CONFIGURING AIRSPACE SECTORS WITH APPROXIMATE DYNAMIC PROGRAMMING

Michael Bloem*, Pramod Gupta**
***NASA Ames Research Center, **University of California Santa Cruz at NASA Ames Research Center**

Keywords: *dynamic airspace configuration, approximate dynamic programming*

Abstract

In response to changing traffic and staffing conditions, supervisors dynamically configure airspace sectors by assigning them to control positions. A finite horizon airspace sector configuration problem models this supervisor decision. The problem is to select an airspace configuration at each time step while considering a workload cost, a reconfiguration cost, and a constraint on the number of control positions at each time step. Three algorithms for this problem are proposed and evaluated: a myopic heuristic, an exact dynamic programming algorithm, and a rollouts approximate dynamic programming algorithm. On problem instances from current operations with only dozens of possible configurations, an exact dynamic programming solution gives the optimal cost value. The rollouts algorithm achieves costs within 2% of optimal for these instances, on average. For larger problem instances that are representative of future operations and have thousands of possible configurations, excessive computation time prohibits the use of exact dynamic programming. On such problem instances, the rollouts algorithm reduces the cost achieved by the heuristic by more than 15% on average with an acceptable computation time.

Nomenclature

α	Parameter in workload cost $g_w(X)$
$\mathbf{A}$	Adjacency matrix for sectors
$\bar{w}(c,T)$	Workload threshold in control position c when the traffic situation is T
$\bar{\mathbf{w}}(X)$	A $M \times 1$ vector in which the i^{th} component is $\bar{w}(c_i,T)$, $c_i \in C$
β	Parameter in single time step cost $g(X_k,u_k,X_{k+1})$
$\hat{T}$	Uncertain prediction of future traffic
$\hat{w}(c,\hat{T})$	Expected workload level in control position c when the traffic prediction is $\hat{T}$
$\hat{\bar{w}}(c,\hat{T})$	Expected workload threshold in control position c when the traffic prediction is $\hat{T}$
$\mathbf{C}$	Set of all valid configurations
$\mathbf{C}_i$	Set of all valid configurations with i control positions
$\mu_{\text{heur}}(X,d)$	Heuristic control policy; outputs a control action
$\mu_k(X_k)$	Indicates what control action to take at time step k and state X_k
π	Control policy specifying μ_k for all time steps of interest
$\tilde{J}_k(X_k)$	Rollouts approximate optimal cost-to-go from state X_k at time step k
$\mathbf{w}(X)$	A $M \times 1$ vector in which the i^{th} component is $w(c_i,T)$, $c_i \in C$
C	A configuration; a set of control positions
c	A control position; a set of sectors
d_k	Required number of control po-

	sitions at time step k
$g(X_k, u_k, X_{k+1})$	Single time step cost
$g_r(X_k, u_k, X_{k+1})$	Reconfiguration cost
$g_w(X)$	Workload cost
H	Number of Monte Carlo simulations
$J_k^\star(X_k)$	Optimal cost-to-go from state X_k at time step k
K	Final time step
k	Time step
L	Reduced number of time steps; $L \le K$
M	The number of control positions
N	Number of sectors
S	A set of sectors to be configured
s	A sector
T	State and flight plan information for all aircraft that are currently in or are planning to fly through the sectors of interest
u_k	Control action time step k; equal to C_{k+1}
$u_k^\star$	Optimal action at time step k
$w(c,T)$	Workload in control position c when the traffic situation is T
X	System state; includes C and T

1 Introduction

In current air traffic management operations, airspace is partitioned into static volumes called sectors to facilitate the division of responsibilities between air traffic controller teams. A sector configuration maps a set of airspace sectors into a set of control positions such that each sector is assigned to exactly one control position. A single team of one to three air traffic controllers monitors each control position. Supervisors select configurations so as to 1) assign enough controller teams to busy airspace and 2) avoid workload associated with transitions between configurations [20]. Current air traffic control operations require so much memorization that each controller is only certified to control a set of six to eight sectors referred to as an “area of specialization," such as the area shown in figure 1. The relatively small number of sectors in each area mean that there are a relatively small number of permitted sector configurations in each area.

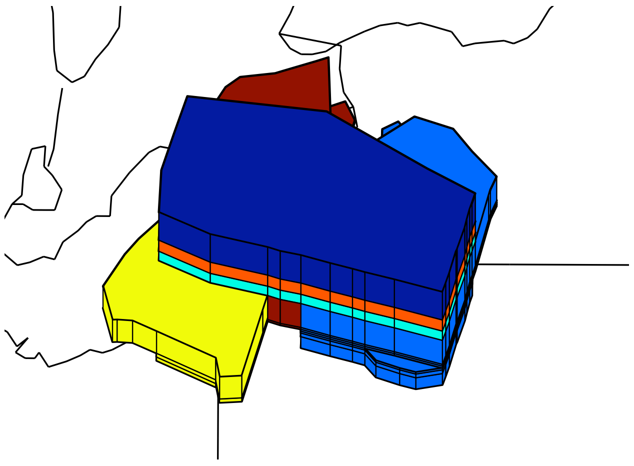

Fig. 1 The six sectors in Area 7 of Cleveland Air Route Traffic Control Center.

With planned improvements in automation and changes in operations proposed by MITRE [1, 28], Mogford [31–33], and the Federal Aviation Administration [11], controllers will be able to control a wider range of sectors. This will increase the number of possible configurations. This increase in configuration options makes it more likely that an appropriate sequence of configurations will exist to fit any given traffic and controller resource situation. However, it will also make the selection of appropriate sector configurations more challenging for supervisors. Therefore, a tactical decision-support tool that suggests good sector configurations will aid supervisors in this context [11].

