Multi-agent Simulation of Airline Operations and Infrastructure Utilization in the Presence of Airline Strategies

Guillermo Calderón-Meza (Ph.D. Candidate)*, Lance Sherry, Ph.D†
Center for Air Transportation Systems and Research
George Mason University
*Email: gcaldero@gmu.edu, †Email: lsherry@gmu.edu

***Abstract*—Investment in the NextGen National Airspace System (NAS) is intended to increase the effective capacity of the airspace and airports. Previous research, in several domains, has identified cases where large infrastructure investments were under-utilized due to migration and/or adaptation of users (e.g. airlines).**

This paper examines the impact of airline "gaming" on the use of trans-continental routes. Two airlines with equal cost structures are given options for two trans-continental routes each with different distances. In the base case, the airlines conform to the rules of an air traffic controller and are assigned to the routes to balance traffic. In other cases, airlines select a preferred route based on information about the length of the route, and the length of the departure queue. The airlines act as independent agents, and each scenario is repeated several times to account for the stochastic elements in the simulation. The results indicate a tradeoff between utilization of the resources and cost to the users. Results also indicate that alternate strategies could migrate closer to the optimal tradeoff between utilization and user costs. The implications of these results are discussed.

***Index Terms*—Multi-agent simulation, airline gaming, optimal traffic management, airline strategies**

I. Introduction

The Air Transportation System of the United States is showing signs of saturation in the form of increasing flight and passenger delays, cancellations, and diversions [3]. Some of the reasons are temporal reductions in the capacity due to weather, and over-scheduling. Weather creates over-scheduling by abruptly and unexpectedly reducing capacity. Also, over-scheduling is the result of the current set of rules that allow the airlines define their own schedules [3]. Clearly, mitigating or eliminating problems due to over-scheduling is an effective way to improve the performance of the system.

Game Theory predicts that all the airlines define their schedules to include as many flights as possible to capture the largest market share. No airline will see any incentive in unilaterally changing its schedule to reduce congestion. The fear of loosing market share due to the "Tragedy of the Commons" is too big. Since all airlines have the same goal, and all are "rational agents" according to Russell and Norvig [10], the strategy profile is a Nash Equilibrium. However, some games can perform better than the Nash Equilibrium if the agents share information [5][9][8]. All the agents could simultaneously use different strategies [10] or other forms of interaction as described by Ferber [2]. The rules of the game can be changed to limit the "greed" of the agents for the sake of the global improvements. *Slot auctions* [3] and the so-called *congestion pricing* as explained by Neufville and Odini [6] are two of these rule-changing techniques intended that could reduce congestion.

Changes to the air transportation system require careful analysis. Creating tools to evaluate the potential effects of the changes is an import research goal [13][4][12]. This paper uses a multi-agent-based simulation to model airlines operations and infrastructure utilization in the presence of airline route selection strategies. Currently, air traffic controllers assign route slots to the aircraft with one criterion only: traffic balancing. This strategy is called *first come-first served* (FCFS). The main hypothesis of the paper is that strategies other than FCFS can improve the performance of the current system and give more flexibility to the airlines. Performance in this case is defined in terms of utilization of the route slots and in terms of aggregated airline cost.

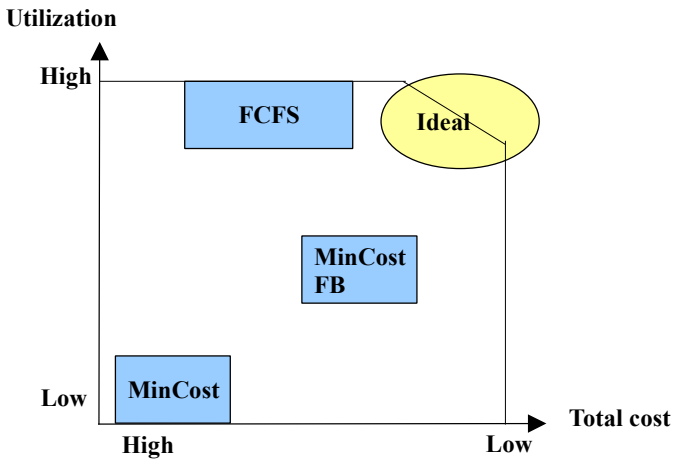

Fig. 1. Pareto frontier to guide the optimization efforts.

