Fourteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)

Reinforcement Learning for Traffic Flow Management Decision Support

Christine Taylor, Erik Vargo, Emily Bromberg, Everett Carson
Center for Advanced Aviation System Development
The MITRE Corporation
McLean, VA, USA
ctaylor@mitre.org, evargo@mitre.org, ebromberg@mitre.org, ecarson@mitre.org

***Abstract*—Recent advances in Artificial Intelligence and Machine Learning are being harnessed to solve increasingly complex problems across a variety of domains, including Air Traffic Management. Application of these methods to the domain of Traffic Flow Management, however, remains a challenge as it is first necessary to effectively represent the dynamics of weather forecasts – and the uncertainty in the resulting constraints – within the construct of the decision-making process. In this paper, we propose a novel approach for capturing weather forecast uncertainty in a reinforcement learning process that generates Traffic Flow Management strategies in a real-time environment. Specifically, we leverage Monte Carlo Tree Search to explore and evaluate potential traffic management actions against an ensemble of weather futures. The results demonstrate that under the assumptions of the operational environment developed and the objective defined, the algorithm can generate effective solutions for managing uncertain constraints, adapt to changing information, and do so in a real-time context.**

Keywords- artificial intelligence; reinforcement learning; traffic flow management; sequential decision making; decision support

I. INTRODUCTION

Within the National Airspace System (NAS) and across Air Navigation Service Providers (ANSPs) globally, there is a recognized need to leverage new automation technologies to create a more robust and efficient Air Traffic Management (ATM) system capable of providing services to both current and new airspace users. A key component of the Federal Aviation Administration's (FAA's) future vision for operations is the application of Artificial Intelligence (AI) based methods to improve prediction accuracy and inform decisions in real-time [1].

AI's recent return to prominence has been driven by the successes of Google's DeepMind in winning against world class champions in Go [2] and StarCraft [3]. As such, researchers in a variety of domains including ATM have sought to employ AI techniques to address long-standing challenges. Much work has focused on the use of Machine Learning (ML) techniques for prediction and analysis such as for predicting taxi-out times [4], analyzing air traffic flows [5] [6], predicting runway occupancy times [7], predicting aircraft delays [8], designing operationally acceptable reroutes [9], and predicting Ground Delay Programs (GDPs) [10].

However, research investigating the use of AI for real-time control of aircraft (nominally, in a human machine team context), is more limited. Reference [11] develops an AI *agent* that can sequence taxiing aircraft at a hypothetical airport, using a Deep-Q-Network – a prominent type of Deep Reinforcement Learning (DRL) algorithm. Reference [12] trains a DRL algorithm to maintain aircraft separation in congested airspace, albeit assuming no uncertainty in either the known information or the result of the actions taken. The approach developed in [13] incorporates uncertainty into the conflict avoidance and separation assurance problem by assuming a normal distribution on the resulting state of an aircraft after executing the agent-prescribed action. Reference [14] builds upon this work to learn controller-specific resolutions with the goal of gaining greater operational acceptance.

Developing an AI agent to create Traffic Flow Management (TFM) strategies represents a new frontier in this evolution and is the focus of this paper. The goal of TFM is to balance demand with available capacity. Designing TFM strategies provides an ideal use case in that 1) today's operation remains experience-driven with limited decision support tools, 2) TFM is a planning process that can have significant impacts on efficiency but is not a safety-critical operation (i.e., as separation assurance is provided by Air Traffic Control (ATC)), and 3) despite significant previous work, a single satisfactory approach for providing real-time decision support has remained elusive.

The challenge of designing TFM strategies – which comprise a time series of Traffic Management Initiatives (TMIs) – is that the planning horizon, especially for some TMIs, may be several hours and both weather and traffic forecast uncertainty remains high at these look ahead times (LATs). Furthermore, weather forecast uncertainty, which can be characterized by an ensemble of deterministic weather futures generated through varying parameters in physics-based models, is not well represented using standard statistical distributions. Given the complexity of the network effects, the decision on when to implement which combination of management actions is not intuitive.

The need for sequential decision-making approaches (i.e., methods that account for the dynamics of uncertainty in the timing of decisions) to address challenges in the TFM space has been long recognized. One prominent approach has been to leverage stochastic mixed-integer programming algorithms to address GDP planning [15-18]. In these earlier works, the

planning scenarios for which the strategy is optimized are constructed by the authors, as opposed to derived from the underlying dynamics of the forecasted weather. Reference [19] used similarly constructed scenarios to evaluate a proposed change to the GDP slot assignment logic, from the airline perspective. Research into the derivation of scenarios from weather forecasts have similarly made significant progress; however, these papers did not fully connect the decision-making challenges under uncertainty to the scenarios derived [20] [21].

In our previous work [22], we developed an Advanced Planning Framework (APF) – a decision tree that was derived from an ensemble weather forecast product to directly capture the range of potential weather-induced constraints – and examined its utility for designing TMIs; however, while the preliminary results showed promise, the investigation was limited due to computational considerations. Applying the APF to the strategic flight cancellation problem showed that improved decision skill could be achieved with a longer decision clock and a smaller decision space [23].

In this paper, we revisit the problem of generating TMI strategies that adapt under changing forecast information. Leveraging insights from the APF, we implement our Monte Carlo Tree Search (MCTS) based on the evolution of the forecast ensemble members, which allows a more precise estimation of future expected costs under different selections of trial TMIs. While the planning aspect leverages the forecast, the propagation of the model, including the impact of any TMI, is assessed against observed capacities, replicating the challenge of managing demand in the presence of forecast uncertainty. To demonstrate the proposed approach, our case study focuses on managing arrivals into Atlanta Hartsfield International Airport (ATL). Using historical forecasts and observed data, we generate TMI strategies for a subset of scenario days spanning 3 non-consecutive months in 2019, chosen to capture different weather (and therefore, forecast) phenomena.

The remainder of the paper is organized as follows. Section II details the environment for our case study, including the scenarios generated, TMIs considered, simulation model, and metrics used to assess the results. Section III describes the MCTS approach, including the process for leveraging the forecast data. Section IV presents the results, and Section V provides additional discussion on the future directions for this work.

II. Case Study

Our case study focuses on designing TMIs to manage arrival demand into ATL under uncertain weather and potentially degraded capacity conditions. As shown in Figure 1, our representation abstracts the arrival routes into a four corner-post configuration common at ATL. The four corner-posts, termed fixes for the remainder of the paper, are labeled as Northwest (NW), Northeast (NE), Southwest (SW), and Southeast (SE) and are positioned 40 km from the airport, which is shown in the center of the circle.

In this section, we describe the development of the scenarios, the TMI options permitted, the queuing simulation that propagates the demand, and the metrics used to evaluate performance.

