Auton Agent Multi-Agent Syst (2012) 24:1–25
DOI 10.1007/s10458-010-9142-5

A multiagent approach to managing air traffic flow

Adrian K. Agogino · Kagan Tumer

Published online: 27 June 2010

Abstract Intelligent air traffic flow management is one of the fundamental challenges facing the Federal Aviation Administration (FAA) today. FAA estimates put weather, routing decisions and airport condition induced delays at 1,682,700 h in 2007 (FAA OPSNET Data, US Department of Transportation website, http://www.faa.gov/data_statistics/), resulting in a staggering economic loss of over $41 billion (Joint Economic Commission Majority Staff, Your flight has been delayed again, 2008). New solutions to the flow management are needed to accommodate the threefold increase in air traffic anticipated over the next two decades. Indeed, this is a complex problem where the interactions of changing conditions (e.g., weather), conflicting priorities (e.g., different airlines), limited resources (e.g., air traffic controllers) and heavy volume (e.g., over 40,000 flights over the US airspace) demand an adaptive and robust solution. In this paper we explore a multiagent algorithm where agents use reinforcement learning (RL) to reduce congestion through local actions. Each agent is associated with a fix (a specific location in 2D space) and has one of three actions: setting separation between airplanes, ordering ground delays or performing reroutes. We simulate air traffic using FACET which is an air traffic flow simulator developed at NASA and used extensively by the FAA and industry. Our FACET simulations on both artificial and real historical data from the Chicago and New York airspaces show that agents receiving personalized rewards reduce congestion by up to 80% over agents receiving a global reward and by up to 90% over a current industry approach (Monte Carlo estimation).

Keywords Air traffic control · Multiagent learning · Agent coordination

A. K. Agogino
University of California, Santa Cruz, CA, USA
e-mail: adrian.k.agogino@nasa.gov

K. Tumer (✉)
Oregon State University, Corvallis, OR, USA
e-mail: kagan.tumer@oregonstate.edu

1 Introduction

The efficient, safe and reliable management of the ever increasing air traffic is one of the fundamental challenges facing the aerospace industry today. On a typical day, more than 40,000 commercial flights operate within the US airspace [41], and the scheduling allows for very little room to accommodate deviations from the expected behavior of the system. As a consequence, the system is slow to respond to developing weather or airport conditions leading potentially minor local delays to cascade into large regional congestions. The Federal Aviation Administration (FAA) data shows that weather, routing decisions and airport conditions caused 1,682,700 h of delays in 2007 [18]. These delays cost a staggering $41 billion (in terms of airline operating costs and cost to passengers/businesses), and resulted in 740 million gallons of fuel being wasted [23].

Current air traffic management relies on a centralized, hierarchical routing strategy that performs flow projections ranging from 1 to 6 hour. Therefore, the system is not only slow to respond to changes, but is also at the limit of its capacity. As the traffic flow increases, the current procedures increase the load on the system, the airports, and the air traffic controllers (more aircraft per region) without providing any of them with means to shape the traffic patterns beyond minor reroutes. Indeed it is difficult to see how the current systems and algorithms can accommodate the expected threefold increase in air traffic, anticipated over the next two decades [15]. Unlike many other flow problems where the increasing traffic is to some extent absorbed by improved hardware (e.g., more servers with larger memories and faster CPUs for internet routing) the air traffic domain needs to find mainly algorithmic solutions, as the infrastructure (e.g., number of the airports) will not change significantly to impact the flow problem. There is therefore a strong need to explore new, distributed and adaptive solutions to the air flow control problem.

1.1 Motivating a multiagent approach

An adaptive, multiagent approach is an ideal fit to this naturally distributed problem where the complex interaction among the aircraft, airports and traffic controllers renders a predetermined centralized solution severely suboptimal at the first deviation from the expected plan. Though a truly distributed and adaptive solution (e.g., free flight where aircraft can choose almost any path) offers the most potential in terms of optimizing flow, it also provides the most radical departure from the current system. As a consequence, a shift to such a system presents tremendous difficulties both in terms of implementation (e.g., scheduling and airport capacity) and political fallout (e.g., impact on air traffic controllers). In this paper, we focus on agent based system that can be implemented readily. In this approach, we assign an agent to a "fix," a specific location in 2D. Because aircraft flight plans consist of a sequence of fixes, this representation allows localized fixes (or agents) to have direct impact on the flow of air traffic.

Controlling air traffic flow is a complex task that involves multiple controls, of various degree of granularity [46,47]. In this work we explore three specific types of actions for actions at fixes: (i) agents set the separation that aircraft that particular fix are required to keep; (ii) agents set ground delays that keep aircraft whose flight plan takes them through that fix on the ground; and (iii) agents set rerouting decisions, altering the routes of the aircraft whose flight plan takes them through that fix. The first action is the simplest one, and lets the agents to slow down or speed up local traffic which allows agents to a have significant impact on the overall air traffic flow. the second action lets the agents ground particular aircraft whose flight plan is most likely to cause delays as estimated by that agent. Finally, the third action is the most intricate and allows the agent to alter the routes of the aircraft, resulting in more direct congestion management. In all cases, the agents learn the most appropriate separation for their location using a reinforcement learning (RL) algorithm [44].

In a RL approach, the selection of the agent reward has a large impact on the performance of the system. In this work, we explore four different agent reward functions, and compare them to simulating various changes to the system and selecting the best solution (e.g, equivalent to a Monte–Carlo search). The first explored reward consisted of the system reward. The second reward was a difference reward which aimed to quantify the impact an agent has on the full system [3,49,53]. The last reward was a difference rewards based on estimates aimed to lower the computational burden. Both these difference rewards aim to align agent rewards with the system reward and ensure that the rewards remain sensitive to the agents' actions.

1.2 Related work

1.2.1 Traditional air traffic management approaches

The problem of air traffic flow management has been approached from numerous first-principles modeling perspectives. These methods tend to be divided into Lagrangian models where the dynamics of individual aircraft are taken into account and Eulerian (along with Aggregate) models where only the aggregate flow patterns are modeled. In both cases creating optimization from the resulting model is a complex process and numerous complexity reduction steps need to be taken.

Lagrangian models for air traffic flow management involve computing the trajectories of individual aircraft [6,29]. These models can be used for a range of problems, from collision avoidance to congestion reduction. Using simplified trajectory models, field based congestion reduction algorithms have been used to avoid separation assurance problems [16,25]. Similarly collision avoidance reduction has been approached using trajectory models and geometric based avoidance heuristics [8]. In addition Lagrangian models have been used to predict sector capacities allowing them to be put into the framework of Eulerian models. Key progress in this area has been in improved differential models of air traffic flow, improved hybrid state estimation and the creation of application specific polynomial time algorithms that can be used in lieu of integer programming [6,23,37,45].

Instead of predicting the path of individual aircraft, Eulerian and aggregate flow models predict flow patterns in the airspace [7,13,33]. While these models lose some of fine granularity of the Lagrangian approach, they lead to a simplification of the airspace allowing for more optimization techniques to be applied. Aggregating over sectors, it was found that linear programming could be used to find integral solutions in the optimization of airflow [7,28]. By modeling flow between control volumes in the airspace, linear algorithms can be used to control the airspace [30,33,41]. In addition, aggregate models have been developed to help analyze departure, enroute and arrival delays [32]. Also models have shown to be effective with piecewise linearization [42].

In addition to being applied to the current airspace model, traditional optimization methods have also been applied to "free flight" paradigms, where aircraft can improve efficiency by using custom routes [9,17,19,26,31,36,50]. Quasi-linearization has been shown to be effective in free flight problems with dense traffic conditions [31]. For creating wind-optimal free flight routes dynamic programming methods have shown to be effective [19]. Also rule based systems have been used to handle conflicts in free flight systems [17].

While these traditional approaches are well understood and effective in many scenarios, they are difficult to apply to situations where conditions change locally or automation works side-by-side with human controllers. First, such centralized algorithms are slow to incorporate changing conditions as they require recomputing a full simulation for each set of new parameters. Second, they provide a monolithic solution where it is difficult for air traffic controllers distributed throughout the air traffic system to provide data or receive suggestions as needed. Third, they are difficult to interpret by a human controller, and thus, it is exceedingly difficult for an air traffic controller to perform "sanity checks" to see if the proposed solution makes sense in the controller's local environment. Because they do not provide adaptive, decentralized and understandable solutions, one can understand why such methods have not gained the trust of both administrators and air traffic controllers. In contrast, an adaptive agent approach can potentially address all three issues and provide, adaptive, local solutions, both independent of the types of control used in other parts of the airspace, and readily analyzable by the local air traffic controller.