Figure 1 shows that the simulation demonstrated that

FCFS achieves the best utilization of the route slots. Other strategies can achieve good performance if they use more information to make decisions. But they can only approach the performance of FCFS. In terms of cost, other strategies can perform better than FCFS under certain circumstances. The results will be explained in more detail in the paper.

The paper is organized as follows. The problem description section of the paper describes in more detail and explains the parameters of the simulation. The method section describes the experiments: the tools, software, and how the results will be analyzed. The next two sections describe the results obtained and the conclusions and some future work.

II. Problem description

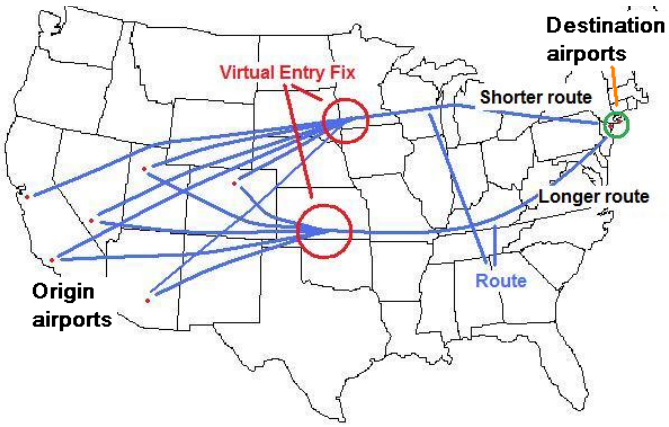

Fig. 2. Conceptual diagram of the situation modeled.

The problem analyzed in this study is an abstraction of a real situation in the field of air transportation (see Figure 2). Aircraft scheduled to depart from several airports on one side of the country can choose from a set of *alternate trans-continental routes* to their common destination airport (or multiplex) on the other side of the country. In the real world, air traffic controllers must assign routes to the aircraft, so that the traffic is balanced among routes, in a *first come-first served* discipline. The routes are divided into slots due to safety rules (*separation distances*). The controller assigns a route to the flight before allowing the flight to enter the route. Usually, aircraft wait on the ground until the controllers allow them to fly just-in-time to enter an available slot in a route. Sometimes this *delay on the ground* has other reasons like weather, security, and mechanical problems. Regardless of the reasons, ground delays can be modeled by *departure queues*. Delays imply wait times and costs for passengers and airlines, but they are consequences of the need for synchronization between different parts of the system. Once an aircraft enters a route, it flies at a fairly constant speed to maintain separation with the other aircraft flying the same route.

When modeling this situation it is assumed that aircraft push back from the gates either on-time or with a *push back delay* that is exponentially distributed. The delays on the ground described before are represented by a *virtual entry queue*, or *entry queue* for short, associated to a route. Each route has its own entry queue that consolidates the departure delays of all the aircraft assigned to the route across all airports. The queues are also divided into slots the same way routes are. The entry queue is virtually located before the so-called *virtual entry fix* which is the point in which all the aircraft enter the route (see Figure 2). The number of slots from the entry fix to the destination is called the *route length*. In this study, all slots are either 5 minutes (for time slots) or 25 nm (for distance slots). A constant speed of 300 nm/h is assumed for flying aircraft so that it takes 5 minutes to fly 25 nm. Under this conditions, an air traffic controller will make the aircraft wait on the ground (modeled by waiting in the entry queue) after they push back until there is an open slot in a route. When there is an open slot the flight will be assigned a route and it will enter the entry queue and, later, the route following the sequence created by the queue. If more resources or information were available, as it is proposed by NextGen (SEVEN, SWIM [1]) either the controller or the airlines themselves could decide which route to choose based on criteria other than the balance of route traffic.