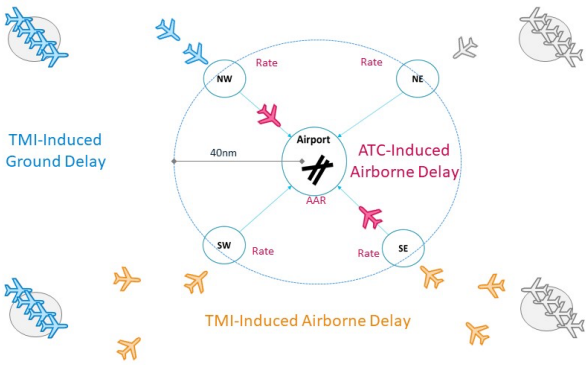


Figure 1. Depiction of ATL Corner Post Configuration

A. Scenario Generation

Each scenario corresponds to a historical day, where scenario data was generated for the period between January 1, 2015 and January 1, 2020. Each scenario contains a time history of demand at each resource (the four fixes and the airport) and the capacities – both derived from forecast and observation – for these resources. For the remainder of this paper, we will refer to capacities derived from observed data as actual capacities.

1) Demand

To obtain traffic data at each of the fixes, the first filed flight plan for each flight arriving to ATL is extracted from Aggregate Demand List (ADL) data and contains the flight ID, origin airport, original estimated time of departure (OETD), and original estimated time of arrival (OETA). Using Jeppesen data, each published Standard Terminal Arrival (STAR) route for ATL was mapped to one of the four fixes shown in Figure 1. As such, any flight with a flight plan containing a published STAR could be automatically assigned to a fix. For flights without a STAR, we assigned the most common fix used by that flight's year, flight ID, aircraft type, or origin airport, based on the data available.

The estimated time en route (ETE) is calculated as the difference between the OETA and OETD. To compute the fix-to-airport transit time, we conducted a historical analysis of 2018 track data and used the median value. The scheduled time of arrival at the fix is the OETA less the fix-to-airport transit time.

2) Airport Capacity

Airport capacities were derived from an analysis of Aviation System Performance Metrics (ASPM) runway configuration data from 2015-2019, where the most common configuration was 26R 27L 28 | 26L 27R. Reference [24] provides the Airport Arrival Rates (AAR) for each of the four meteorological conditions: Visual MC (VMC), Low VMC (LVMC), Instrument MC (IMC), and Low IMC (LIMC), where the published hourly rates were divided by four and rounded down to provide the 15-minute values shown in Table 1. The ceiling and visibility rules described in Table 1 were taken from [24] and [25].

Table 1. Meteorological Conditions for ATL Case Scenario

Category	**Visual MC (VMC)**	**Low VMC (LVMC)**	**Instrument MC (IMC)**	**Low IMC (LIMC)**
Rate per 15 min	33	31	27	24

Ceiling and Visibility conditions	ceil ≥ 3600 ft and vis ≥ 7 SM	ceil < 3600 ft or vis < 7 SM	ceil < 1000 ft or vis < 3 SM	ceil < 500 ft or vis < 1 SM

To compute the applicable MC rate, we used Automated Surface Observing System (ASOS) and Meteorological Terminal Air Report (METAR) data to identify the observed ceiling and visibility for each 15-minute period and recorded the associated rate as the capacity for the airport at that time.

3) Fix Capacities

As noted, the corner post fixes represent an aggregation of arrival routes and thus do not have a published nominal capacity. To estimate these values, we analyzed the historical arrival throughput in each quadrant and computed the 95% percentile value, as shown in Table 2.

Table 2. Nominal Fix Capacities per 15-minute time bin

Fix	**NW**	**NE**	**SW**	**SE**
Capacity	10	10	7	7

To compute the weather-impacted capacities, we leverage Corridor Integrated Weather System (CIWS) nowcast to provide measurements of Vertically Integrated Liquid greater than or equal to 3 mm of surface accumulation (VIL3+) in the 80 NM area surrounding the airport for each 15-minute period. We compute the weather-impacted capacity of fix i at time t ($C_{i,t}^{W}$) as a fraction of the nominal capacity (C_i^N) using the relationship provided in [26]. For our purposes, however, we limit the capacity reduction to 50%, as shown in Equation 1.

$$C_{i,t}^{W} = C_i^N \times \max\left[\left(1 - 2\frac{N_{i,t}^{VIL}}{N_i}\right), 0.5\right] \qquad 1$$

Here, $N_{i,t}^{VIL}$ is the number of grid points in the enlarged quadrant with VIL3+ weather, and N_i is the number of grid points captured by the quadrant. The values computed represent the actual fix capacities for the scenario.

4) Predicted Capacities

Each scenario includes a set of predicted resource capacities which will be used by MCTS to generate TMI actions, as opposed to the actual capacities which are used by the simulation. Specifically, we leverage the Short-Range Ensemble Forecast (SREF), which consists of 26 deterministic trajectories of weather variables at hour-long intervals. It is assumed that each member of the ensemble is equally likely to occur and, together, the ensemble members span the space of future outcomes [27]. Furthermore, a new SREF is issued every 6 hours.

To compute the predicted AAR, we identify the SREF grid cell that contains the airport center. Using the ceiling and visibility variables, we can directly calculate the MC for the hour. To obtain the 15-minute predictions, we divide the hourly rate by four, rounding down to the integer.

To compute the predicted fix capacities, we identify the proportion of SREF grid points in the extended 80-km quadrant. However, as the SREF does not contain VIL measurements, we use the reflectivity values to approximate VIL3+ using the relationship described in [28]. Using a threshold of reflectivity >38 dBZ to identify VIL3+ conditions, we compute the weather-impacted capacity using Equation 1. Again, the value is divided by four and rounded down to the integer to generate 15-minute capacities.

The resulting predicted capacity ensemble contains 26 members, where each member contains the 15-minute integer capacities for each of the five resources (airport and four fixes).

B. Traffic Management Intitiatives

Two types of TMIs are considered in this case study: GDPs and metering.

1) Ground Delay Program

A Ground Delay Program (GDP) is a strategic TMI that delays flights on the ground prior to departure. The GDP is defined by four parameters:

- Rate: The maximum quarterly arrival rate for flights.
- Scope: The scope defines the set of origin airports whose departures are subject to delays by the GDP.
- Start time: The start time of the GDP expressed in local time at the destination airport.
- Duration: The duration of the GDP.

For the case study in this paper, the scope was set to include all departures.

The GDP implementation was based on the logic of Flight Schedule Monitor (FSM) [28] and incorporates the option to cancel or revise (i.e., alter the parameters of an existing GDP) an existing GDP or to revise a GDP. Arrival slots are generated in accordance with the specified GDP rates and a ration-by-schedule logic is used to assign slots based on their scheduled arrival times (OETA). Controlled arrival times (CTAs) and controlled departure times (CTDs) are computed based on the assigned arrival slot times and the flights' ETEs.

In the case of revisions, delay is released (to the extent possible) from flights no longer included in the revised GDP and re-assigned based on the new slots, where flights that were impacted by the previous GDP have precedence over flights that were not included. Additionally, flights can be exempt for several reasons (scope, departure time), and exempt flights are assigned slots before any other groups. In contrast to non-exempt flights, exempt flights take up slots and are assigned CTDs and CTAs but are not delayed.

2) Metering

The metering TMI is intended to represent coordinated air delay assigned to flights prior to arrival in the terminal airspace. Each fix can have a separate "metering TMI" that is defined by:

- Rate: The permissible arrival rate per 15 minutes. This rate is translated into a new time of arrival at each corner-post fix.
- Start time: The start time of the rate restriction.
- Duration: The duration of the metering program in minutes.