1.2.2 Agent based air traffic control

As agent methods are a natural tool to help automate existing air traffic systems, there has been significant research into other agent solutions as well [21,24,34,35]. These solutions typically involve a set of autonomous agents that try to optimize some overall goal either through learning or through negotiation (note these "learning agents" are not closely related to "agents" in fields that attempt to model or augment human behavior) [11,12]. Agent interactions, inspired by economic principles, have been shown to achieve fairness in air traffic management through an artificial monetary system that allows for retaliation against greedy agents [24]. In addition learning agents have also been used for air traffic systems. Though one key problem that needs to be addressed with learning agents is how to derive reward functions for each agent so that agents do not learn to hinder each other. In other contexts this has been addressed through a "satisficing" reward that specifically encourages cooperation and penalizes anti-cooperative behavior [21], and difference rewards where the actions of the agents aim to improve the system wide performance criteria [4,47,48]. An extension of that work also investigated the impact of agent suggestions on human decision making [5]. To date, the most complete path planning, control and collision avoidance based on agent technology is AgentFly which achieves reliable air traffic flow control without centralized planning [34,35,39,40].

Instead of independently maximizing a reward, agents can often form solutions to air traffic problems through negotiation and other explicit coordination mechanisms. To avoid conflicts in free flight, agent negotiation has been combined with utilities that incorporate overall risk [52]. Also, to coordinate runway slot times agent negotiation has been successfully applied without the need for centralized control or perfect situation knowledge [51]. Outside of negotiation, tokens have been used in distributed air traffic management to optimize paths while retaining safety [14].

Agent methods can also involve distributed control laws and distributed scheduling that seek to implicitly reduce traffic conflict. In the context of setting safe landing distances, agents operating on distributed control laws effectively reduce conflict [20]. In the context of free flight, agents using a combination of aircraft dynamical laws and theorem provers were also able to produce solutions to reduce conflict [27]. In addition a number of methods where agents model aircraft control laws has been successfully tried. In order to create compromise scheduling solutions involving airlines with different cost structures, models using partial differential equations have been combined with Nash Bargaining between airlines [38].

A control law approach has also been used in the avoidance control domain in systems that have been decomposed into combinations of local non-linear functions [43].

1.3 Contributions of this work

The main contribution of this paper is to present an adaptive air traffic flow management algorithm that both provides a coordinated multiagent solution, and can be readily implemented and tested using FACET. The work in this paper extends our earlier work on applying multiagent learning to air traffic control [4,47,48]. In particular, in addition to a more detailed discussion of agent selection and decision making, it contains a thorough mathematical derivation of agent rewards to ensure coordinated global behavior, the derivation of estimates for those agent rewards, increased action space for the agents, in depth scaling results, and results from historical data obtained from the Chicago and New York airspaces. We not only show that allowing agents multiple actions provide further improvements over our initial results [47], but that the results both scale well and extend to real world data.

In Sect. 2, we describe the air traffic flow problem, describe the FACET simulator and present the system evaluation function. In Sect. 3, we present the agent-based approach, focusing on the selection of the agents, their action space, their learning algorithms and their reward structure. In Sect. 4 we present results in a simulated domain with one and two congestions, and explore the impact of the different action spaces, the trade-offs between the various terms of the system evaluation function and the computational cost of achieving certain levels of performance. In Sect. 5 we present results from the Chicago and New York airspaces that show that the agent based approach performs well in a real world setting. Finally, in Sect. 6, we discuss the implications of these results and highlight future research directions, including how to include humans in the decision making process.

2 Air traffic flow management

With over 40,000 flights operating within the United States airspace on an average day, the management of traffic flow is a complex and demanding problem. Not only are there concerns for the efficiency of the system, but also for fairness (e.g., different airlines), adaptability (e.g., developing weather patterns), reliability and safety (e.g., airport management). In order to address such issues, the management of this traffic flow occurs over four hierarchical levels:

1. Separation assurance (2–30 min decisions);
2. Regional flow (20 min to 2 hour);
3. National flow (1–8 hour); and
4. Dynamic airspace configuration (6 hour to 1 year).

Because of the strict guidelines and safety concerns surrounding aircraft separation, we will not address that control level in this paper. Similarly, because of the business and political impact of dynamic airspace configuration, we will not address the outermost flow control level either. Instead, we will focus on the regional and national flow management problems, restricting our impact to decisions with time horizons between 20 min and 8 hour. The proposed algorithm will fit between long term planning by the FAA and the very short term decisions by air traffic controllers.

2.1 Airspace configuration

The continental US airspace consists of 20 regional centers (handling 200–300 flights on a given day) and 830 sectors (handling 10–40 flights). The flow control problem has to address the integration of policies across these sectors and centers, account for the complexity of the system (e.g., over 5,200 public use airports and 16,000 air traffic controllers) and handle changes to the policies caused by weather patterns. Two of the fundamental problems in addressing the flow problem are: (i) modeling and simulating such a large complex system as the fidelity required to provide reliable results is difficult to achieve; and (ii) establishing the method by which the flow management is evaluated, as directly minimizing the total delay may lead to inequities towards particular regions or commercial entities. Below, we discuss how we addressed both issues, namely, we present FACET a widely used simulation tool and discuss our system evaluation function.

2.2 FACET

FACET (Future ATM Concepts Evaluation Tool), a physics based model of the US airspace was developed to accurately model the complex air traffic flow problem [10]. It is based on propagating the trajectories of proposed flights forward in time. FACET can be used to either simulate and display air traffic (a 24 h slice with 60,000 flights takes 15 min to simulate on a 3 GHz, 1 GB RAM computer) or provide rapid statistics on recorded data (4D trajectories for 10,000 flights including sectors, airports, and fix statistics in 10 s on the same computer) [10,41].

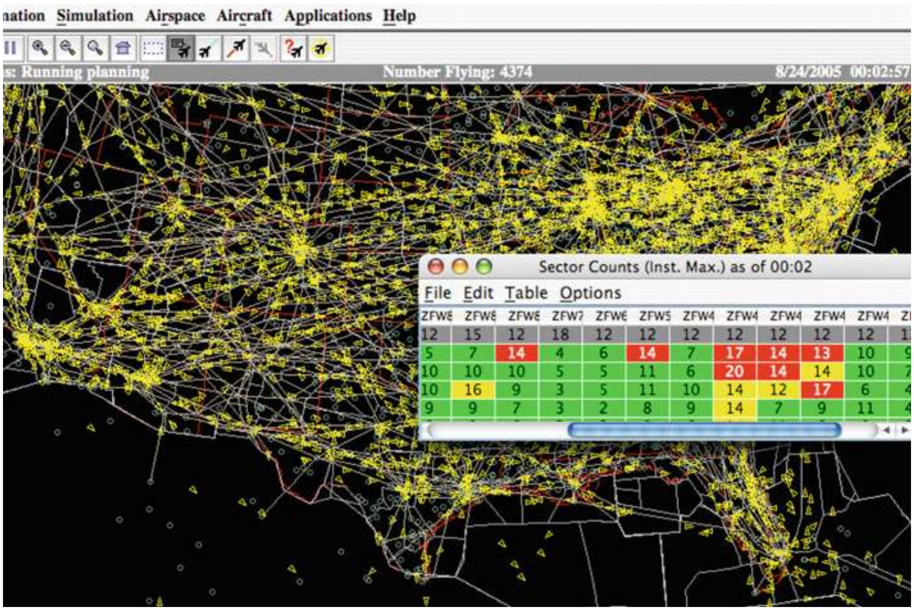
FACET simulates air traffic based on flight plans and through a graphical user interface allows the user to analyze congestion patterns of different sectors and centers (Fig. 1). FACET also allows the user to change the flow patterns of the aircraft through a number of mechanisms, including metering aircraft through fixes. The user can then observe the effects of these changes to congestion. In this paper, agents use FACET to computing the routes of aircraft after the agent applies a control action. The agents then produce their rewards based on received feedback from FACET about the impact of these controls.

2.3 System evaluation

The system performance evaluation function we selected focuses on delay and congestion but does not account for fairness impact on different commercial entities. Instead it focuses on the amount of congestion in a particular sector and on the amount of measured air traffic delay. The linear combination of these two terms gives the full system evaluation function, $G(z)$ as a function of the full system state z. More precisely, we have:

$$G(z) = -(B(z) + \alpha C(z)), \tag{1}$$

where $B(z)$ is the total delay penalty for all aircraft in the system, and $C(z)$ is the total congestion penalty. The relative importance of these two penalties is determined by the value of α, a congestion cost. For instance setting a to 5 (as it is for the all experiments in this paper, except those reported in Fig. 6 where we specifically investigate the impact of changing α) means that a sector over capacity by one aircraft for one time step has the same cost as 5 min of delay. With this interpretation the entire system evaluation, $G(z)$, can be seen as total cost in terms of minutes of delay equivalents.


nation Simulation Airspace Aircraft Applications Help
s: Running planning
Number Flying: 4374
8/24/2005 00:02:57
Sector Counts (Inst. Max.) as of 00:02
File Edit Table Options

ZFW8	ZFW8	ZFW8	ZFW7	ZFW6	ZFW5	ZFW4	ZFW4	ZFW4	ZFW4	ZFW4	Z
12	15	12	18	12	12	12	12	12	12	12	1
5	7	14	4	6	14	7	17	14	13	10	9
10	10	10	5	5	11	6	20	14	14	10	7
10	16	9	3	5	11	10	14	12	17	6	4
9	9	7	3	2	8	9	14	7	9	11	4

Fig. 1 FACET screenshot displaying traffic routes and air flow statistics

The total delay, B, is a sum of delays over the set of aircraft A and is given by:

$$B(z) = \sum_{a \in A} B_a(z) \tag{2}$$

where $B_a(z)$ is the delay of each aircraft caused the agents' controls. For controls that delay aircraft (discussed in Sects. 4.1 and 4.2), the $B_a(z)$ is simply the amount of delay applied to that aircraft. For controls involving rerouting (Sect. 4.3), the delay is the amount of additional time it takes an aircraft to go on its new route instead of its schedule route.