For the first come-first served discipline, the controller is modeled by airlines that chose a route based only on the length of the virtual entry queue, the aircraft is assigned to the shortest queue. This is basically what the controllers do: they try to balance traffic by equally distributing flights among the routes. Therefore, the simulation does not contain an explicit "controller agent", the behavior of the airlines implement the tasks of a controller as they are today. This behavior is called *conforming* (or *FCFS*) since it conforms to the current rules of the air transportation system. Situations in which the airlines have more information to decide, and can game the system, are represented by the other strategies: *minimum cost* (*MinCost*), and *minimum cost with feedback* (*MinCostFB*). The MinCost behavior chooses always the shortest route to minimize the flying cost. The MinCostFB chooses the route to minimize the cost of flying plus the cost of waiting in the queue. The feedback part is the cost of waiting: choosing the shortest route (as in MinCost) generates long waiting queues and the cost of waiting becomes important in the decision. Theoretically, the airlines described above behave like *rational agents* [10]. They show "*greedy*" behavior since they decide based on local information only: no system-wide, historic or future information is considered to decide. In this situation all airlines have the same objective of maximizing profit by minimizing cost. Airlines can use their own strategies and tactics to win this *partially informed game*. In some cases the decisions of the airlines will result in under-utilization of some resources and congestion in others. The goal of this study is to evaluate the effects of several strategies of the airlines in the utilization of the routes and the aggregated airline cost of the operations.

The *independent variables* in this study are:

- the *behavior* of the airline, which can be *FCFS*, *MinCost* or *MinCostFB*;
- the flight *schedule* of the airline. It is *flat* (one flight per slot) for all scenarios and contains 30 consecutive flights for each airline;
- the nature of the *push back delay*, which could be constant (zero in this case) or exponentially distributed either with $\lambda = 3$ slots, equivalent to 15 minutes in average, or $\lambda = 5$ slots, equivalent 25 minutes in average;
- the *distance* of each alternate route measured in 25 nm slots. This is an integer number from 39 to 75 slots (from 975 nm to 1875 nm). One of the routes remains constant at 57 slots (1425 nm $\approx$ great circle distance from DEN to JFK) through all the scenarios;
- the *cost of flying* each mile (for each airline), which is normalized to 1.0 for all airlines;
- the *cost of waiting* in queue (for each airline), which is relative to the cost of flying and can take one of three values: 0.3, 0.5, or 0.7. This cost is also the same for all airlines;
- the *cost of the push back delay*, which is the same as the cost of waiting for these experiments.

The *dependent variables* are: total *unused route slots*, and the total *cost* for the whole system. *Total delay* can also be a dependent variable, but the results show that it behaves the same way as the number of unused route slots. Then it will not be presented here. The same variables could be measured individually for each airline, but that comparison is out of the scope of this study.

III. Method

These experiments are simulated using the MASON discrete event multi-agent simulation platform [11][1]. MASON is a free, open-source, Java library designed and implemented at George Mason University and it is comparable to Swarm[2], Repast[3], NetLogo[4], StarLogo[5], Arena[6], Simula[7], Psim[8], Ascape[9], and many other tools. Considering the small number of agents involved in the situation being modeled, MASON is more powerful than necessary, but it provides an easy-to-use framework to do multi-agent simulations and to support future planned experiments.

A. The simulation model

The environment of the airlines, i.e. the Air Transportation System, is represented in the simulation by a Java class called *Universe* (see Figure 3) that extends the MASON class called *SimState*. This class is required in any MASON simulation. It initializes the simulation, creates, and schedules the agents. Objects of this class can also be used as shared memory for the communication between agents. The method public *start()* of the this class is where the actual execution starts. Classes that implement the *Steppable* interface are the agents of the system. The interface requires the method *step()* to be implemented. This method is executed once every step for each of the active agents. The order in which the step method is executed is random among the agents. Agents can be grouped to guarantee that all the agents of a group get their step method executed before or after the step method of agents in other groups. The *BasicAirline* abstract class for the basis for the all the airline agents. A direct descendant of this this class must implement the *myBehavior()* abstract method, and use the step method implementation of BasicAirline. In general, the myBehavior method is implemented as follows:

- If there is no flight scheduled for the current time or if the airline has used all its aircrafts then return a special value (i.e.: finish the current execution of the method) to signal that there was no decision in terms of which route was chosen;
- Sort the routes in ascending order according to the length of their entry queues (*FCFS*), cost of flying the route (*MinCost*), or aggregated cost of waiting on the ground (entry queue length) and flying the route (*MinCostFB*): smaller value first. Therefore, the only change between airlines is in the *objective function* to minimize;
- If there is a tie in the smaller values, pick any of the tied routes randomly (uniform distribution). This is one of the sources of uncertainty in the simulation. Another source is that the airline agents are processed in a random order. But this one is compensated because the airlines do not change the state of the system but after all of them have being processed;
- Communicate the decision by sending the flight to the intents list of the route. Return the index (id number) of the selected route.