Several algorithms for configuring sectors that could be used in such a decision-support tool have been proposed [4, 5, 7, 9, 10, 12–19, 36]. These algorithms vary in the version of the sector configuration problem they attempt to solve. One supports tactical decision-making [9], but the rest focus on pre-tactical or strategic decisions concerned with staff planning. Some attempt to minimize the number of control positions required to safely manage the traffic [4, 5, 10]. Others set a required number of control positions at each time [7] or put an upper bound on the number of control positions at each time [9, 16, 17, 36].

These algorithms also vary in how they quantify controller workload. Workload is quantified using aircraft count [4, 5, 10], inter-control position coordination [7], control position throughput [36], detailed human task load models [9], or neural networks trained on historical traffic data and sector configurations [13,15,18,19]. Furthermore, none of the existing algorithms explicitly consider the uncertainties involved in predictions of air traffic. Such uncertainties impact predictions of any measure of controller workload.

The workload associated with reconfigurations is treated differently by existing algorithms for configuring sectors. Several algorithms ignore this reconfiguration workload [4, 5, 9, 10], but some take steps to control the frequency of reconfigurations [12, 14, 15, 36]. None specify an explicit reconfiguration cost that considers the differing levels of workload induced by different reconfigurations, even though such differences exist and have been studied [20,21,26,27].

These algorithms also vary in the solution methods they utilize to select configurations. The methods include original heuristics [4,5], integer programming [10,36], genetic algorithms [7, 16, 17], brute-force exhaustive search [9], and tree-search methods like branch-and-bound and the $A^\star$ algorithm [12,14–17].

In this paper, a new formulation of the tactical sector configuration problem is proposed. This formulation explicitly considers uncertainties in predictions of air traffic by using predictions of future air traffic sampled from an appropriate distribution. The control actions in this problem are the sector configurations at each time step over a finite time horizon. These configurations must contain a required number of control positions at each time step. The objective is to minimize the expected value of a weighted sum of a workload cost and a reconfiguration cost. Three algorithms for this problem are proposed and evaluated. One of these algorithms is a myopic heuristic, another is an exact dynamic programming algorithm that can be solved with value iteration, and the third uses rollouts to generate an approximate dynamic programming solution.

The rest of this paper is structured as follows. The new sector configuration problem is specified in Section 2. The algorithms for this problem are described in Section 3. In Section 4, the performance of the algorithms is analyzed by applying them to small problem instances from current operations and large problem instances from possible future operations. Section 5 contains conclusions and Section 6 contains suggestions for future work.

2 Problem Specification

The problem is to select an airspace configuration at each time step while considering a workload cost, a reconfiguration cost, and a constraint on the number of control positions at each time step. The problem specification consists of a system model and dynamics, problem constraints, and an objective function.

Let $S = \{s_1, s_2, \ldots, s_N\}$ be the set of N sectors to be configured. The matrix $\mathbf{A}$ is an adjacency matrix that specifies which sectors are neighbors. Element $\mathbf{A}_{ij}$ is 1 if s_i is adjacent to s_j, otherwise it is 0. A configuration of S is specified by $C = \{c_1, c_2, \ldots, c_M\}$, where M is the number of control positions. Each control position $c_i \in C$ is itself a set that contains one or more sectors.

2.1 System Model

The system state at time step k is $X_k = (C_k, T_k)$, where C_k is the sector configuration at k and T_k specifies the aircraft traffic situation for the sectors in S at k. T_k contains state and flight plan information for all aircraft that are currently in or are planning to fly through the sectors in S. It contains all the aircraft data available for predicting future traffic situations.

The size of the state space for C_k can be in the thousands or millions for problems of interest in the future air traffic management system. For one such problem studied in Section 4, there are 8,362 valid configurations of 11 sectors. For one current-day problem in the French airspace, there are around 124,000 valid configurations [17]. The size of the state space for T_k is infinite because aircraft state variables can be continuous.

The control at a time step sets the configuration for the next time step: $u_k = C_{k+1}$. Aircraft trajectory predictors can be used to predict the traffic situation at future time steps, but there is no simple function or stationary statistical distribution that specifies the dynamics of T_k. However, there are some distributions available that describe the dynamics of the number of aircraft in a sector [37]. Rather than explicitly modeling the dynamics of traffic, these distributions are used to model the dynamics of the number of aircraft in each sector, one measure of workload, as will be discussed in the next two sections.

2.1.1 Workload

There are many functions that attempt to quantify the workload in a control position, given the control position and the current traffic [6, 22–25, 29, 35]. Let the selected workload function be specified by $w(c,T)$. Let $\mathbf{w}(X)$ return a $|C| \times 1$ vector in which the i^{th} component is the workload for $c_i \in C$ and where $X = (C,T)$. Uncertain predictions of future traffic $\hat{T}$ can also be passed into the workload function, yielding corresponding predictions of the control position workloads.

There is a workload threshold associated with each control position. This threshold is not a hard constraint, but as workload approaches and exceeds the threshold, operations in the control position probably become less safe and less efficient. This threshold may depend on characteristics of the traffic in the sector, such as sector transit times [38]. Let $\bar{w}(c,T)$ specify the workload threshold for c when the traffic situation is T, and let $\bar{\mathbf{w}}(X)$ be a $|C| \times 1$ vector in which the i^{th} component is the workload threshold for $c_i \in C$. Generally, the workload threshold for a control position is much smaller than the sum of the thresholds of the individual sectors that make up the control position.