The total congestion penalty is a sum over the congestion penalties over the sectors of observation, S:

$$C(z) = \sum_{s \in S} C_s(z) \tag{3}$$

where

$$C_s(z) = \sum_t \Theta(k_{s,t} - c_s)(k_{s,t} - c_s)^2, \tag{4}$$

where c_s is the capacity of sector s as defined by the FAA and $\Theta(\cdot)$ is the step function that equals 1 when its argument is greater or equal to zero, and has a value of zero otherwise. Intuitively, $C_s(z)$ penalizes a system state where the number of aircraft in a sector exceeds the FAAs official sector capacity. Each sector capacity is computed using various metrics which include the number of air traffic controllers available. The quadratic penalty is intended to provide strong feedback to return the number of aircraft in a sector to below the FAA mandated capacities.

3 Agent based air traffic flow

The multi agent approach to air traffic flow management we present is predicated on adaptive agents taking independent actions that maximize the system evaluation function discussed above. In this model, each agent first chooses an action. The results of all the agents' actions are then simulated in FACET. From the simulation all congestion and lateness values are observed, a system reward is computed and agents compute their appropriate rewards. While agents may use the system evaluation as their reward there are other possibilities too, discussed in Sect. 3. These rewards are then used to modify the agents' control policies, which are then used to choose the next action. For the agent-based air traffic management system to succeed, we need to make four critical decisions that need to be made: defining the agents, defining the agents' action space, selecting the agents' learning algorithms, and selecting the agents' reward structure, and we discuss each step below.

3.1 Defining the agents

Selecting the aircraft as agents is perhaps the most obvious choice for defining an agent. That selection has the advantage that agent actions can be intuitive (e.g., change of flight plan, increase or decrease speed and altitude) and offer a high level of granularity, in that each agent can have its own policy. However, there are several problems with that approach. First, there are in excess of 40,000 aircraft in a given day, leading to a massively large multiagent system. Second, as the agents would not be able to sample their state space sufficiently, learning would be prohibitively slow.

As an alternative, we assign agents to individual ground locations throughout the airspace called "fixes". Each agent is then responsible for any aircraft going through its fix. Fixes offer many advantages as agents:

1. Their number can vary depending on need. The system can have as many agents as required for a given situation(e.g., agents coming "live" around an area with developing weather conditions).
2. Because fixes are stationary, collecting data and matching behavior to reward is easier.
3. Because Aircraft flight plans consist of fixes, agent will have the ability to affect traffic flow patterns.
4. They can be deployed within the current air traffic routing procedures, and can be used as tools to help air traffic controllers rather than compete with or replace them.

Figure 2 shows a schematic of this agent based system. Agents surrounding a congestion or weather condition affect the flow of traffic to reduce the burden on particular regions. Notice in our formulation, a single agent controls one flow into a congestion. While arrangement decouples agents somewhat in terms of their actions, they are still strongly coupled in terms of their performance: how an agent best controls one flow, is dependent on how all the other agents are controlling their flows.

3.2 Defining agent actions

The second issue that needs to be addressed, is determining the action set of the agents. Again, an obvious choice may be for fixes to "bid" on aircraft, affecting their flight plans. Though appealing from a free flight perspective, that approach makes the flight plans too unreliable and significantly complicates the scheduling problem (e.g., arrival at airports and the subsequent gate assignment process).

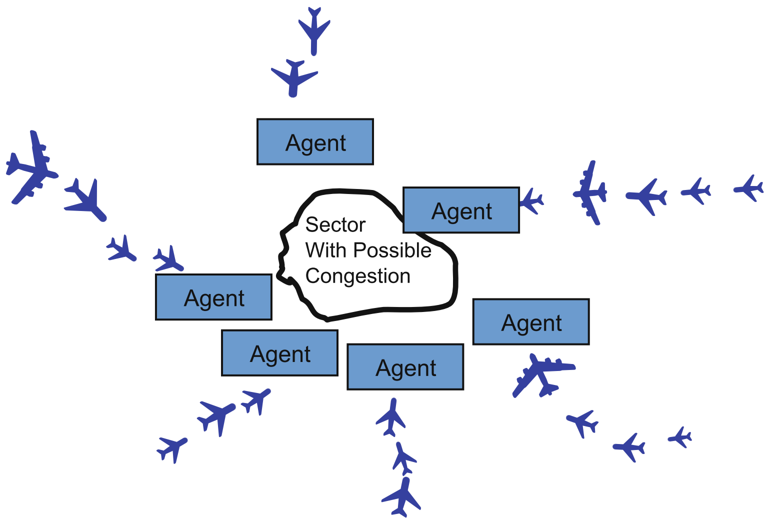


Fig. 2 Schematic of agent architecture. The agents corresponding to fixes surrounding a possible congestion become "live" and start setting new separation times

Instead, we explore three methods for the agent based fixes to control the flow. Allowing agents to have the flexibility to control aircraft in multiple ways is essential to their ability to be integrated into existing systems. Even if all the methods work relatively well, an organization or a sector controller may only be comfortable with a particular form of flow control. Agents that are not flexible enough to conform to these needs will not be used. The methods used in this paper are as follows:

1. *Miles in trail (MIT)* Agents control the distance aircraft have to keep from each other wile approaching a fix. With a higher MIT value, fewer aircraft will be able to go through a particular fix during congested periods, because aircraft will be slowing down to keep their spacing. Therefore setting high MIT values can be used to reduce congestion downstream of a fix.
2. *Ground delays* An agent controls how long aircraft that will eventually go through a fix should wait on the ground. Imposing a ground delay will cause aircraft to arrive at a fix later. With this action, congestion can be reduced if some agents choose ground delays and others do not, as this will spread out the congestion. However, note that if all the agents choose the same ground delay, then the congestion will simply happen at a later moment in time.
3. *Rerouting* An agent controls the routes of aircraft going through its fix, by diverting them to take other routes that will (in principle) avoid the congestion.

Note that these methods also differ in the cost to implement them. For instance delaying an aircraft on the ground may be far preferable to delaying it in the air. The relative cost/benefit will depend on the circumstances of the airspace and the airlines, so it is important to have several options available.

3.3 Selecting agent learning algorithms

The objective of each agent is to select the action that leads to the best system performance, G (given in Eq. 1). Each agent will have its own reward function and will aim to maximize that reward using a RL algorithm [44] (though alternatives such as evolving neuro-controllers are also effective [1]). For delayed-reward problems, sophisticated RL systems such as temporal difference may have to be used. However, due to our agent selection and agent action set, the air traffic congestion domain modeled in this paper only needs to utilize immediate rewards. As a consequence, a simple table-based immediate reward reinforcement learner is used. Our reinforcement learner is equivalent to an ϵ-greedy action-value learner [44]. At every episode an agent takes an action and then receives a reward evaluating that action. After taking action a and receiving reward R an agent updates its value for action a, $V(a)$ (which is its estimate of the value for taking that action [44]) as follows:

$$V(a) \leftarrow (1-\lambda)V(a) + (\lambda)R, \tag{5}$$

where λ is the learning rate. At every time step, the agent chooses the action with the highest table value with probability $1-\epsilon$ and chooses a random action with probability ϵ. In the experiments described in this paper, λ is equal to 0.5 and ϵ is equal to 0.25. The parameters were chosen experimentally, though system performance was not overly sensitive to these parameters.

3.4 Selecting agent reward structure

The final issue that needs to be addressed is selecting the reward structure for the learning agents. The first and most direct approach is to let each agent receive the system performance as its reward. However, in many domains such a reward structure leads to slow learning. We will therefore also set up a second set of reward structures based on agent-specific rewards. Given that agents aim to maximize their own rewards, a critical task is to create "good" agent rewards, or rewards that when pursued by the agents lead to good overall system performance. In this work we focus on "difference rewards" which aim to provide a reward that is both sensitive to that agent's actions and aligned with the overall system reward [2,49,53].