Each airline is represented by an object that has a schedule associated. The schedule is represented by a *FlightSchedule* object (see Figure 3). The profile of the schedule through time can be modified. In these experiments, all the schedules are flat: one flight per time slot, per airline, until all the flights of the airline are used. *The push back delays* are generated in this class too, before the simulation takes place. The delay could be intentional (*ground delay programs*) or unintentional (missing crew, technical problem). The probability distribution of this type of delay is usually exponential, but it could also be constant (usually zero).

A trans-continental route is represented by a *Route* agent (see Figure 3). The agent is composed by three objects: a queue to represent the *flights* en-route, another queue to represent the *virtual entry queue*, and a list of intents (of the flights assigned to this queue but not in it yet). The *virtual entry fix queue* is a representation of the sequence in which flights must enter the route when they reach the entry point.

[1] See http://cs.gmu.edu/~eclab/projects/mason/ for more information.
[2] http://www.swarm.org/index.php/Main_Page
[3] http://repast.sourceforge.net/
[4] http://ccl.northwestern.edu/netlogo/
[5] http://education.mit.edu/starlogo/
[6] http://www.systemsnavigator.com/sn_website/?q=node/38
[7] http://en.wikipedia.org/wiki/Simula
[8] http://www.powersys.fr/
[9] http://ascape.sourceforge.net/