2.1.2 Workload Dynamics

Wanke et al. used historical prediction errors to estimate distributions that describe the dynamics of a measure of workload [37]. This workload measure, which is used in the current air traffic management system, is the peak aircraft count that occurs in a sector during a 15-minute time period. Wanke et al. specify a distribution for future peak aircraft count based on prediction lookahead time, sector type, and properties of a particular prediction of the future traffic in the sector (peak predicted number of aircraft in the sector during the time period and the fraction of those aircraft that have departed at the time of the prediction). Unfortunately, there are no published joint or conditional distributions capturing the relationship between peak aircraft count levels over time and in different control positions.

For this research, the 15-minute lookahead distributions of peak aircraft count derived by Wanke et al. are used to model the dynamics of the workload in each sector [37]. Algorithms also use these distributions when making predictions about the future, even if the predictions are made with a lookahead time that is larger than 15 minutes. It is assumed that the distributions are independent across sectors and time steps because no joint or conditional distributions are available. The traffic-related distribution parameters (predicted number of aircraft in the sector and the fraction of those aircraft that have departed at the time of the prediction) are computed from historical traffic rather than predicted traffic because the operational prediction data is not available. A sample distribution is shown in figure 2.

This approach allows for algorithm performance evaluations based not only on the single sample path of historical traffic but also on other sample paths that could have occurred. This sort of evaluation is essential when evaluating any algorithm that claims to be robust to uncertainties. Predictions that are consistent with the workload dynamics can be generated by sampling the independent distributions that govern the system dynamics, enabling algorithms to accurately and quickly approximate expected values of future costs. This approach assumes that samples in nearby sectors and at nearby times are independent. A second assumption is that uncertainties in predictions are constant with respect to looka-

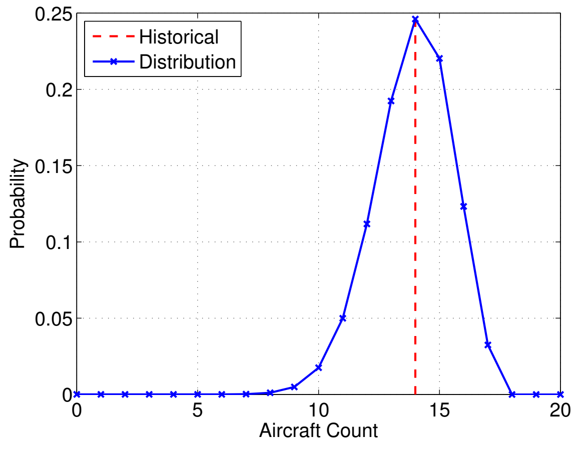

Fig. 2 Probability mass function for the 12:45 pm (15-minute lookahead) prediction of the maximum number of aircraft in sector ZOB38 in Cleveland Center between 1:00 pm and 1:15 pm local time on 22 February 2007.

head time. The impact of these assumptions and alternative approaches to handling system uncertainties should be evaluated in future research.

2.2 Objective

There are two components of the cost at each stage: a workload cost g_w and a reconfiguration cost g_r. The total cost g is a weighted sum of the two.

The workload cost penalizes cases where the $\mathbf{w}(X_k)$ exceeds $\bar{\mathbf{w}}(X_k)$. This cost function starts with the amount by which the workload exceeds the workload threshold in each control position, multiplies it by the parameter α, squares the result, and finally sums the results for all the control positions in C_k:

$$g_w(X_k) = \|\alpha[\mathbf{w}(X_k) - \bar{\mathbf{w}}(X_k)]_+\|_2^2, \qquad (1)$$

where the operators are defined as follows. For a real input a, $[a]_+$ is 0 if $a < 0$ and a if $a \geq 0$. When operating on a vector, as in (1), the operator $[\cdot]_+$ applies separately to each element in the vector. Also, for a vector a, $\|a\|_2^2$ is the squared 2-norm of a. Control positions with workload levels below the threshold are not penalized. Alternative workload cost function formulations are possible. Airspace sector design and configuration research has pursued objectives like workload balance, traffic flow conformance, and low inter-control position coordination [8,30,34,39]; these could be added to this cost function in future work.

The next component is the reconfiguration cost, which increases as the next configuration becomes more different from the current configuration. Significant configuration changes induce workload for air traffic control staff as they transition from one control position to another [20,21,26,27]. This cost specifies the number of control positions in C_{k+1} that were not present in C_k:

$$g_r(X_k, u_k, X_{k+1}) = |C_{k+1} \setminus C_k|, \qquad (2)$$

where $\setminus$ is the set minus operator. Alternate reconfiguration cost formulations may be considered. Gianazza et al. use $\min\{|C_k|, |C_{k+1}|\} - |C_k \cap C_{k+1}|$ as a distance function in a genetic algorithm [17]. The authors of this paper have previously used $|C_k \setminus C_{k+1}|$ [20]. These possibilities are independent of the traffic situations T_k and T_{k+1}, but the reconfiguration workload does depend on the traffic situation, and more sophisticated cost functions may consider the traffic in control positions that are being combined and split to move from one configuration to the next [21,26,27].

The overall single time step cost is simply the weighted sum of equations (1) and (2):

$$g(X_k, u_k, X_{k+1}) = g_w(X_k) + \beta g_r(X_k, u_k, X_{k+1}). \qquad (3)$$

It is parameterized by α and β, a nonnegative weight on the reconfiguration cost. Appropriate values for α and β are a topic for future research.