A flight's actual fix crossing time can be delayed relative to the scheduled crossing time by Air Traffic Control (ATC), metering, or some combination of the two. When a fight's fix

crossing event is processed by the simulation, the scheduled crossing time – call it t_B – is compared to the previous flight's crossing time at the fix – call it t_A. If the time gap $t_B - t_A$ between flights exceeds the minimum time constraint imposed by both the fix capacity and the current metering restriction (if any), then the flight crosses the fix at the scheduled time without any delay. Otherwise, the flight's fix crossing time is delayed by the smallest amount such that both the fix capacity and metering time constraints are satisfied. Note that delay is first attributed to the metering restriction (if any), and any additional required delay is attributed to ATC. Note that while the metering TMI cannot be canceled, the agent may override the current metering parameters at any fix by simply implementing a new metering action.

C. Simulation Model

The simulation is a bi-level queuing model [29], where each fix has an associated queue which then feeds the airport queue. The arrival time of each flight to each fix is defined by the scenario demand data. While these times may change due to the implementation of a TMI, this impact occurs prior to the time period where the flight would arrive at the fix. As such, the simulation proceeds with the current inventory list and processes flights based on the actual capacities associated with each resource. The simulation terminates when all flights have passed the airport queue or after 24 hours.

D. Metrics

1) Delay and Reward Function

The total delay accrued for each flight is the sum of the TMI Ground Delay (d^g), TMI Air Delay (d^m), and ATC-induced Delay (d^a). The TMI Ground Delay for a flight is calculated as the difference between the CTD and the OETD. TMI Air Delay is measured as the difference between the scheduled fix arrival time and the assigned fix arrival time resulting from the metering TMI. ATC-induced delay is the total queuing delay imposed by the simulation, capturing both queues at the fix and at the airport.

The reward function is defined to minimize the *delay impact* for the entire scenario, where delay impact represents a non-linear aggregation of the three components of delay. Using Subject Matter Expert (SME) guidance, the delay impact for each type of delay was defined as a piece-wise linear function to capture not only the difference between sources of delay but how the duration of the delay changes the impact. These relationships are shown in Equations 2-4 and depicted in Figure 2.

$$I^g = \begin{cases} 10 \cdot d^g & d^g \leq 2 \\ 20 + (d^g - 2) & 2 < d^g \leq 15 \\ 35 + 2 \cdot (d^g - 15) & 15 < d^g \leq 60 \\ 1000 & d^g > 60 \end{cases} \quad 2$$

$$I^m = \begin{cases} 10 \cdot d^m & d^m \leq 1 \\ 10 + 2 \cdot (d^m - 1) & 1 < d^m \leq 15 \\ 40 + 3 \cdot (d^m - 15) & 15 < d^m \leq 30 \\ 2000 & d^m > 30 \end{cases} \quad 3$$

$$I^a = \begin{cases} 5 \cdot d^a & d^a \leq 15 \\ 75 + 10 \cdot (d^a - 15) & 15 < d^a \leq 30 \\ 3000 & d^a > 30 \end{cases} \quad 4$$

Viewing Figure 2, we see that for flight specific delays that are less than two minutes, ATC-induced delay (magenta line) induces the least delay impact; however, as ATC delays increase, delay impact grows at the fastest rate and for 30 minutes or more a large penalty is assigned to represent the disruption of potentially requiring a diversion. For flight-specific delays between two and ten minutes, metering delay (orange line), induces the least delay impact and increases more slowly than ATC delay, but still assigns a large penalty for delays over 30 minutes. GDP delay is the least impactful for delays over 10 minutes and does not have a large penalty assigned at higher delay values.

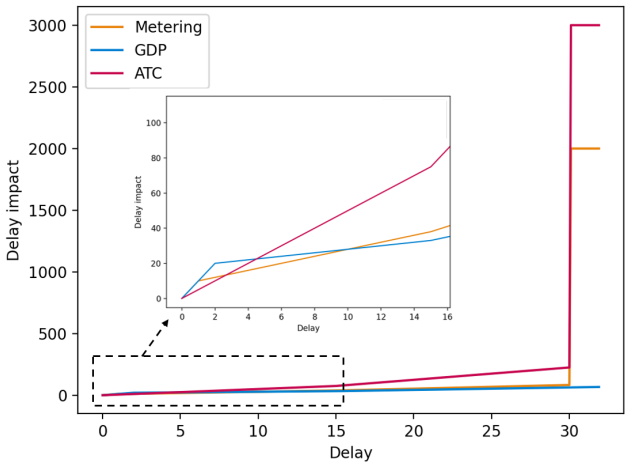


Figure 2. Delay Impact vs. Delay Minutes

The reward function computes the negative of the total delay impact for all flights (f), as shown in Equation 5.

$$Reward = -\sum_f \left(I_f^g + I_f^m + I_f^a\right) \quad 5$$

2) Airport Utilization

While the selection of the TMI strategy is determined solely based on the reward function described above, we are also interested in measuring how efficiently the selected strategy uses available capacity. To compute this, we define the Airport Utilization at time t (AU_t) and the total airport utilization for the scenario as (AU) as shown in Equations 6 and 7, respectively.

$$AU_t = NA_t / \min\left(AD_t^0, AAR_t\right) \quad 6$$

$$AU = \sum_t AU_t / \mathrm{T} \quad 7$$

Here, NA_t is the number of aircraft that process through the airport queue in time t, AD_t^0 is the number of aircraft that were originally scheduled to arrive in that time period, as determined by each flight's OETA, and AAR_t is the actual airport capacity, or AAR, at that time. The total airport utilization is the average over the airport utilization at each time step, where T is the total number of time steps in the scenario. Note that in cases where the denominator of Equation 6 is zero (e.g., there was no scheduled demand), we set $AU_t = 1$.

3) Forecast Uncertainty

The predicted capacity ensemble described in Section II.A captures the information used by MCTS to plan TMI actions. As it is important to distinguish the planning skill relative to the certainty of the forecast information, we compute the informational entropy associated with each resource using the first SREF issuance of each scenario day [30] [31].

For each resource, there are a finite number of capacity levels available at each time. Using the airport as an example, one of four MC levels may be assigned at each time bin for each member. If instead of viewing the 26 members individually, we compute the number of members assigned to each capacity bin, we can evaluate the spread of information across both capacity bins and time periods. Specifically, if we define E_{jt} to be the number of ensemble members with capacity level j in time bin t, we can compute the normalized probability distribution across resource levels and time bins using Equation 8.

$$p_{jt} = \frac{E_{jt}}{\sum_j E_{jt}} \ \forall t \qquad 8$$

Note that the denominator is equal to 26 in our example. The entropy associated with that resource's forecast is then defined as shown in Equation 9, where T is the total number of time bins.

$$H = -\frac{1}{T}\sum_j \sum_t p_{j,t} ln\,(p_{j,t}) \qquad 9$$

III. TFM Agent

In the AI domain, the *agent* refers to the automation that identifies and selects the action (i.e., TMI) to implement with the goal of maximizing a reward, which in this work corresponds to minimizing delay impact. In this paper, the agent uses MCTS to select the best action at the current (hourly) decision time t which the agent estimates will achieve the lowest future expected delay impact. During the one-hour decision duration allotted for planning, MCTS is applied to build a tree that estimates the optimal *adaptive* TMI policy with respect to the possible future capacity trajectories as derived from the capacity ensemble. The policy is *adaptive* over the target planning horizon in the sense that the best action at time t takes into account future contingencies, i.e., downstream TMI actions that could be triggered by future observed capacities. Note, however, that the time t tree is used to select only the time t action, and a new tree is built to select the action at each subsequent decision time. As is depicted in Figure 3, the agent's action at time t results in an updated system state, which is used to initialize the topmost "root" node of the time $t + 1$ tree.

