, pp. 2456-2462*, December 1999.

[8] R. A. Brualdi and H. J. Ryser. Combinatorial matrix theory. *Cambridge University Press, Cambridge*, 1991.

[9] V. Lund. Automated conflict resolution in air traffic management. *M.S. Thesis, U. of Utah*, August, 2000.