The overall problem considered here is a finite horizon expected total cost problem with the objective

$$\min_{\pi=(\mu_0,\mu_1,\ldots,\mu_{K-1})} \mathbf{E}\left[\sum_{k=0}^{K-1} g(X_k, u_k, X_{k+1}) \middle| X_0\right], \qquad (4)$$

subject to the system model described in subsection 2.1 and the constraints discussed in subsection 2.3. For some random variable a and a variable it depends on b, $\mathbf{E}[a|b]$ denotes the expected value of a given b. Also, π is a control policy in which each $\mu_k(X_k)$ indicates what control action to take at time step k and state X_k.

2.3 Constraints

A valid C must assign each $s_i \in S$ to a control position:

$$\bigcup_{i=1}^{M} c_i = S, \tag{5}$$

where $\cup$ is the set union operator. A valid C also cannot assign any s_i to more than one control position:

$$c_i \cap c_j = \emptyset \quad \forall\, c_i, c_j \in C,\ i \neq j, \tag{6}$$

where $\cap$ is the set intersection operator and $\emptyset$ is the empty set. These two requirements ensure that $1 \leq M \leq N$. All the s_i in a valid c must define a single connected component in the graph specified by $\mathbf{A}$. This ensures that sectors in the same control position are adjacent to each other.

Furthermore, certain control positions may not be allowed for reasons such as a preference for convexity in the top-view of control positions or a maximum control position size constraint. Any valid configuration C contains no disallowed control position.

The set $\mathbf{C}$ contains all valid configurations of a given set of sectors S. Furthermore, let $\mathbf{C}_i$ be the set of all valid configurations with i control positions (i.e. $|C| = i \ \forall\ C \in \mathbf{C}_i$). It is easy to see that $|\mathbf{C}_1| = 1$ and $|\mathbf{C}_N| = 1$, but there may be many valid configurations when the number of control positions is between 1 and N. There is a constraint on the control that specifies the number of control positions that can be used at each time step. This constraint is needed because a certain number of controller teams are available and should be working at each time step. Let d_k specify the required number of control positions at k. Therefore, $u_k \in \mathbf{C}_{d_{k+1}}$ is a constraint.

3 Algorithms

This section proposes three algorithms that attempt to solve the problem defined in Section 2. The first is a myopic heuristic that is similar to the heuristic in Refs. [4,5]. It computes a solution quickly but makes no guarantees on its optimality. The next algorithm uses dynamic programming and solves for the optimal costs-to-go with value iteration. It computes the optimal solution but is computationally intensive and can be slow when $|\mathbf{C}|$ is large. The final algorithm utilizes an approximate dynamic programming technique known as rollouts. It uses the myopic heuristic to approximate the optimal costs-to-go and is therefore a hybrid of the first two algorithms. It can guarantee better performance than the myopic heuristic and produces a solution faster than exact dynamic programming.

3.1 Myopic Heuristic

A myopic heuristic algorithm was designed for this problem. This algorithm is myopic because when selecting u_k it only considers the expected workload levels in the next time step and the required number of control positions in the next time step. The expected workload levels $\hat{w}(c, \hat{T}_{k+1})$ and workload thresholds $\hat{\bar{w}}(c, \hat{T}_{k+1})$ are approximated with the mean workload from Monte Carlo simulations of the next time step.

If $|\mathbf{C}_{d_{k+1}}| > |\mathbf{C}_{d_k}|$ or $\hat{w}(c, \hat{T}_{k+1}) > \hat{\bar{w}}(c, \hat{T}_{k+1})$ for some $c \in C_k$, then $c_{\max} = \operatorname{argmax}_{c \in C_k} \hat{w}(c, \hat{T}_{k+1})$ is split into two control positions. If there are multiple ways to split $c_{\max}$, the split that leaves the lowest maximum $\hat{w}(c, \hat{T}_{k+1})$ in the two resulting control positions is selected.

If $|\mathbf{C}_{d_{k+1}}| < |\mathbf{C}_{d_k}|$, then the algorithm selects a combination to implement. It searches for an acceptable combination by starting with $c_{\min} = \operatorname{argmin}_{c \in C_k} \hat{w}(c, \hat{T}_{k+1})$. All possible ways to combine $c_{\min}$ with a neighboring c are investigated, and the combination that results in the control position with the lowest is implemented, assuming that expected workload is below a threshold. If no acceptable combination involving $c_{\min}$ is found, the control position with the next lowest expected workload is considered until eventually an acceptable combination is determined. Some additional logic helps ensure that some combination is found so that the constraint $u_k \in \mathbf{C}_{d_{k+1}}$ is not violated.

3.2 Dynamic Programming Solution via Value Iteration

A fundamental quantity in dynamic programming is the optimal cost-to-go; it is the cost that would be incurred if optimal actions are taken from some state onward. The optimal cost-to-go function here depends on the time step k and is specified as

$$J_k^\star(X_k) = \min_{\mu_k,\dots,\mu_{K-1}} \mathbf{E}\left[\sum_{j=k}^{K-1} g(X_j,u_j,X_{j+1}) \middle| X_k\right]. \tag{7}$$

This optimal cost-to-go is approximated with the average cost from Monte Carlo simulations constructed from the workload distributions discussed in sub-section 2.1.2. The expression for this approximation is

$$J_k^\star(X_k) \approx \min_{\mu_k,\dots,\mu_{K-1}} \frac{1}{H}\sum_{h=1}^{H}\left[\sum_{j=k}^{K-1} g(X_j^h,u_j,X_{j+1}^h)\right], \tag{8}$$

where X_j^h is the state at time step j in Monte Carlo simulation h, $h = 1,2,\dots,H$. In this finite time horizon case, the $J_k^\star(X_k)$ values can be computed with value iteration starting at the final time step [2].