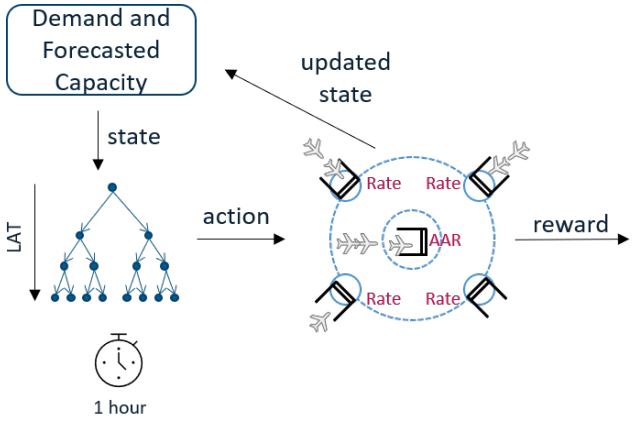


Figure 3. Overview of MCTS Decision Process

At decision time t we use MCTS to select an action with the goal of reducing future expected delay impact. Our MCTS algorithm is initialized by defining a root node N at depth $d = 0$. The root node's state reflects the current state of the TFM environment at time t, which accounts for all actions taken up to time t, all actual capacities observed up to time t, and the impact on the flight schedules. At the root node, we assume that all 26 members of the capacity ensemble are equally likely, as they were derived from the SREF which relies on that assumption. Furthermore, we assign all 26 members to the same partition S_N for root node N, since – relative to the root time t – no capacities have been observed to distinguish one member from another.

Before optimizing over TMI actions, we first construct a "baseline" tree from the root downward that captures the space of possible futures when no TMI actions are taken. The possible futures are enumerated in tree form as follows: From the root node, first identify which members in the capacity ensemble remain consistent over the next hour. Here, two members are said to remain consistent if and only if their binned capacities are identical across all 5 resources over the 4 (15-minute separated) observations in the next hour. For the airport resource, the capacity bins are aligned with the four MC capacities. For the NW and NE fixes, 6 capacity levels (integral values between 5 and 10, inclusive) may be present in the forecast. For these two resources we define 5 capacity bins, where the lowest two capacities are binned together. For the SW and SE fixes, 5 capacity levels (integral values between 3 and 7, inclusive) may be present. For these resources we define 4 capacity bins, where, again, the lowest two capacities are binned together. This approach is similar to that used in [23].

Ensemble members that are consistent over the next hour are placed into the same partition at depth $d = 1$, and associated with each partition we create a child node of the root. If we let N represent the root, then $N = Parent(C)$ for all child nodes C. For the baseline tree construction, the parent-to-child transition also assumes that no TMI action is taken. For simplicity, we represent this action as $\tilde{a}$. We apply this branching process recursively to each child to create a complete tree. Note that we terminate the branching process when the child node depth reaches a predefined target value of $d_{max} = 5$ – or the child node represents a terminating state of the simulation, whichever comes first. By limiting the depth of the tree in this way, we ensure that the MCTS algorithm will focus its policy search on more immediate actions (here, over the next 5 hours) where the future – and hence action impact – is more certain.

The next step is to initialize a value function at all nodes in the baseline tree. This process begins at the leaf nodes of the baseline tree with depth d_{max} and propagates upward to the root. Assuming that a given leaf node N does not represent a terminating state of the simulation, we estimate the value $V_{N,\tilde{a}}$ of action $\tilde{a}$ at the node by performing a *rollout* from N. In our MCTS implementation, a rollout simulates the current node state to the completion of the simulation, such that all future actions default to $\tilde{a}$. In particular, the rollout is executed over all ensemble members in S_N and their resulting future rewards are averaged to obtain $V_{N,\tilde{a}}$. The optimal action at node N is given by

$$a_N^* = \underset{a}{\operatorname{argmax}}\, V_{N,a} \qquad 10$$

where the argmax is taken over all actions a explored at node N. In the baseline tree, $a_N^* = \tilde{a}$ since no other actions have yet been

explored. The corresponding optimal value at node N is denoted $V_N = V_{N,a_N^*}$. Next, we use dynamic programming to recursively update parent node values at action $\tilde{a}$:

$$V_{N,\tilde{a}} = \sum_{C \in Children(N,\tilde{a})} \frac{1}{w_C}\left(\bar{r}_{C,\tilde{a}} + V_{C,\tilde{a}}\right) \qquad 11$$

Here w_C denotes the proportion of ensemble members in S_N that appear in child C, and $\bar{r}_{C,\tilde{a}}$ denotes the immediate reward received when action $\tilde{a}$ is taken at node N and we assume the capacities over the transition from N to C are the average capacities (by resource) over the ensemble members in S_C. Just like with the leaf nodes, we let $a_N^* = \tilde{a}$ represent the optimal action at each subsequent parent node N until the recursive process terminates at the root. A notional illustration of the resulting baseline tree is depicted in Figure 4, where we use $d_{max} = 2$ for the sake of simplicity.

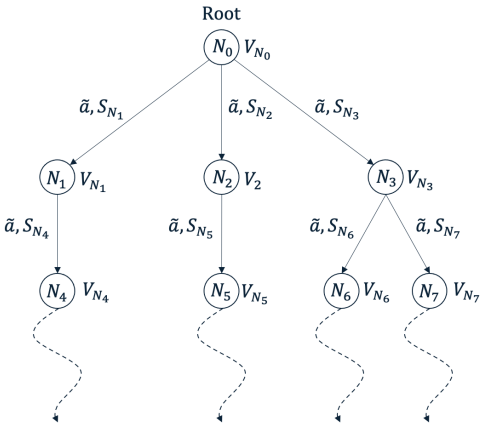


Figure 4. Notional depiction of baseline MCTS tree with max depth equal to two, for simplicity. Dashed lines represent rollouts from leaf nodes of the tree. Note that all transitions from parent to child involve the "no TMI" action $\tilde{a}$.

Once the baseline tree is constructed, the MCTS algorithm explores new actions in an effort to improve upon the baseline policy. Our specific MCTS implementation is inspired by a MCTS algorithm developed for the Combinatorial Multi-armed Bandit problem [32]. In that paper, the authors propose a factored ϵ-greedy strategy to balance the tradeoff between *exploration* – i.e., randomly sampling new actions – and *exploitation* – i.e., resampling actions that are known to be good – when the action space is combinatorially large. As discussed in the next section, our problem's action space falls into this category, admitting more than 74 million possible actions at each decision time. This is because the joint action space is expressed as the product of the 5 resources' individual action spaces.