3.4.1 Difference rewards

Consider *difference* rewards of the form [2,49,53]:

$$D_i \equiv G(z) - G(z - z_i + c_i), \tag{6}$$

where z_i is the action of agent i. All the components of z that are affected by agent i are replaced with the fixed constant c_i.[1]

In many situations it is possible to use a c_i that is equivalent to taking agent i out of the system. Intuitively this causes the second term of the difference reward to evaluate the performance of the system without i and therefore D evaluates the agent's contribution to the system performance. There are two advantages to using D: First, because the second term removes a significant portion of the impact of other agents in the system, it provides an agent with a "cleaner" signal than G. This benefit has been dubbed "learnability" (agents have an easier time learning) in previous work [2,49]. Second, because the second term does not depend on the actions of agent i, any action by agent i that improves D, also improves G. This term which measures the amount of alignment between two rewards has been dubbed "factoredness" in previous work [2,49].

[1] This notation uses zero padding and vector addition rather than concatenation to form full state vectors from partial state vectors. The vector "z_i" in our notation would be $z_i e_i$ in standard vector notation, where e_i is a vector with a value of 1 in the ith component and is zero everywhere else.

3.4.2 Difference reward estimate

Though providing a good compromise between aiming for system performance and removing the impact of other agents from an agent's reward, one issue that may plague D is computational cost. Because it relies on the computation of the counterfactual term $G(z - z_i + c_i)$ (i.e., the system performance without agent i) it may be difficult or impossible to compute, particularly when the exact mathematical form of G is not known. Let us focus on G functions in the following form:

$$G(z) = G_f(f(z)), \tag{7}$$

where $G_f(\,)$ is non-linear with a known functional form and,

$$f(z) = \sum_i f_i(z_i), \tag{8}$$

where each f_i is an unknown non-linear function. We assume that we can sample values from $f(z)$, enabling us to compute G, but that we cannot sample from each $f_i(z_i)$. In addition, we assume that G_f is much easier to compute than $f(z)$, or that we may not be able to even compute $f(z)$ directly and must sample it from a "black box" computation. This form of G matches our system evaluation in the air traffic domain. When we arrange agents so that each aircraft is typically only affected by a single agent, each agent's impact of the counts of the number of aircraft in a sector, $k_{t,s}$, will be mostly independent of the other agents. These values of $k_{t,s}$ are the "$f(z)$s" in our formulation and the penalty functions form "G_f." Note that given aircraft counts, the penalty functions (G_f) can be easily computed in microseconds, while aircraft counts (f) can only be computed by running FACET taking on the order of seconds.

To compute our counterfactual $G(z - z_i + c_i)$ we need to compute:

$$\begin{align} G_f(f(z - z_i + c_i)) &= G_f\left(\sum_{j \neq i} f_j(z_j) + f_i(c_i)\right) \tag{9}\\ &= G_f\,(f(z) - f_i(z_i) + f_i(c_i)). \tag{10}\end{align}$$

Unfortunately, we cannot compute this directly as the values of $f_i(z_i)$ are unknown. However, if agents take actions independently (it does not observe how other agents act before taking its own action) we can take advantage of the linear form of $f(z)$ in the f_is with the following equality:

$$E(f_{-i}(z_{-i})|z_i) = E(f_{-i}(z_{-i})|c_i) \tag{11}$$

where $E(f_{-i}(z_{-i})|z_i)$ is the expected value of all of the fs other than f_i given the value of z_i and $E(f_{-i}(z_{-i})|c_i)$ is the expected value of all of the fs other than f_i given the value of z_i is changed to c_i. We can then estimate $f(z - z_i + c_i)$:

$$\begin{align*} f(z) - f_i(z_i) + f_i(c_i) &= f(z) - f_i(z_i) + f_i(c_i)\\ &\quad + E(f_{-i}(z_{-i})|c_i) - E(f_{-i}(z_{-i})|z_i)\\ &= f(z) - E(f_i(z_i)|z_i) + E(f_i(c_i)|c_i)\\ &\quad + E(f_{-i}(z_{-i})|c_i) - E(f_{-i}(z_{-i})|z_i)\\ &= f(z) - E(f(z)|z_i) + E(f(z)|c_i). \end{align*}$$

Therefore we can evaluate $D_i = G(z) - G(z - z_i + c_i)$ as:

$$D_i^{\text{expected}} = G_f(f(z)) - G_f(f(z) - E(f(z)|z_i) + E(f(z)|c_i)), \tag{12}$$

leaving us with the task of estimating the values of $E(f(z)|z_i)$ and $E(f(z)|c_i)$. These estimates can be computed by keeping a table of averages where we average the values of the observed $f(z)$ for each value of z_i that we have seen. This estimate should improve as the number of samples increases.

To improve our estimates, we can set $c_i = E(z)$ and if we make the mean squared approximation of $f(E(z)) \approx E(f(z))$ then we can estimate $G(z) - G(z - z_i + c_i)$ as:

$$D_i^{\text{est}} = G_f(f(z)) - G_f(f(z) - E(f(z)|z_i) + E(f(z))). \tag{13}$$

This formulation has the advantage in that we have more samples at our disposal to estimate $E(f(z))$ than we do to estimate $E(f(z)|c_i)$.

3.5 Pre-computed difference rewards

In addition to the estimates to the difference rewards, it is possible to pre-compute certain values that can be later used by the agents. In particular, $-E(f(z)|z_i) + E(f(z)|c_i)$ can be computed exactly if the function f can be interrogated for certain values of z:

$$\begin{aligned}
&-E(f(z)|z_i) + E(f(z)|c_i)\\
&= -E(f_{-i}(z_{-i})) - E(f_i(z_i)|z_i)\\
&\quad + E(f_{-i}(z_{-i})) + E(f_i(z_i)|c_i)\\
&= -E(f_i(z_i)|z_i) + E(f_i(z_i)|c_i)\\
&= -f_i(z_i) + f_i(c_i)\\
&= -f_{-i}(k_{-i}) - f_i(z_i) + f_{-i}(k_{-i}) + f_i(c_i)\\
&= -f(k_{-i} + z_i) + f(k_{-i} + c_i),
\end{aligned}$$

where k_{-i} is a constant set of actions for all the agents other than i and f_{-i} is equal to $\sum_{j \neq i} f_j(z_j)$. Then by precomputing $f(k_{-i} + z_i)$ for each action for each agent, the value of D can be computed exactly. With n agents and m possible actions this requires

$$N_{\text{pre}} = 1 + m \times n \tag{14}$$

computations of f.

4 Experimental results with simulated air traffic

In this section we test the performance of our agent based air traffic optimization method on a series of simulations using the FACET air traffic simulator. In all experiments we test the performance of four different methods. The first method is Monte Carlo estimation, where random policies are created, with the best policy being chosen. The other three methods are agent based methods where the agents are maximizing one of the following rewards:

1. The system reward, $G(z)$, as define in Eq. 1.
2. The difference reward $D_i(z)$, assuming that agents can calculate counterfactuals.
3. Estimation to the difference reward $D_i^{\text{est}}(z)$, where agents estimate the counterfactual using $E(f(z)|z_i)$ and $E(f(z))$.

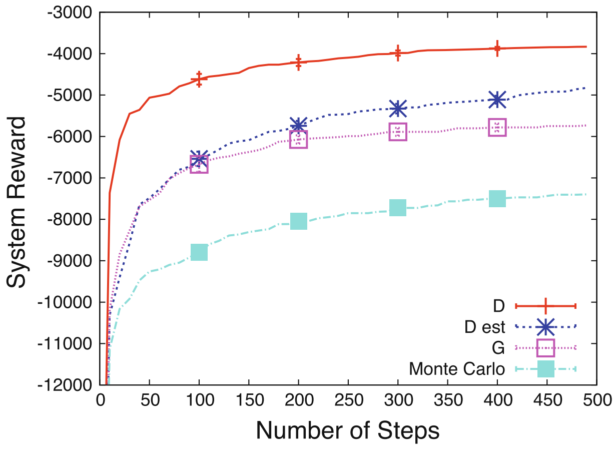


Fig. 3 Performance of agents controlling MIT on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

We test the performance of these reward for all three action spaces. In all cases, we investigate two types of congestion. The first one consists of an air traffic scenario using artificial data with there are 1,000 aircraft. Here 600 of the aircraft are going through an area of high congestion, while 400 aircraft are going through an area of moderate congestion. Our second scenario consists of real-world historical data in the Chicago and New York areas. All experiments are for an eight our window of traffic.

In all experiments the goal of the system is to maximize the system performance given by $G(z)$ with the parameters, $\alpha = 5$. In all experiments to make the agent results comparable to the Monte Carlo estimation, the best policies chosen by the agents are used in the results. All results are an average of 30 independent trials with the differences in the mean $(\sigma/\sqrt{n})$ shown as error bars, though in most cases the error bars are too small to see.

4.1 Controlling miles in trail

In our first set of experiments, agents control MIT: the distance aircraft have to separate themselves from each other when approaching a fix. Here agents choose between the 11 actions of setting the MIT value ranging from 0 to 100 miles in increments of 10 miles. Setting the MIT to 0 produces no effect, while setting it to high values forces the aircraft to slow down to keep their separation distance. Therefore setting high MIT values upstream of a congestion can alleviate a congestion, at the cost the increased delay.