The optimal action to take at each time step can then be found with

$$u_k^\star \in \operatorname*{argmin}_{u_k\in\mathbf{C}_{d_{k+1}}} \mathbf{E}\left[g(X_k,u_k,X_{k+1}) + J_{k+1}^\star(X_{k+1}) \middle| X_k\right]. \tag{9}$$

Again, the expected value here is approximated with Monte Carlo simulations.

3.3 Rollouts Approximate Dynamic Programming Solution

Rollouts is a technique for approximating the optimal cost-to-go [2]. The approximate optimal cost-to-go is then used in equation (9) to select an action at each time step. The idea is to approximate the optimal cost-to-go at each possible next state with the cost-to-go that would be incurred if a heuristic were used for the remainder of the time horizon. This approach is closely related to policy iteration, a technique that can be used to solve for the optimal costs-to-go [2]. Rollouts is guaranteed to lead to better cost results than are achieved by the raw heuristic that it uses. The approximations of the optimal cost-to-go are computed as needed during algorithm execution, which is appropriate for this problem because the state space is infinite. These approximations can make use of the latest traffic situation data, so this approach will be even more appropriate for more sophisticated versions of the sector configuration problem in which new traffic information enables more accurate predictions of future traffic and therefore future costs.

More precisely, let $\mu_{\text{heur}}(X_k,d_{k+1})$ be a heuristic control policy such as the myopic heuristic suggested in sub-section 3.1. The optimal cost-to-go is approximated by assuming that this heuristic policy is used for the remainder of the time horizon:

$$J_k^\star(X_k) \approx \mathbf{E}\left[\sum_{j=k}^{K-1} g(X_j,\mu_{\text{heur}}(X_j,d_{j+1}),X_{j+1}) \middle| X_k\right]. \tag{10}$$

As with the dynamic programming solution via value iteration, the expected value is approximated with the average of H Monte Carlo simulations. The rollouts algorithm computes its approximate optimal cost-to-go according to

$$\tilde{J}_k(X_k) = \frac{1}{H}\sum_{h=1}^{H}\left[\sum_{j=k}^{K-1} g(X_j^h,\mu_{\text{heur}}(X_j^h,d_{j+1}),X_{j+1}^h)\right]. \tag{11}$$

Three techniques are used to speed up the computation of the approximate optimal cost-to-go in equation (11). The first is to compute the cost-to-go incurred by the heuristic for some L time steps, rather than all the way until the end of the problem time horizon. This will hinder the performance of the algorithm because there may be a reason to select a control action that is not revealed until beyond the next L time steps. The second technique is to stop computing the approximate optimal cost-to-go for a configuration when the approximation already exceeds the smallest approximate optimal cost-to-go found so far. This technique will not adversely affect the algorithm's cost performance. The third method for speeding up computation is to reduce the number of Monte Carlo simulations used to approximate the expected value. Such Monte Carlo simulations are computationally cheap to produce using the workload distributions assumed for each sector at each time step (see sub-section 2.1.2), so this method is probably not helpful for this version of the sector configuration problem. However, some Monte Carlo simulations and workload metrics are computationally expensive to produce, so this method could become important in future work.

4 Quantitative Analysis of the Algorithms

4.1 Small Problem Instances from Current Operations

Areas of specialization typically contain six to eight sectors, and they make convenient small problem instances for studying the performance of the three proposed algorithms. They are also interesting problem instances to consider because historical sector configuration data is available, so the performance of the algorithm-generated configurations can be compared with the performance of the historical configurations. Finally, differences between the historical configurations and those selected by the algorithms can help reveal problems with the algorithms [20]. A detailed comparison of this type is an opportunity for future research.

For these experiments, historical air traffic data from three Thursdays in February of 2007 and the eight areas of specialization in Cleveland Air Route Traffic Control Center were used to construct 19 different problem instances. Figure 1 shows the sectors in one such area. Each of the eight areas considered for these problems contains six or seven sectors. The valid configurations for each day were limited to those that use control positions from the historical sector configurations on that day. Also, the number of required control positions at each time step was set to the number in the historical data. There were between 12 and 24 possible configurations for each problem instance. The number of configurations with the required number of control positions at each time step ranged between one and nine but averaged only about three.

Each problem instance ran from 3:45 am to 10:45 pm local time. The time steps were 15 minutes long, but the algorithms did not usually change airspace configurations every 15 minutes due to the reconfiguration cost. For each of the 19 instances, 10,000 Monte Carlo runs were created to help determine how each algorithm would do under a variety of possible workload sample paths based on the recorded traffic sample path. These runs were generated using the independent distributions discussed in sub-section 2.1.2. The dynamic programming algorithm used a different set of 10,000 Monte Carlo simulations to compute expected values, while the heuristic and rollouts algorithms used only 100 such simulations in an attempt to increase their computational speed. The rollouts algorithm approximated the optimal cost-to-go by running the heuristic for four hours rather than for the entire remainder of the problem. Finally, workload was specified as the maximum number of aircraft in a control position during the time step divided by a control position-specific upper bound called the Monitor Alert Parameter (MAP) so that all workloads ranged between zero and about one. For some control positions, published MAP values are available in the Future ATM Concepts Evaluation Tool [3]. If no published value was available, the MAP value was set to the largest of the MAP values of the sectors in the control position. The workload threshold returned by $\bar{w}(c,T)$ was set to 0.9 for all control positions and traffic situations. In the single stage cost function, the α parameter was set to 10 and the β parameter was set to 1. This means that the workload cost of the workload equaling the threshold in a control position for one time step is equal to the reconfiguration cost of one new control position. Table 1 contains a summary of the parameters used for these analyses.