Given a baseline tree, each iteration of our MCTS algorithm proceeds as follows. First, the ϵ-greedy strategy described in [32] is used to select an action at the root node N. Subject to user-defined probabilities, the strategy either (i) greedily selects the current best action a_N^*, (ii) randomly samples a joint action uniformly from the list of joint actions previously taken at the node, (iii) greedily selects an action for each resource (independently), or (iv) randomly samples an action for each resource uniformly (independently). The greedy approach of option (iii) is based on the "naïve" assumption that the joint action value function V can be decomposed as a sum over value functions of the independent resource-specific actions. In practice, this assumption need only hold loosely for the sampling strategy to perform well [32].

If the action a sampled at root node N by the ϵ-greedy strategy has not been sampled before, then we create new child nodes of N based on the partitioning of S_N over the next one hour transition. From these child nodes we conduct rollouts, initialize their value functions, and recursively update the parent's value function for the action a just sampled.

On the other hand, if the action a has been sampled before, then we sample an existing child $C \in Children(N, a)$ such that each child has probability w_C of being selected. We carry out the ϵ-greedy strategy from the child just as before. In general, the forward sampling process continues until we either sample a previously unexplored action or reach a leaf node corresponding to a terminal simulation state. In both cases, we perform recursive value function updates back up the tree to the root node, effectively backtracking the forward sampled path along the way. This process of forward sampling followed by rollout and backpropagation is repeated until 1 hour has passed, after which the agent implements the action $a_N^* = \underset{a}{\operatorname{argmax}} V_{N,a}$ based on the value function estimates at the root node N.

IV. Results

A. Experimental Design

To evaluate the performance of the TFM agent, we generated TFM strategies for three non-consecutive months: June, October, and December 2019. For each scenario day, the agent was provided with the capacity ensemble and had an hour to generate the recommended action for the upcoming hour. For the two TMIs described in Section II.B, we permitted the agent to select among the TMI parameter values listed in Table 3.

Table 3. TMI Parameter Options

TMI Parameters	**Start Time**	**Duration**	**Rate**
GDP	{60, 120, 180, 270}	{120, 180, 240, 300}	{0.5, 0.6, 0.7, 0.8, 0.9}
Metering	{30, 60}	{30, 60, 90}	{0.5, 0.6, 0.7, 0.8, 0.9}

As shown in Table 3, the TMI start time (in minutes) specifies the offset between the decision time and the start of the program. For the GDP, this offset can dictate which flights will receive ground delay as flights within 30 minutes of their OETD will be exempt. The duration (in minutes) specifies how long the TMI will be in effect and the rate specifies the fraction of nominal 15-minute capacity that will be imposed as the rate. In addition to these options, the agent can opt to take no action, cancel, or revise an existing action. Note that while only two TMIs are defined and only a handful of parameter values are considered for each, the resulting design space has more than 74 million discrete choices at each time step.

B. *Single Day Evaluation*

Before presenting the results for the 3-month period, we first analyze two separate days in greater detail to highlight both the capability of the model and the challenges wrought by forecast uncertainty. To evaluate the impact of forecast uncertainty, we compare the results generated by the MCTS as described in Section III with a MCTS implementation that was provided the actual capacity values. While having a perfect prediction of future capacity is unrealistic, this comparison provides the lower bound on the delay impact that can be achieved.

1) *June 10th 2019*

The 10 June scenario day highlights the TFM agent's ability to respond to uncertain capacity imbalances as well as adapt the strategy as new observations are acquired. Figure 5 shows the capacity forecasts for each of the five resources, where the 26 ensemble members are shown using colored lines. The black line in Figure 5 corresponds to the actual capacities of the resources. Note that all times are relative to the start of the simulation; hour zero is 6AM local time at ATL.

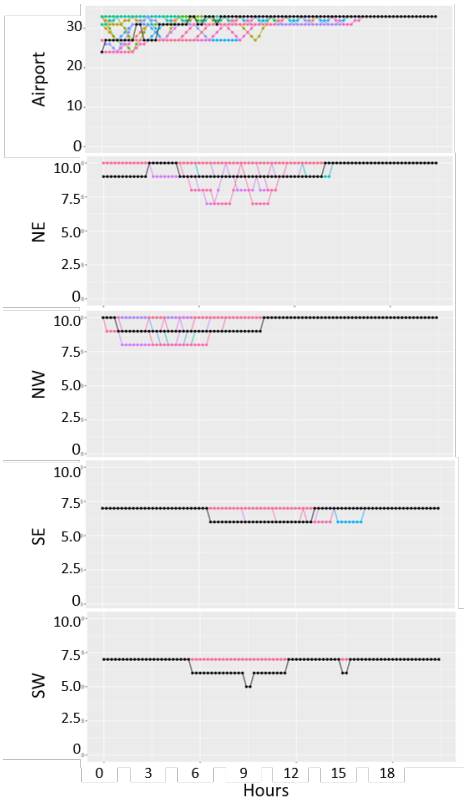
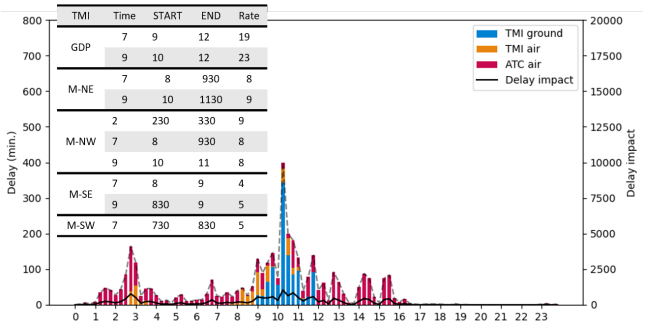


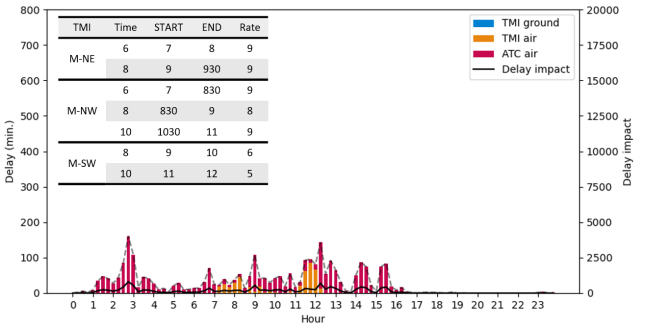
Figure 5. June 10th Forecasted and Actual Capacities

Figures 6 and 7 show the TMIs – abbreviated as GDP or M-fix for metering at the specific fix – accumulated delay by source, and delay impact for the MCTS under forecast uncertainty and with perfect information, respectively. Comparing the two plots, we readily notice that with perfect information (Figure 7), the capacity imbalance is managed solely through metering (shown as orange delays), whereas the strategy shown in Figure 6 contains both metering and GDP TMIs (shown as blue delays). Referring to Figure 5, however, we see that the forecasted capacities at the airport, NE and NW fixes are sustained at a rate below that of the actual capacity and with such an agreement between the members, the MCTS



TMI	Time	START	END	Rate
GDP	7	9	12	19
	9	10	12	23
M-NE	7	8	930	8
	9	10	1130	9
M-NW	2	230	330	9
	7	8	930	8
	9	10	11	8
M-SE	7	8	9	4
	9	830	9	5
M-SW	7	730	830	5

Figure 6. June 10th MCTS TMI Strategy under Uncertainty



TMI	Time	START	END	Rate
M-NE	6	7	8	9
	8	9	930	9
M-NW	6	7	830	9
	8	830	9	8
	10	1030	11	9
M-SW	8	9	10	6
	10	11	12	5

Figure 7. June 10th MCTS TMI Strategy with Perfect Information

determines the best strategy is to respond to the forecasted imbalances.