In the first experiment we test the performance of the four methods on the artificial data using two points of congestion, with 20 agents. The first point of congestion is created by setting up a series of flight plans that cause the number of aircraft in the sector of interest to be significantly more than the number allowed by the FAA. The second congestion is placed in a different part of the US airspace and is less severe than the first one, so agents have to form different policies depending which point of congestion they are influencing. The results displayed in Fig. 3 show the performance of all four algorithms. These results show that the agent based methods significantly outperform the Monte Carlo method. This result is not surprising since the agent based methods intelligently explore their space, where as the Monte Carlo method explores the space randomly.

Among the agent based methods, agents using difference rewards perform very well and considerable better than agents using the system reward. They were able to achieve a result with a performance penalty of about 4,000. Recall this is measured in lateness in minutes plus five times congestion. We can view this combination as a "lateness equivalent." Therefore this solution achieves a "lateness equivalent" of about 4 min per-aircraft. This result is pretty good and is within 2% of the best solution obtained after 20,000 trials using difference rewards and competing methods. Also in our analysis of good solutions, the tradeoffs made between lateness and congestion varied. Though in many of these solutions they were fairly equal after weighting: about 2,000 min of lateness with the rest of the penalty coming from congestion.

Having agents using the difference reward perform better is not surprising, since with 20 agents, an agent directly trying to maximize the system reward has difficulty determining the effect of its actions on its own reward. Even if an agent takes an action that reduces congestion and lateness, other agents at the same time may take actions that increase congestion and lateness, causing the agent to wrongly believe that its action was poor. In contrast agents using the difference reward have more influence over the value of their own reward, therefore when an agent takes a good action, the value of this action is more likely to be reflected in its reward. This experiment also shows that estimating the difference reward is not only possible, but also quite effective, when the true value of the difference reward cannot be computed. While agents using the estimate do not achieve as high of results as agents using the true difference reward, they still perform significantly better than agents using the system reward. Note, however, that the benefit of the estimated difference rewards are only present later in learning. Earlier in learning, the estimates are poor, and agents using the estimated difference reward perform no better then agents using the system reward.

To verify that the performance improvement of our methods is maintained when there are a different number of agents, we perform additional experiments with 50 agents. In these experiments we also increased the number of aircraft to 2,500 so that the number of aircraft for each agent remained constant, allowing us to use the same MIT values for the agents' actions. The results displayed in Fig. 4 show that when there are more agents, the agents using the difference reward perform even better. Figure 5 shows scaling results and demonstrates that the conclusions hold over a wide range of number of agents. Note that as the number of agents increases, the performance of agents using the difference reward remain relatively stable, while the performance of agents using other rewards goes down significantly. This can be explained by the agents having increasing difficulty separating the impact of their actions from the impact of the actions of all the other agents on their reward. However, here the difference reward is cleanly separating out an agent's impact on its reward from the influence of all the other agents, allowing performance to remain steady as the number of agents increases.[2]

4.1.1 Penalty tradeoffs

The system evaluation function used in the experiments is $G(z) = -(B(z) + \alpha C(z))$, which comprises of penalties for both congestion and lateness. This evaluation function forces the agents to tradeoff these relative penalties depending on the value of α. With high α the optimization focuses on reducing congestion, while with low α the system focuses on reducing

[2] This agent-scaling results here are significantly different than those reported in [47] on similar data. The main differences here are that the number of aircraft is scaled with the number of agents, and that the system reward function is different, significantly changes the results.

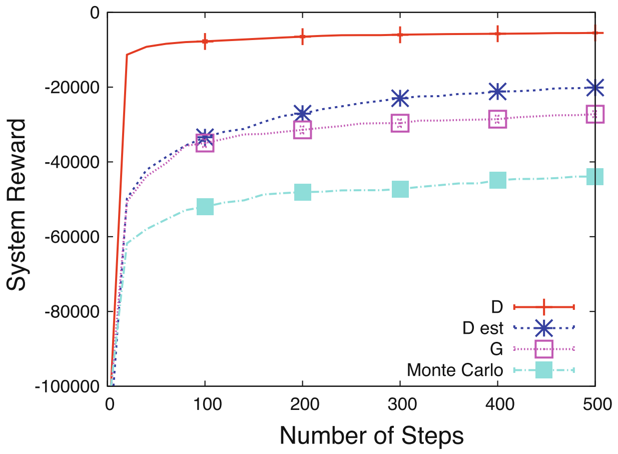


Fig. 4 Performance of agents controlling MIT on the dual congestion problem, with 2,500 aircraft, 50 agents and $\alpha = 5$

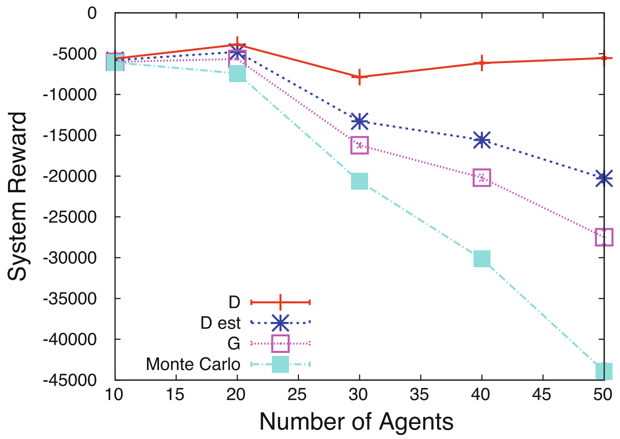


Fig. 5 Scaling properties of agents controlling MIT on the dual congestion problem, with $\alpha = 5$. Number of aircraft is proportional to number of agents ranging from 500 to 2,500 aircraft (Note sector capacities have been scaled according to number of aircraft also)

lateness. In this experiment we explore the impact of α on the system performance. Figure 6 shows the results of varying α for controlling MIT. At $\alpha = 0$, the only goal of the agents is to minimize lateness, allowing for the zero penalty solution of setting all MIT values to zero. Agents using difference rewards find this solution, while most other agents find low-cost solutions. Note however that agents using Monte Carlo have trouble optimizing even this trivial problem. As α get larger, the problem becomes more difficult as agents have to both minimize lateness and congestion. As α increases the curves tend to flatten out as the agents are successful in creating solutions designed primarily to reduce congestion. Note that under all values of α agents using difference rewards or estimates of difference rewards perform better than agents using Monte Carlo or agents directly maximizing the system reward.

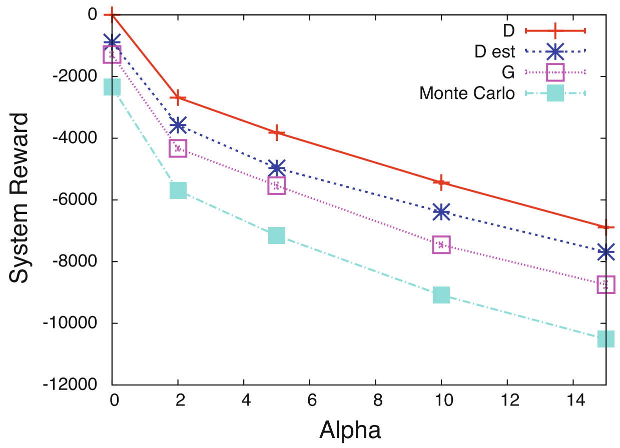


Fig. 6 Tradeoff between delay and congestion on the dual congestion problem, with 1,000 aircraft and 20 agents where agents control MIT. When only optimizing for lateness, a zero penalty solution is possible. The problem becomes for difficult when both lateness and congestion need to be optimized at once

4.2 Controlling ground delays

In the second set of experiments, agents control aircraft through ground delays. Here an agent can order aircraft that are scheduled to go through its fix to be delayed on the ground before they can takeoff. In this scenario agents choose between one of 11 actions ranging from no delay to 50 min of delay in increments of 5 min. Note that the dynamics of ground delays are quite different than with MITs since if all the agents choose the same ground delay, the congestion will still happen, just at a later time. Instead agents have to form the correct pattern of ground delays.

The results show (Fig. 7) that the different rewards' performance is qualitatively similar to the case where agents control MITs. Agents using the difference reward perform the best, while agents using the estimated difference reward also performed well. Note however, that agents using G or Monte Carlo estimation perform particularly poorly in this problem. This can be attributed to the problem being more difficult, since the action-reward mapping is more dependent on the actions of other agents. In essence, there is more "noise" in this system, and agent rewards that do not deal well with noise perform poorly.