Table 1 Problem and Algorithm Parameters for Small Operational Problem Instances

Parameter	Value
Problem start time	3:45 am Eastern Time
Problem end time	10:45 pm Eastern Time
Time step duration	15 minutes
K	75
L	16
H for dynamic programming	10,000
H for heuristic and rollouts	100
$w(c,T)$	$\frac{\text{maximum aircraft count during time step}}{\text{monitor alert parameter}}$
$\bar{w}(c,T)$	0.9
α	10
β	1

Table 2 Cost and Computation Time Results for Small Operational Problem Instances

Approach	Fraction of Optimal Cost (g)			Mean Compute Time
	Min	Mean	Max	per Time Step [seconds]
Dynamic Programming	1.000	1.000	1.000	0.095
Historical	1.000	1.217	2.174	–
Heuristic	1.000	1.165	2.100	0.012
Rollouts	1.000	1.014	1.159	0.065

Table 2 summarizes the cost and computation time results for the Monte Carlo runs of these 19 problem instances. The dynamic programming approach computes the optimal solution to the problems, assuming that it uses enough Monte Carlo simulations to accurately approximate expected values. The costs incurred by the other approaches are expressed as a fraction of the optimal cost found by dynamic programming. Figure 3 graphically portrays the information in Table 2. Note that the compute time per time step is set to zero for the operational configurations because the actual compute time in operations is not known. The historical sector configurations perform relatively well on these small problems. On at least one problem instance they achieve the optimal cost, and on average the cost from the historical configurations is within about 22% of optimal. However, on one problem instance the historical sector configurations led to a cost that was more than double the optimal cost. The performance of the heuristic is similar to that of the historical configurations, but slightly better. The rollouts algorithm achieves a cost that is within 2% of optimal on average and never more than 16% from optimal on these 19 problem instances.

The dynamic programming algorithm took about a tenth of a second per time step to select a configuration. It benefitted from the fact that all of the workload distributions were assumed to be fixed and independent. This meant that the dynamic programming algorithm only had to perform value iteration at the first time step, and then could re-use the optimal costs-to-go at all future time steps. In a more realistic problem formulation where new information becomes available over time and changes the distribution of future traffic and workload, the dynamic programming solution would have to perform value iteration at each time step, and it would therefore be slower. The heuristic algorithm computed a solution in about a hundredth of a second. The rollouts algo-

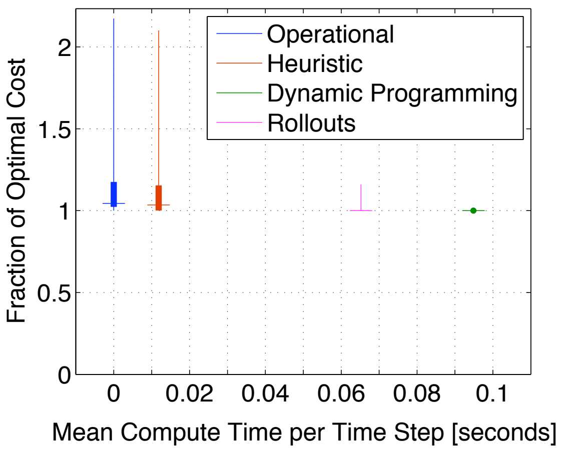

Fig. 3 Mean compute time per time step and spread of cost results for each algorithm. The bottom of the vertical line is the lowest cost result, the bottom of the box is the 25th percentile, the horizontal line is the median, the top of the box is the 75th percentile, and the top of the vertical line is the maximum.

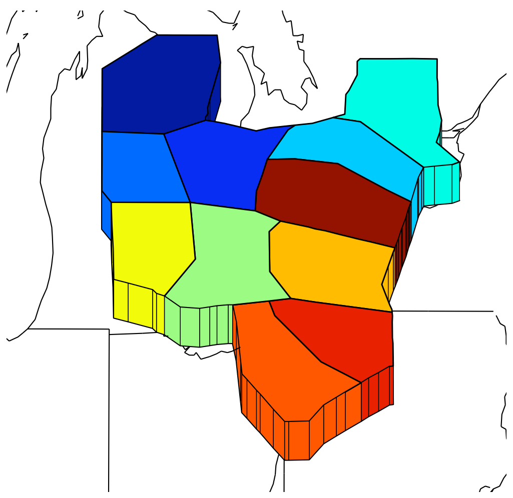

Fig. 4 The 11 sectors in a possible future area of Cleveland Air Route Traffic Control Center.

rithm took about two-thirds of the time required by the dynamic programming solution per time step to select a configuration. The efforts to speed up the rollouts algorithm discussed in sub-section 3.3 approximately halved its computation time.

4.2 Large Problem Instances from Possible Future Operations

A hypothetical future area of specialization consistent with the research of Mogford [31–33] was constructed by slightly modifying the super-high altitude sectors in Cleveland Air Route Traffic Control Center. It contains 11 sectors, all with altitude floors at 34,000 feet, and is shown in figure 4.