While unnecessary control is less than desirable, it is worth highlighting two behaviors. First, in Figure 6 we note that two GDPs are issued; on closer inspection, we see that the 2nd GDP issued at hour 9 is a revision of the first GDP, raising the rate. As such, the TFM agent effectively adapts to the changing information regarding the delays and capacities observed. In addition, while the total delay is higher in Figure 6, the delay impact is much closer to that achieved with perfect information. Given that delay impact and not total delay is the objective sought by the agent, these results conform to our performance expectation.

2) *June 8th 2019*

The 8 June scenario day is challenging not only because of the significant capacity reduction that occurred but also because this reduction was not well forecasted, as shown in Figure 8. Viewing Figure 8, we note that the underlying assumption about the ensemble, namely that it spans the space of future outcomes, does not hold on this day as the actual capacities are lower than any of the forecasted values for several time periods at multiple resources.

Figures 9 and 10 show the delay and delay impact under forecast uncertainty and with perfect information, respectively. With perfect knowledge (Figure 10), a multi-hour strategy consisting of both GDP and metering actions across all four fixes, starting at hour three, can effectively reduce the delay impact associated with this scenario.

When basing its decisions on forecast information, however, the TFM agent is not able to proactively act to successfully reduce the large delay impact. Instead, metering actions are used to mitigate some of the impact as the implementation times are shorter and therefore can be issued

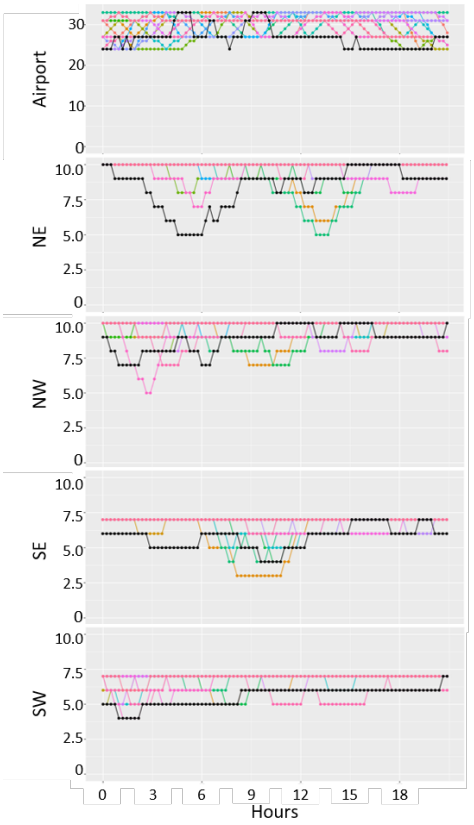


Figure 8. June 8th Forecasted and Actual Capacities

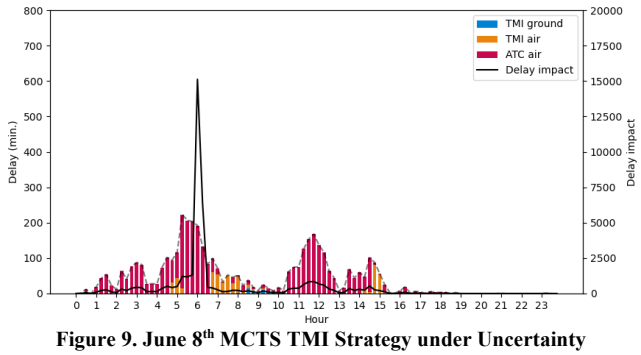


Figure 9. June 8th MCTS TMI Strategy under Uncertainty

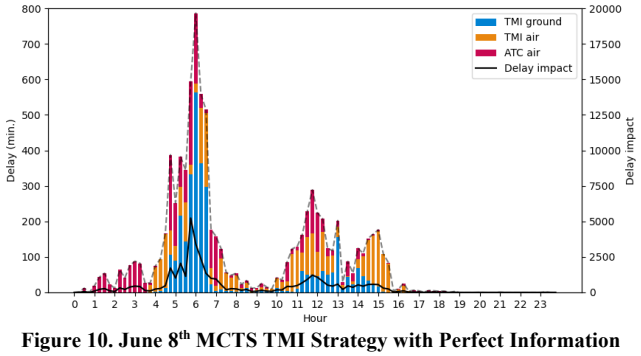


Figure 10. June 8th MCTS TMI Strategy with Perfect Information

closer to the time of the constraint. Thus, while the overall delays are lower, much of the imbalance is not managed by the TMIs and instead handled tactically by ATC, resulting in significantly higher delay impact at hour 6. Again, while there is no expectation of perfect information in TFM planning, having a forecast that captures the range of future outcomes is critical and without it, the results generated by any decision support systems will suffer from these discrepancies.

C. Analysis of Three Months of Testing Data

Given the insight provided by our detailed case-day analyses, we next evaluate the TFM agent's performance across our three-month testing period. For each scenario day in the period, we compute the delay impact, airport utilization and forecast entropy, as described in Section II.D. Figure 11 displays these results.

Viewing Figure 11, we see that across the three-month testing data, the agent is able to keep delay impact to a value similar to the June 10th case day, with two exceptions. The first peak corresponds to the June 8th day examined in the previous section while the second day corresponds to June 22nd. It is worth noting that on this day, even with perfect information and pro-active action, the delay impact is similar to that obtained by the TFM agent under uncertainty, 140,356 verses 156,155, respectively.

By examining the second plot in Figure 11, we see that the delay impact values do not fluctuate with respect to the forecast uncertainty (2nd plot in Figure 11). Furthermore, the airport utilization remains high with minimum values greater than 95%, implying that the agent is not over-controlling the scenario. Taken together, these results demonstrate that the MCTS can effectively respond to uncertain capacity information when generating TMI strategies.

V. Discussion

In this paper, we described the development of a MCTS-based TFM agent capable of designing TMI strategies that directly capture the complexities associated with weather forecast uncertainties and adapting to changing information in a real-time context. By leveraging the ensemble forecast to design the decision tree, the agent was able to readily evaluate how potential strategies would evolve under the different forecast futures. Furthermore, by limiting the decision time to one hour for the next recommendation, we demonstrated that such an approach holds promise for real-time applications. That said, there are several aspects of this challenging domain that require additional investigation. The remainder of this section provides some discussion on these important directions.