4.3 Controlling reroutes

In this experiment agents alleviate congestions by rerouting aircraft around congestions. Here an agent's action is to set the probability that it will reroute an aircraft that goes through it's associated fix. In this experiment agents choose between one of 11 probabilities ranging from 0 to 100% in increments of 10%. The results show (Fig. 8) that again agents using D or D^{est} perform significantly better than agents using G or Monte Carlo estimation. Also note that the reward values here are much higher for all reward methods for agents controlling rerouting than for agents controlling MITs or ground delays. The reason for this is that rerouting aircraft around congested areas is highly effective as it completely removes the aircraft from the congested area instead of just delaying it. However, the reward values are not necessary comparable since delays incurred for rerouting aircraft represent having the aircraft going

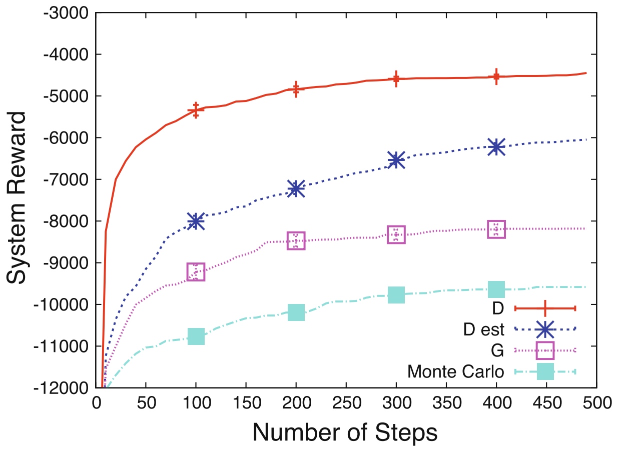


Fig. 7 Performance of agents controlling ground delays on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

longer distances and spend more time in the air, which could be significantly more costly than holding them on the ground.

4.4 Computational cost

The results in the previous sections show the performance of the different algorithms after a specific number of episodes. Those results show that D is significantly superior to the other algorithms. One question that arises, though, is what computational overhead D puts on the system, and what results would be obtained if the additional computational expense of D is made available to the other algorithms.

The computation cost of the system evaluation, G (Eq. 1) is almost entirely dependent on the computation of the airplane counts for the sectors $k_{t,s}$, which need to be computed using FACET. Except when D is used, the values of k are computed once per episode. However, to compute the counterfactual term in D, if FACET is treated as a "black box", each agent would have to compute their own values of k for their counterfactual resulting in $n + 1$ computation of k per episode. While it may be possible to streamline the computation of D with some knowledge of the internals of FACET, given the complexity of the FACET simulation, it is not unreasonable in this case to treat it as a black box.

Table 1 shows the performance of the agents controlling MIT using different reward structures after 2,100 G computations for each of the algorithms for the simulations presented in Fig. 3. These results are based on 20 agents and two congestions with $\alpha = 5$. All the algorithms except the fully computed D reach 2,100 k computations at time step 2,100. D however computes k once for the system, and then once for each agent, leading to 21 computations per time step. It therefore reaches 2,100 computations at time step 100. We also show the results of the full D computation at $t = 2{,}100$, which needs 44,100 computations of k as $D^{44\mathrm{K}}$. Although $D^{44\mathrm{K}}$ provides the best result by a slight margin, it is achieved at a considerable computational cost. Indeed, the performance of the D estimate is remarkable in this case, as it was obtained with about 20 times fewer computations of k. Furthermore, the D estimate significantly outperformed the full D computation for a given number of computations of k and validates the assumptions made in Sect. 3.4.2. This shows that for this domain, in practice

Table 1 Performance of 20 agents, dual congestion and $\alpha = 5$ (for agents controlling MIT), after 2,100 G evaluations (except for D^{44K} which has 44,100 G evaluations at $t = 2{,}100$)

Reward	G	$\sigma/\sqrt{n}$	Time
D^{est}	−4282	36	2,100
D	−4716	126	100
D^{44K}	−3757	1	2,100
G	−5109	51	2,100
MC	−6492	79	2,100

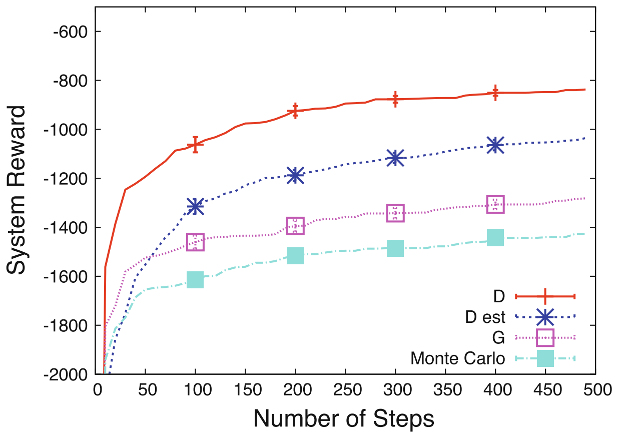


Fig. 8 Performance of agents controlling reroutes on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

it is more fruitful to perform more learning steps and approximate D, than few learning steps with full D computation when we treat FACET as a black box (Fig. 8).

4.5 Increased congestion

In our next experiment we simulate a situation where the amount of congestion present is increased by 30%. This experiment models one of the goals of the next generation airtraffic system initiative to increase the level of airtraffic with minimal increase of infrastructure. The results (Fig. 9) show that the relative performance of agents using different reward functions is similar for this higher level of congestion. Again agents using G and Monte Carlo estimation perform poorly. Also the learning system maximizing the difference reward performs significantly better than the other methods. This trend is emphasized in Fig. 10 where we vary the amount of congestion from 60% of the original traffic to 130%. Note that as congestion increases the problem becomes much harder. However, the absolute difference between the performance of the agents using different rewards tends to remain relatively constant at different levels of congestion. This can be explained by there being less degrees of freedom at high levels of congestion—all agents need to force a high level of delay. At moderate levels of congestion there is much more opportunity for optimization and agents using the difference reward can take advantage of this.

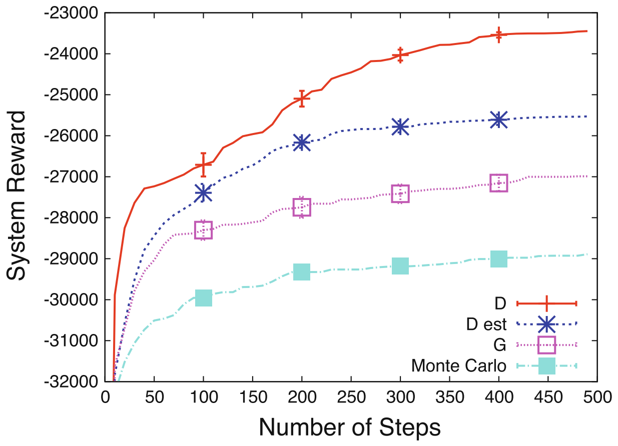


Fig. 9 Impact of 130% congestion on agents controlling MIT on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

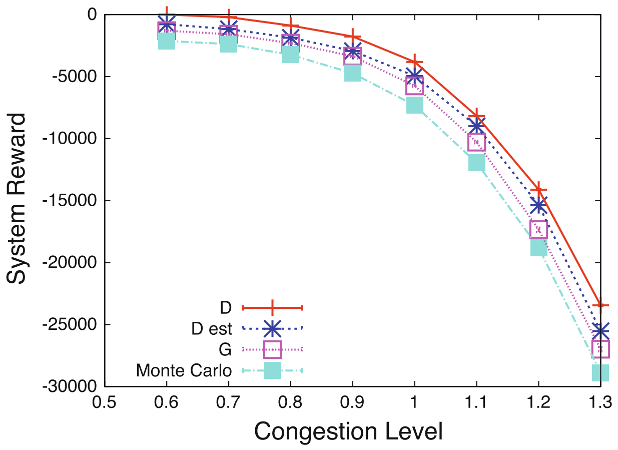


Fig. 10 Sensitivity of system performance to congestion. Agents control MIT on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

4.6 Rule-based agents

In the experiments so far, all the agent algorithms were based on RL, where agents gradually try to maximize an objective function. In this section we contrast this form of agent, to a rule-based agent that never even observes the objective function. Instead the rule-based agents are hand-coded with prior knowledge about a desirable state of the system. Using a closed-loop feedback system, the rule-based agents observe lateness and congestion directly and through a pre-programmed table try to modify the state of the system to try to send these properties back to desirable levels.