Three problem instances from the same three Thursdays in February of 2007 were created for this possible area. For these instances, there are 8,362 possible valid configurations. At each time step, the number of valid control actions ranged between 1 and 2,288 and averaged almost 500. The required number of control positions for each problem instance was set so that the total number of aircraft in the area divided by the number of control positions would not be larger than 10 aircraft. The other details relating to these instances are the same as for the smaller problem instances as described in sub-section 4.1, except that these simulations ran for one more time step.

An exact dynamic programming solution was attempted for one of these problem instances but after 10 hours it had still not finished computing the optimal costs-to-go. Since it takes prohibitively long to compute the optimal solution with exact dynamic programming problem instances of this size, the cost results in table 3 are expressed in terms of a percentage improvement over the cost achieved by the heuristic. The rollouts approximate dynamic programming solution achieves a cost reduction of between 10% and 28% for these three problem instances, with an average cost reduction of more than 15%.

Information and automation tools are being developed to enable larger areas of specialization [1, 11, 28, 31–33]. If the heuristic performs better than configurations selected by a human supervisor in larger areas of specialization, as it does in current areas of specialization, then the rollouts solution would offer a cost improvement of 15% or more over configurations selected by a human supervisor in larger areas of specialization. This should be investigated in future research, as should the benefits over current operations of larger areas of specialization configured with the algorithms proposed here.

Table 3 Cost and Computation Time Results for Large Possible Future Problem Instances

Approach	**Percent Cost (g) Improvement over Heuristic**			**Mean Compute Time per Time Step [seconds]**
	Min	**Mean**	**Max**	
Heuristic	–	–	–	0.021
Rollouts	10.83%	17.87%	27.50%	97.0

The heuristic algorithm computes a solution for each time step in about two hundredths of a second on average, which is about twice as long as it took for the smaller problems in subsection 4.1. The rollouts algorithm takes on average 100 seconds to compute each next configuration. This should be acceptable because this computation would only need to complete within the 15-minute time step.

5 Conclusions

A finite horizon airspace sector configuration problem was formulated. It considers workload costs, reconfiguration costs, and a constraint on the number of control positions at each time step. Three algorithms for this problem were proposed and evaluated. On small problem instances from current operations, an exact dynamic programming solution can be computed with value iteration. This solution gives the optimal cost value for the problem instances. Historical sector configurations and those generated by a myopic heuristic on average led to costs that were about 22% and 17% larger than the optimal cost, respectively. A rollouts approximate dynamic programming algorithm based on the myopic heuristic achieved costs within 2% of optimal, on average. For large problem instances based on possible future operations, the rollouts algorithm reduced the cost achieved by the heuristic by more than 15% on average with an acceptable computation time.

6 Future Work

There are several opportunities for extensions of this work. A more sophisticated measure of workload than aircraft count divided by control position monitor alert parameter should be used. A more detailed comparison of historical and algorithmically-generated configurations may reveal ways to improve the algorithms and their cost function, particularly with the help of airspace operations experts. The reconfiguration cost in particular may need further research and refinement because research on reconfiguration costs is still in its early stages [21, 26, 27]. Furthermore, traffic and workload uncertainties should be modeled more realistically. Finally, the algorithms proposed here should be used to approximate the benefits of increasing the size of areas of specialization and thereby allowing more possible configurations.

Acknowledgements

The authors thank Mark Evans for answering questions about the airspace sector configuration problem.

References

[1] High altitude trajectory-based (generic) airspace update concept exploration and refinement. MITRE Center for Advanced Aviation System Development presentation, March 2010.

[2] Bertsekas D. P. *Dynamic Programming and Optimal Control*. Vol. I, Athena Scientific, Nashua, NH, 2005.

[3] Bilimoria K, Sridhar B, Chatterji G, Sheth K, and Grabbe S. FACET: Future ATM Concepts Evaluation Tool. *Air Traffic Control Quarterly*, Vol. 9, No 1, pp 1–20, 2001.

[4] Bloem M, Gupta P, and Kopardekar P. Algorithms for combining airspace sectors. *Air Traf-*

fic Control Quarterly, Vol. 17, No 3, October 2009.

[5] Bloem M and Kopardekar P. Combining airspace sectors for the efficient use of air traffic control resources. *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit*, Honolulu, HI, August 2008.

[6] Chatterji G. B and Sridhar B. Measures for air traffic controller workload prediction. *Proc. of AIAA Aircraft, Technology, Integration, and Operations Forum*, Los Angeles, CA, October 2001.

[7] Delahaye D, Alliot J.-M, Schoenauer M, and Farges J.-L. Genetic algorithms for automatic regroupment of air traffic control sectors. *Proc. of 4th Conference on Evolutionary Programming*, San Diego, CA, March 1995.

[8] Delahaye D, Durand N, Alliot J.-M, and Schoenauer M. Genetic algorithms for air traffic control system. *Proc. of INFAUTOM*, 1996.

[9] Dorado M. ETLM support tool for dynamic airspace configuration. *NASA Dynamic Airspace Configuration Workshop*, Moffett Field, CA 2009.

[10] Drew M. C. A method of optimally combining sectors. *Proc. of AIAA Aviation Technology, Integration and Operations Conference*, Hilton Head, SC, September 2009.

[11] FAA ATO Planning, Research, and Technology. An operational concept for mid-term high altitude trajectory-based airspace. Technical report, Federal Aviation Administration, Atlantic City, NJ, August 2009.

[12] Gianazza D, Allignol C, and Saporito N. An efficient airspace configuration forecast. *Proc. of USA/Europe Air Traffic Management Research & Development Seminar*, Napa, CA, July 2009.