The decision to embed the forecast information into the structure of the tree search resulted in a computationally tractable framework in which to evaluate the enormous design space associated with this problem. However, as our analysis of the June 8th scenario showed, this reliance can result in missed opportunities if the forecast does not reflect the range of potential future outcomes. One approach to mitigate these situations is by introducing a data assimilation function that takes as input both (i) an ensemble member's capacity forecast and (ii) observed capacity values at each resource on the current day and returns a "corrected" forecast that aims to minimize the error between it and the day's remaining capacity values. However, the training of such a function (e.g., as a neural network) would require a wealth of historical data, and there is no guarantee that the resulting capacity projections would satisfy physics constraints in the way that the original ensemble members do. As a result, we believe the development of

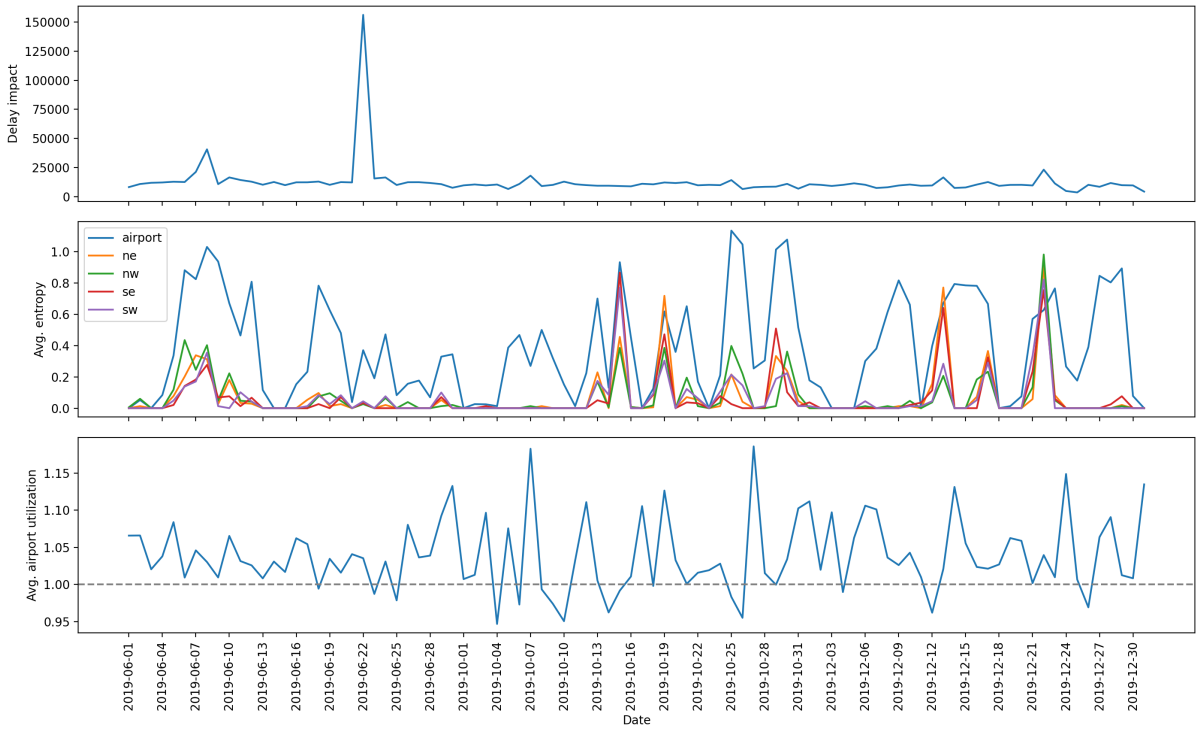


Figure 11. Delay Impact, Average Forecast Entropy and Average Airport Utilization for Three-Month Testing Data

improved physics-based ensemble forecasting models may be a better long-term solution.

In addition, any automation designed to provide recommended strategies in a real time environment requires a well-defined objective function. For AI algorithms, however, it is imperative as the automation is not simply evaluating and ranking candidate solutions but effectively encoding what a good solution entails. Even if the automation could ensure optimality, if the objective does not truly reflect the priorities of the operational environment, the results generated will not provide substantive decision support.

While the MCTS-based agent developed in this paper was able to learn a TMI strategy for a given scenario, it was not able to generalize this information across scenario days. A natural next step is to replicate the success of AlphaGo [2] and AlphaStar [3] by combining MCTS with DRL to iteratively learn neural network-based policy and value functions. Specifically, the value network could generate more accurate future expected rewards than the current simulated rollouts – which are limited, in part, by assuming no TMIs are used from the current time onward – and the policy network could be used to sample actions during the search based on which performed best under similar historical conditions. Yet, there exist fundamental differences between the games played by AlphaGo and AlphaStar and the TFM domain. First, whereas the self-play nature of games like Go enable the agent to generate samples that are specifically targeted to address shortcomings in the agent's current policy network, in the TFM context the range of samples is limited to the weather behavior observed in the static historical training set. In practice, the training set may not capture the range of weather events that may be encountered on future days, and so the policy network may generalize poorly outside of the training set. This is a particular concern since impactful weather days – where decision support is most needed – are comparatively rare in historical data.

The MCTS results presented in this paper overcome a major challenge noted in our previous APF research [22] [23] in that the TFM agent can generate recommendations in a real-time context. But beyond efficiency gains, this approach can be extended to incorporate additional learned responses, something that neither the APF nor related research [15-21] could accommodate. First, the MCTS can be paired with an *expert* policy model – a separate ML algorithm that leverages historical data to inform the search and value updates. The expert policy would function analogously to the DRL-based policy network discussed but would be trained independent from and prior to the execution of MCTS. Secondly, by incorporating observations through data assimilation, subsequent tree constructs could be optimized to account for knowledge gained. Finally, the approach could incorporate a DRL algorithm, encoding the relationship between operational situation, TMI strategy, and positive outcome in neural network-based policy and value functions. As such, there are many future directions that can be evaluated to achieve the automation enhancements essential to achieving the future vision for TFM.

NOTICE

This work was produced for the U.S. Government under Contract DTFAWA-10-C-00080 and is subject to Federal Aviation Administration Acquisition Management System Clause 3.5-13, Rights In Data-General, Alt. III and Alt. IV (Oct. 1996).

For further information, please contact The MITRE Corporation, Contracts Management Office, 7515 Colshire Drive, McLean, VA 22102-7539, (703) 983-6000.

Approved for Public Release 21-1106 Distribution Unlimited.

REFERENCES

[1] The MITRE Corporation "A 2035 Vision for Air Traffic Management Services", May 2020, *Preliminary draft*

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L Sifre, D. van den Driessche, et al., "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484-489, 2016. doi: 10.1038/nature16961.

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning" *Nature* 575, 350-354, October 2019, doi: 10.1038/s41586-019-1724-z

[4] H. Lee, W. Malik, and Y. C. Jung, "Taxi-out time prediction for departures at charlotte airport using machine learning techniques," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3910.

[5] G. Gui, Z. Zhou, J. Wang, F. Liu, and J. Sun, "Machine learning aided air traffic flow analysis based on aviation big data", IEEE Transactions on Vehicular Technology, Vol. 69, No. 5., May 2020, doi: 10.1109/TVT.2020.2981959

[6] M. C. R. Murca, R. DeLaura, R. J. Hansman, R. Jordan, T. Reynolds, and H. Balakrishnan, "Trajectory clustering and classification for characterization of air traffic flows," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3760.