Using prior knowledge of the system, levels of congestion and lateness are broken down into high, medium and low values. Depending on the severity of the congestion and lateness measured in the system, the rule-base agents can increase or decrease the MIT values as summarized in Table 2. For instance if there is high congestion and low lateness, each agent

Table 2 Rules for rule-based agent

	Low	Medium	High
Low	0	+10	+20
Medium	−10	*	+10
High	−20	−10	*

Up-down is levels of lateness. Left-right is levels of congestion. Depending on the levels of lateness and congestion, MIT values are increased, decreased, or held constant. Entries with "*" are either +10, −10 or randomly set to ±10 depending on the algorithm

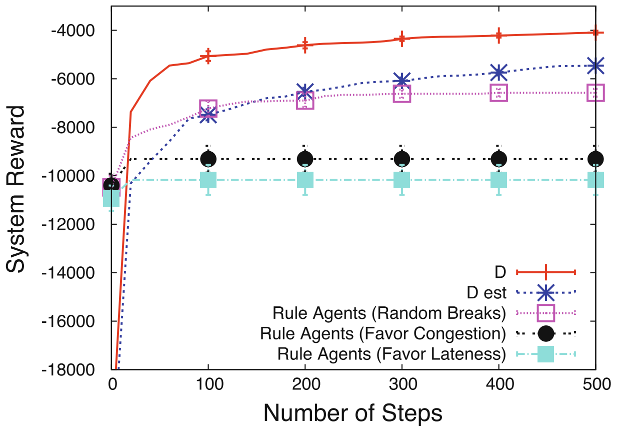


Fig. 11 Comparison of rule-based agents vs. learning agents controlling MIT on the dual congestion problem, with 1,000 aircraft, 20 agents and $\alpha = 5$

will increase its MIT value by 20. In this process the agents start with random MIT values, but through there rules, they attempt to find metering values that lead to acceptable levels of congestion and lateness. The hardest decision for a rule-based agents, comes when both congestion and lateness are high or medium. In these situation we have three experiments: (1) agents increase MIT values (favoring more lateness), (2) agents decrease MIT values (favoring more congestion), (3) agents randomly break the tie by either increase MIT by 10 or decrease it by 10.

Figure 11 shows the performance level of the rule-based agents. It is clear the rule-based agents with no-randomness, perform very poorly. They are unable to fine-tune the situation where both congestion and lateness are at similar levels, as there is no clear solution to this problem. Instead rule-based agents that are able to "break the tie" in this situation, using a randomly generated choice, perform better. In fact these rule-based agents are initially able to perform better than learning agents using D^{est}. However, all the learning agents are eventually able to perform better, and learning agents using D perform much better.

5 Experimental results with real air traffic

In this section we test the performance of the multiagent air traffic flow management algorithms on historical data from the Chicago and New York areas. For experiments in the

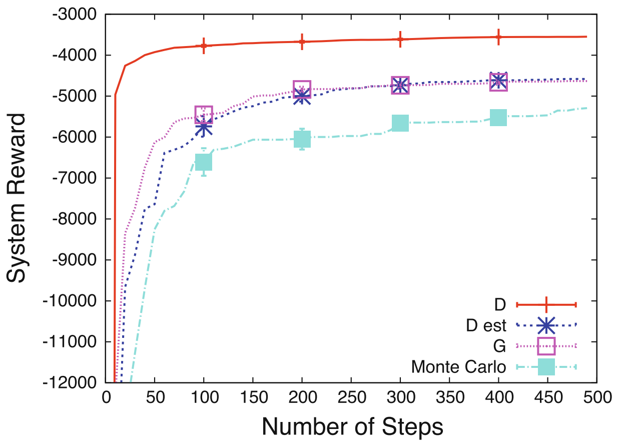


Fig. 12 Performance of agents controlling MIT on historical data from Chicago area. Agents control 1,297 aircraft

Chicago area, 20 agents are responsible for 20 fixes commonly used for aircraft arriving at Chicago OHare. Experiments in the New York area are similar, using 20 fixes commonly used for aircraft arriving at JFK, Newark and LaGuardia. Each agent is responsible for setting MIT values for aircraft that are schedule to go through its fix ranging from 0 to 10 miles. As with the previous results on artificial data, the agents generate rewards representing the impact of their choice of MIT using the FACET simulator. However, to speed up simulation, only the aircraft going through the 20 fixes are simulated. Sector counts generated from these simulations are combined with sector counts of a previous simulation using all air traffic for the entire U.S. airspace.

5.1 Chicago air traffic

We perform experiments with Chicago area data on a day where congestion peaks at certain times, and is moderate at other times. The results (Fig. 12) are similar to those with artificial data. They show that the learning method directly maximizing the system reward and Monte Carlo estimation still perform poorly compared to agents using the difference reward. Agents using D learn quickly and again reach the highest level of performance. Note however in this experiment, agents using D^{est} to not perform any better than agents using G.

5.2 New York air traffic

Along with Chicago area data, we perform experiments using New York area data. The congestion scenario in this data is significantly different than that in the Chicago area, in that congestion is heavy throughout the day. This scenario is more difficult, since in many situations slowing down aircraft to avoid a current congestion just adds to aircraft in a future congestion. The results show (Fig. 13) that agents using D were able to overcome this challenge and significantly outperform agents using the other rewards. In addition agents using the estimate to D were able to perform better than agents using G.

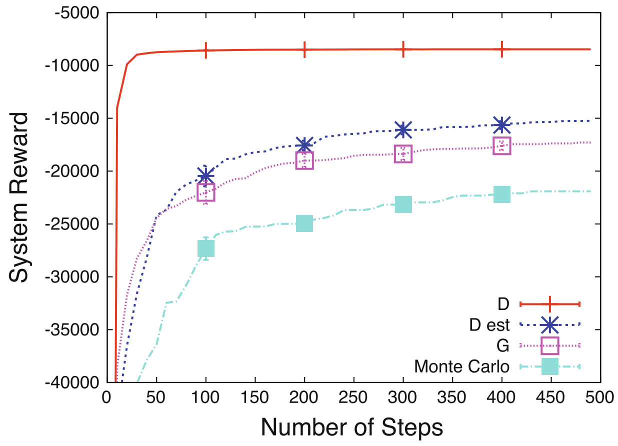


Fig. 13 Performance of agents controlling MIT on historical data from New York area. Agents control 1,577 aircraft

6 Summary

The efficient, safe and reliable management of air traffic flow is a complex problem, requiring solutions that integrate control policies with time horizons ranging from minutes up to a year. The main contribution of this paper is to present a distributed adaptive air traffic flow management algorithm that can be readily implemented and to test that algorithm using FACET, a simulation tool widely used by the FAA, NASA and the industry. Our method is based on agents representing fixes and having each agent determine the separation between aircraft approaching its fix. It offers the significant benefit of not requiring radical changes to the current air flow management structure and is therefore readily deployable. The agents use RL to learn control policies and we explore different agent reward functions and different ways of estimating those functions.

We are currently extending this work in three directions. First, we are exploring new methods of estimating agent rewards, to further speed up the simulations. Second we are investigating deployment strategies and looking for modifications that would have larger impact. One such modification is to extend the definition of agents from fixes to sectors, giving agents more opportunity to control the traffic flow, and allow them to be more efficient in eliminating congestion. Finally, in cooperation with domain experts, we are investigating different system evaluation functions, above and beyond the delay and congestion dependent G presented in this paper.

Acknowledgments The authors thank Banavar Sridhar and Shon Grabbe for their help with FACET. This research was partially supported by NSF grant CNS-0931591.

References

1. Agogino, A., & Tumer, K. (2004). Efficient evaluation functions for multi-rover systems. In *The genetic and evolutionary computation conference*. Seatle, WA, pp. 1–12.
2. Agogino, A., & Tumer, K. (2005). Multi agent reward analysis for learning in noisy domains. In *Proceedings of the fourth international joint conference on autonomous agents and multi-agent systems*. Utrecht, Netherlands.