[13] Gianazza D. Airspace configuration using air traffic complexity metrics. *Proc. of USA/Europe Air Traffic Management Research & Development Seminar*, Barcelona, Spain, July 2007.

[14] Gianazza D. Smoothed traffic complexity metrics for airspace configuration schedules. *Proc. of 3rd International Conference on Research in Air Transportation*, Fairfax, VA, June 2008.

[15] Gianazza D. Forecasting workload and airspace configurations with neural networks and tree search methods. *Artificial Intelligence*, 2010.

[16] Gianazza D and Alliot J.-M. Optimization of air traffic control sector configurations using tree search methods and genetic algorithms. *Proc. of AIAA/IEEE Digital Avionics Systems Conference*, October 2002.

[17] Gianazza D, Alliot J.-M, and Granger G. Optimal combinations of air traffic control sectors using classical and stochastic methods. *Proc. of International Conference on Artificial Intelligence*, Las Vegas, NV, June 2002.

[18] Gianazza D and Guittet K. Evaluation of air traffic complexity metrics using neural networks and sector status. *Proc. of 2nd International Conference on Research in Air Transportation*, Belgrade, Serbia and Montenegro, June 2006.

[19] Gianazza D and Guittet K. Selection and evaluation of air traffic complexity metrics. *Proc. of AIAA/IEEE Digital Avionics Systems Conference*, October 2006.

[20] Gupta P, Bloem M, and Kopardekar P. An investigation of the operational acceptability of algorithm-generated sector combinations. *Proc. of AIAA Aviation Technology, Integration and Operations Conference*, Hilton Head, SC, September 2009.

[21] Jung J, Lee P, Kessell A, Homola J, and Zelinkski S. Effect of dynamic sector boundary changes on air traffic controllers. *Proc. of AIAA Guidance, Navigation, and Control Conference*, Toronto, Canada, August 2010.

[22] Klein A, Rogers M. D, and Leiden K. Simplified dynamic density: a metric for dynamic airspace configuration and NEXTGEN analysis. *Proc. of AIAA/IEEE Digital Avionics Systems Conference*, St. Paul, MN, October 2008.

[23] Kopardekar P and Magyarits S. Dynamic density: Measuring and predicting sector complexity. *Proc. of AIAA/IEEE Digital Avionics Systems Conference*, Irvine, CA, October 2002.

[24] Kopardekar P and Magyarits S. Measurement and prediction of dynamic density. *Proc. of USA/Europe Air Traffic Management Research & Development Seminar*, Budapest, Hungary, 2003.

[25] Laudeman I. V, Shelden S. G, Branstrom R, and Brasil C. L. Dynamic density: An air traffic management metric. Technical Report NASA/TM–1998-112226, NASA Ames Re-

search Center, 1998.

[26] Lee P. U, Prevot T, Homola J, Lee H, Kessell A, and Smith N. Sector design and boundary change considerations for flexible airspace. *Proc. of AIAA Aviation Technology, Integration, and Operations Conference*, Fort Worth, TX, September 2010.

[27] Lee P. U, Prevot T, Homola J, Lee H, Kessell A, Brasil C, and Smith N. Impact of airspace reconfiguration on controller workload and task performance. *Proc. of Applied Human Factors and Ergonomics*, Miami, FL, July 2010.

[28] Levin K. M. Universal High Altitude Airspace (UHAA). MITRE Center for Advanced Aviation System Development presentation, February 2007.

[29] Massalonis A, Callaham M. B, and Wanke C. R. Dynamic density and complexity metrics for realtime traffic flow management. *Proc. of 5th USA/Europe Air Traffic Management Research & Development Seminar*, Budapest, Hungary, December 2003.

[30] Mitchell J. S. B, Sabhnani G, Krozel J, Hoffman B, and Yousefi A. Dynamic airspace configuration management based on computational geometry techniques. *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit*, Honolulu, HI, August 2008.

[31] Mogford R. Generic airspace research phase 3 test plan. NASA Ames Research Center test plan, November 2009.

[32] Mogford R, Evans M, Gibson J, Miller J, Peknik D, Pfeiffer J, Preston W, Shih J, and West F. Generic airspace concepts and research. *Proc. of AIAA/IEEE Digital Avionics Systems Conference*, Salt Lake City, UT, October 2010.

[33] Mogford R. H. Generic operations research phase 1 report. Technical report, NASA Ames Research Center, February 2009.

[34] Sabhnani G. *Geometric Algorithms for Dynamic Airspace Sectorization*. PhD thesis, Stony Brook University, May 2009.

[35] Sridhar B, Sheth K. S, and Grabbe S. Airspace complexity and its application in air traffic management. *Proc. of 2nd USA/Europe Air Traffic Management Research & Development Seminar*, Orlando, FL, December 1998.

[36] Verlhac C and Manchon S. Optimization of opening schemes. *Proc. of 4th USA/Europe Air Traffic Management Research & Development Seminar*, Santa Fe, NM, December 2001.

[37] Wanke C, Song L, Zobell S, Greenbaum D, and Mulgund S. Probabilistic congestion management. *Proc. of 6th USA/Europe Air Traffic Management Research & Development Seminar*, Baltimore, MD, June 2005.

[38] Welch J. D, Martin B. D, and Sridhar B. Macroscopic workload model for estimating en-route sector capacity. *Proc. of 7th USA/Europe ATM Research and Development Seminar*, Barcelona, Spain, July 2007.

[39] Xue M. Airspace sector redesign based on Voronoi diagrams. *AIAA Journal of Aerospace Computing*, September 2008.

Copyright Statement