[7] F. Herrema, V. Treve, B. Desart, R. Curran, and D. Visser, "A novel machine learning model to predict abnormal runway occupancy times and observe related precursors", *12th USA/Europe Air Traffic Management R&D Seminar*, Seattle, WA, June 2017., #107.

[8] N. Takeichi, R. Kaida, A. Shimomura, and T. Yamauchi, "Prediction of delay due to air traffic control by machine learning" *AIAA SciTech Forum,* 9-13 January 2017, doi: 10.2514/6.2017-1323.

[9] A. Evans and P. Lee, "Using machine-learning to dynamically generate operationally acceptable strategic reroute options", *13th USA/Europe Air Traffic Management R&D Seminar*, June 2019, Vienna Austria, Paper # 25.

[10] E. Vargo, C. Taylor, and C. Wanke, "Probabilistic time-series models for ground delay program decision support", *AIAA Aviation Forum* 22-26 June, 2015, Dallas, TX, doi: 10.2514/6.2015-3333

[11] S-L. Tien, H. Tang, D. Kirk, E. Vargo, and S. Liu, "Deep Reinforcement Learning Applied to Airport Surface Movement Planning", *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC),* 8-12 Sept. 2019, San Diego, CA, doi: 10.1109/DASC43569.2019.9081720.

[12] M. Brittain and P. Wei, "Autonomous Air Traffic Controller: A Deep Multi-Agent Reinforcement Learning Approach," *arXiv preprint* arXiv:0902.0885, 2019.

[13] Pham, D-T, Tran N. P., Alam, S., Duong, V., and Delahaye, D., "A Machine Learning Approach for Conflict Resolution in Dense Traffic Scnearios with Uncertainties", 13th USA/Europe Air Traffic Management R&D Seminar, June 2019, Vienna Austria, Paper #18.

[14] Tran, P. N., Pham, D-T., Goh, S. K., Alam, S., Duong, V., "An Interactive Conflict Solver for Learning Air Traffic Conflict Resolutions", AIAA Journal of Aerospace Information Systems, Vol., 17, No. 6, June 2020. DOI. 10.2514/1.I010807

[15] Richetta, O. and Odoni, A., "Dynamic Solution to the Ground Holding Problem in Air Traffic Control" *Transportation Research Part A*, Vol. 28, No. 3, May 1994, pp. 167-185. https://doi.org/10.1016/0965-8564(94)90015-9

[16] Ball, M., Hoffman R., Odoni, A. and Rifkin, R., "A Stochastic Integer Program with Dual Network Structure and its Application to the Ground Holding Problem", *Operations Research*, Vol 51., 167-171, 2003. https://doi.org/10.1287/opre.51.1.167.12795

[17] Mukherjee, A., "Dynamic Stochastic Optimization Models for Air Traffic Flow Management with En Route and Airport Capacity Constraints." *The 6th U.S.A./Europe Air Traffic Management Research and Development Seminar*, Baltimore, MD, June 2005.

[18] Liu, P-C. B., Hansen, M., and Mukherjee, A., "Scenario-based Air Traffic Flow Management: From Theory to Practice." *Transportation Research Part B: Methodological*, Vol. 42, No. 7, 2008, pg. 685-702. https://doi.org/10.1016/j.trb.2008.01.002

[19] Jones, J., Lovell, D., and Ball, M., "Combining Control by CTA and Dynamic En Route Speed Adjustment to Improve Ground Delay Program Performance" *The 11th U.S.A./Europe Air Traffic Management Research and Development Seminar*, Lisbon, Portugal, 2015.

[20] Hoffman, R., Krozel, J., Davidson, G., and Kierstead, D., "Probabilistic Scenario-Based Event Planning for Traffic Flow Management" *AIAA Guidance, Navigation, and Control Conference*, Hilton Head, SC, August 2007. https://doi.org/10.2514/6.2007-6361

[21] Nilim, A., El Ghaoui, L., and Duong, V. "Multi-Aircraft Routing and Traffic Flow Management Under Uncertainty." *The 5th U.S.A./Europe Air Traffic Management Research and Development Seminar*, Budapest, Hungary. 2003.

[22] Taylor, C., Masek, T., Wanke, C., Roy, S., "Designing Traffic Flow Management Strategies Under Uncertainty", *The 11th U.S.A./Europe Air Traffic Management Research and Development Seminar*, Lisbon, Portugal, 2015.

[23] Taylor, C., Tien, S-L., Vargo, E., Wanke C., "Strategic Flight Cancellation under Ground Delay program Uncertainty", *Journal of Air Transportation,* Vol 29, No.1 Jan-Mar 2021, doi: 10.2514/1.D0178.

[24] https://www.fly.faa.gov/Information/east/ztl/atl/frames.htm, Accessed 7 April 2021.

[25] https://www.faasafety.gov/gslac/alc/libview_normal.aspx?id=9091. Accessed 7 April 2021.

[26] Tien, S-L, Taylor, C., Wanke, C., "Identifying Representative Weather Scenarios for Flow Contingency Management" *AIAA Aviation Forum,* 12-14 Augut 2013, Los Angeles, CA, doi: 10.2514/6.2013-4216.

[27] Bright, D. and Nutter, P., "On the Challenges of Identifying the "Best" Ensemble Member in Operational Forecasting," *84th AMS Annual Meeting*, Seattle, WA, January 2004.

[28] FSM 9.0 Algorithm Specification. Version 1.0. December 17, 2010. Prepared for FAA by CSC. "FSM Algorithm Specifications Ver 1.0 2010-12-17.pdf".

[29] Wan, Y., Taylor, C., Roy, S., Wanke, C., and Zhou, Y., "Dynamic Queuing Network Model for Flow Contingency Management," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1380-1392, Sept. 2013, doi: 10.1109/TITS.2013.2260745.

[30] Z. Toth, Y. Zhu, and T. Marchok, "The Use of Ensembles to Identify Forecasts with Small and Large Uncertainty", *American Meteorolgical Society,* August 2001, pg 463-477. doi: 10.1175/1520-0434(2001)016<0463:TUOETI>2.0.CO;2

[31] C. E. Shannon, "A Mathematical Theory of Communication", Bell System Technical Journal, Vol 27, 1948, pp. 379-423 and 623-656 (July and October).

[32] S. Ontanon, "Combinatorial Multi-armed Bandits for Real-Time Strategy Games", *Journal of Artifiical Intelligence Research*, March 2017, Vol. 58, doi: 10.1613/jair.5398.

Christine P. Taylor is a Principal Artificail Intelligence Engineer at MITRE specializing in decision support system development for traffic flow management applications. She holds a B.S. from Cornell University, and M.S. and Ph.D. degrees in aeronautical engineering from the Massachusetts Institute of Technology.

Erik Vargo is a Lead Artificial Intelligence Engineer at The MITRE Corporation in McLean, Virginia. His research interests include the application of machine learning and probabilistic models to problems in aviation and beyond. He received his PhD in Systems Engineering from the University of Virginia in 2013.

Emily Bromberg is a Senior Artificial Intelligence Engineer at MITRE specializing in big data modeling and analysis. She holds a B.S. from the Massachusetts Institute of Technology in mathematcis.

Everett Carson is a Senior Computer Scientist at MITRE specializing in machine learning and reinforcement learning. He holds a B.S. from Boston University in computer engineering.