3. Agogino, A. K., & Tumer, K. (2006). Handling communication restrictions and team formation in congestion games. *Journal of Autonomous Agents and Multi Agent Systems, 13*(1), 97–115.
4. Agogino, A., & Tumer, K. (2008). Regulating air traffic flow with coupled agents. In *Proceedings of the seventh international joint conference on autonomous agents and multi-agent systems*. Estoril, Portugal.
5. Agogino, A. K., & Tumer, K. (2009). Learning indirect actions in complex domains: Action suggestions for air traffic control. *Advances in Complex Systems, 12*, 493–512.
6. Bayen, A. M., Grieder, P., Meyer, G., & Tomlin, C. J. (2005). Lagrangian delay predictive model for sector-based air traffic flow. *AIAA Journal of Guidance, Control, and Dynamics, 28*, 1015–1026.
7. Bertsimas, D., & Stock-Patterson, S. (1998). The air traffic management problem with enroute capacities. *Operations Research, 46*(3), 406–422.
8. Bilimoria, K. D. (2000). A geometric optimization approach to aircraft conflict resolution. In *AIAA guidance, navigation, and control conference and exhibit*. Denver, CO.
9. Bilimoria, K., Sheth, K., Lee, H., & Grabbe, S. (2000). Performance evaluation of airborne separation assurance for free flight. In *AIAA guidance, navigation, and control conference*. Denver, CO.
10. Bilimoria, K. D., Sridhar, B., Chatterji, G. B., Shethand, K. S., & Grabbe, S. R. (2001). FACET: Future ATM concepts evaluation tool. *Air Traffic Control Quarterly* **9**(1), 1–20.
11. Bonaceto, C., Estes, S., Moertl, P., & Burns, K. (2005). Naturalistic decision making in the air traffic control tower: combining approaches to support changes in procedures. In *Proceedings of the seventh international NDM conference*. Amsterdam, The Netherlands.
12. Campbell, K., Cooper, W. J., Greenbaum, D., & Wojcik, L. (2000). Modeling distributed human decision making in traffic flow management operations. In *3rd USA/Europe air traffic management R & D seminar*. Napoli.
13. Christodoulou, M., & Costoulakis, C. (2004). Nonlinear mixed integer programming for aircraft collision avoidance in free flight'. In *IEEE MELECON*, Vol. 1. Dubrovnik, Croatia, pp. 327–330.
14. Devasia, S., Heymann, M., & Meyer, G. (2002). Automation procedures for air traffic management: A token-based approach. In *Proceedings of American control conference*, Vol. 1. pp. 736–741. Atlanta, GA
15. Donohue, G. L., & Shaver III, R. D. (2008). *TERMINAL CHAOS: Why U.S. Air Travel is broken and how to fix it*. Webster: American Institute of Aeronautics and Astronautics.
16. Eby, M. S. & III, W. E. K. (1999). Free flight separation assurance using distributed algorithms. In *Proceedings of aerospace conference*. Aspen, CO.
17. Erzberger, H., McNally, D., Foster, M., Chiu, D., & Stassart, P. (1999). Direct-to tool for en route controllers. In *IEEE workshop on advanced technologies and their impact on air traffic management in the 21st century*. Capri, Italy.
18. FAA OPSNET Data Jan–Dec 2007. (2007). US Department of Transportation website. http://www.faa.gov/data_statistics/.
19. Grabbe, S. & Sridhar, B. (2006). Central East Pacific Flight Routing'. In *AIAA guidance, navigation, and control conference and exhibit*. Keystone, CO.
20. Heymann, M., Meyer, G., & Resmerita, S. (2004). An agent based framework for control of merging air traffic. In *Proceedings 16th IFAC symposium on automatic control in aerospace*. St. Petersburg, Russia.
21. Hill, J. C., Johnson, F. R., Archibald, J. K., Frost, R. L., & Stirling, W. C. (2005). A cooperative multi-agent approach to free flight. In *AAMAS '05: Proceedings of the fourth international joint conference on autonomous agents and multiagent systems*. New York, NY, USA, pp. 1083–1090.
22. Hwang, I., Balakrishnan, H., & Tomlin, C. J. (2006). State estimation for hybrid systems: applications to aircraft tracking. IEE Proceedings of Control Theory and Applications, 153, 556–566.
23. Joint Economic Commission Majority Staff (2008). Your Flight Has Been Delayed Again'. pp. 1–12.
24. Jonker, G., Dignum, F., & Meyer, J.-J. (2007). Achieving cooperation among selfish agents in the air traffic management domain using signed money. In *Proceedings of the sixth international joint conference on autonomous agents and multi-agent systems*. Honolulu, HI.
25. Kelly, W. I., & Eby, M. (2000). Advances in force field conflict resolution algorithms. In *AIAA guidance, navigation, and control conference and exhibit*. Denver, CO.
26. Knight, J. C. & Parikh, S. (2002). Simulation technology for free flight system performance and survivability analysis. In *Proceedings of the 21st digital avionics systems conference*, pp. 13D5–1–13D5–9. Irvine, CA.
27. Kohn, W., Remmel, J., Moser, W., & Cummings, B. (1997). Free flight ATC using hybrid agent systems. In *Proceedings of the 36th IEEE CDC*. San Diego, CA.

28. Ma, Z., Cui, D., & Cheng, P. (2004). Dynamic network flow model for short-term air traffic flow management. *IEEE Transaction on Systems, Man, and Cybernetics Part A: Systems and Humans, 34*(3), 351–358.
29. McNally, D., & Gong, C. (2006). Concept and laboratory analysis of trajectory-based automation for separation assurance. In *AIAA guidance, navigation and control conference and exhibit*. Keystone, CO.
30. Menon, P. K., Sweriduk, G. D., & Bilimoria, K. D. (2004). New approach for modeling, analysis and control of air traffic flow. *Journal of Guidance, Control, and Dynamics, 27*(5), 737–744.
31. Menon, P. K., Sweriduk, G. D., & Sridhar, B. (1999). Optimal strategies for free flight air traffic conflict resolution. *Journal of Guidance, Control, and Dynamics, 22*(2), 202–211.
32. Mueller, E. R., Chatterji, G. B. (2002). Analysis of aircraft arrival and departure delay characteristics. In *AIAA aircraft technology, integration and operations (ATIO) conference*. Los Angeles, CA.
33. Pallottino, L., Feron, E., & Bicchi, A. (2002). Conflict resolution problems for air traffic management systems solved with mixed integer programming. *IEEE Transactions on Intelligent Transportation Systems, 3*(1), 3–11.
34. Pechoucek, M., & Sislak, D. (2009). Agent-based approach to free-flight planning, control, and simulation. *IEEE Intelligent Systems, 24*(1), 14–17.
35. Pechoucek, M., Sislak, D., Pavlicek, D., & Uller, M. (2006). Autonomous agents for air-traffic deconfliction. In *Proceedings of the fifth international joint conference on autonomous agents and multi-agent systems*. Hakodate, Japan.
36. Quinn, C., & Zelenka, R. (1998). ATC/air carrier collaborative arrival planning. In *2nd USA/Europe air traffic management R & D seminar*. Orlando, FL.
37. Raffard, R. L., & Tomlin, C. J. (2005). Second order optimization of ordinary and partial differential equations with application to air traffic flow. In *Proceedings of the AACC American control conference*, pp. 798–803. Portland, OR.
38. Raffard, R. L., Waslander, S. L., Bayen, A. M., & Tomlin, C. J. (2005). A cooperative distributed approach to multi-agent Eulerian network control: Application to air traffic management. In *AIAA guidance, navigation, and control conference and exhibit*. San Francisco, CA.
39. Sislak, D., Pechoucek, M., Volf, P., Pavlicek, D., Samek, J., Mařík, V., & Losiewicz, P. (2008a). AGENTFLY: Towards multi-agent technology in free flight air traffic control. In M. Pechoucek, S. Thompson, & H. Voss (Eds.), *Defense industry applications of autonomous agents and multi-agent systems* (Chap. 7, pp. 73–97). Birkhauser Verlag, Basel.
40. Sislak, D., Samek, J., & Pechoucek, M. (2008b). Decentralized algorithms for collision avoidance in airspace. In *Proceedings of seventh international conference on autonomous agents and multi-agent systems*, pp. 543–550. Estoril, Portugal.
41. Sridhar, B., Soni, T., Sheth, K., & Chatterji, G. B. (2006). Aggregate flow model for air-traffic management. *Journal of Guidance, Control, and Dynamics, 29*(4), 992–997.
42. Sridhar, B., & Swei, S. (2006). Relationship between weather, traffic and delay based on empirical methods. In *6th AIAA aviation technology, integration and operations conference (ATIO)*. Wichita, Kansas.
43. Stipanovic, D. M., Shankaran, S., & Tomlin, C. J. (2005). Multi-agent avoidance control using an M-matrix property. *Electronic Journal of Linear Algebra, 12*, 64–72.
44. Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.
45. Tomlin, C. (1998). Hybrid control of air traffic management systems. Ph.D. thesis, University of California at Berkeley.
46. Tomlin, C., Pappas, G., & Sastry, S. (1998). Conflict resolution for air traffic management: A study in multiagent hybrid systems. *IEEE Transaction on Automatic Control, 43*(4), 509–521.
47. Tumer, K., Agogino, A. (2007). Distributed agent-based air traffic flow management. In *Proceedings of the sixth international joint conference on autonomous agents and multi-agent systems*, pp. 330–337. Honolulu, HI.
48. Tumer, K., & Agogino, A. (2009). Improving air traffic management with a learning multiagent system. *IEEE Intelligent Systems, 24*(1), 18–21.
49. Tumer, K. & Wolpert, D. (Eds.). (2004). *Collectives and the design of complex systems*. New York: Springer.
50. van Gent, J., Ruigrok, R., & Ruigrok, R. (1998). Conceptual design of free flight with airborne separation assurance. In *AIAA guidance, navigation, and control conference*, pp. 807–817. Boston, MA.
51. Wangermann, J. P. & Stengel, R. F. (1999). Optimization and coordination of multiagent systems using principled negotiation. *Guidance, Control and Dynamics, 22*(1), 43–50.

52. Wollkind, S., Valasek, J., & Ioerger, T. (2004). Conflict resolution for air traffic management using cooperative multiagent negotiation. In *AIAA guidance, navigation, and control conference*. Providence, RI.
53. Wolpert, D. H., & Tumer, K. (2001). Optimal payoff functions for members of collectives. *Advances in Complex Systems, 4*(2/3), 265–279.