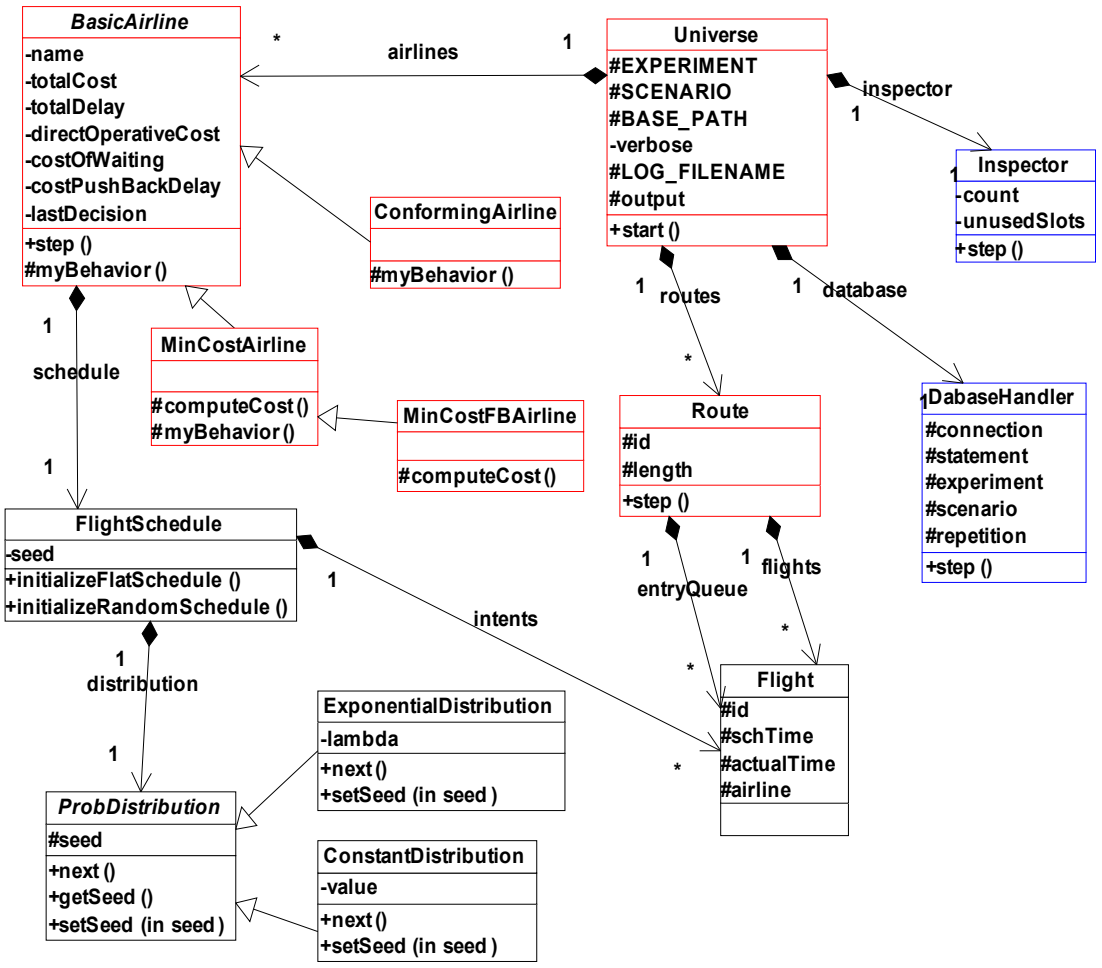

Fig. 3. Class diagram of the simulation model.

The route itself, after the entry point, is also represented by a queue; the implications are that flights travel at the same speed (300 nm/h), there are no passings en-route, the route is divided in *slots* of the 25 nm each. The behavior of the route is as follows:

- All the flights in the intents list are pushed into the entry queue. Also the cost and delay for each flight are computed and aggregated to the total for the corresponding airline;
- If the oldest flight has reached the end of the route (its maximum length) remove it;
- If the entry queue is not empty, remove the oldest flight from it and push it into the route. The cost of flying the route is computed and aggregated to the total of the corresponding airline. If the queue is empty, push a null (special) value into the route to signal that there is an unused slot there.

Table I shows the great circle distances between some airports and JFK and their equivalent number of 25 nm slots. This table is given for reference only and to give and to explain the choice of some of the values in the simulations.

The *Inspector* agent (see Figure 3) computes the total number of unused slots for all the routes. The number of unused slots for a route is the number of times a null value is pushed into the flights en-route queue. The Inspector object starts counting unused slots from the time slot of the first scheduled flight across all airlines. The Inspector object stops counting slots when all the airlines have sent all their scheduled flights, none of the flights is delayed in push back, and all the entry queues are empty. In every step of the simulation the value of the total number of unused slots is stored into the database, but only the last one is used in further analyses.

TABLE I
GREAT CIRCLE DISTANCES AND NUMBER OF SLOTS FROM SEVERAL AIRPORTS TO THE NEW YORK JOHN F. KENNEDY AIRPORT (JFK)

Origin airport	**Great circle distance (nm)**	**Number of slots (5 min ≈ 25 nm/slot at 300 ks)**
Denver International (DEN)	1413	57
Salt Lake City (SLC)	1729	69
Phoenix (PHX)	1871	75
Los Angles International (LAX)	2151	86

The *DatabaseHandler* agent (see Figure 3) stores the current state of the simulation in a database. The object allows several databases since it uses the *JDBC* technology. The current version of the code can use MS-SQL Server databases and MS-Access files.

B. Experimental design

This study consists of nine *experiments*. Each experiment is divided into several *scenarios*. And each scenario is *repeated* several times (*Monte Carlo* simulation) to account for the stochastic nature of the simulation. The combination of the two independent variables (behavior and push back delay) defines an experiment. The scenario is defined by the combination of two variables (distance and cost of waiting, since cost of flying is always 1.0).

The distance of one route is set to 57 slots for all the scenarios and the other route (only two routes are used in these experiments) varies from 39 to 75 for a total of 19 x 3 = 57 scenarios per experiment. The reason to use two routes is ease of results analysis. The reason to use a range of 39 to 75 for one route is to keep the route distance ratio reasonably close to 1.0; alternate routes are usually not very much longer or shorter that the original route. In these experiments the ratio of distances between the two routes goes from 1.32 to 0.68. The cost of waiting has an infinite number of possible values, but in these experiments only three values are used to observe the effect of this price in the decisions of the airlines. It is assumed that the behavior of the other values can be extrapolated from the behavior of these three values.

Each repetition of a scenario starts with the same parameters and conditions except for the seed of the internal random number generator of MASON and the seed of the random number generator of the push back delay generator. Both random number generators are implementations of the *Mersenne Twister Fast* algorithm [7] and their seeds are Java long numbers. These two different seeds mean that when there is a tie in costs during the decision making process of an airline, the decision will be different among repetitions. It also means that an individual flight would be delayed differently among repetitions. Every scenario is executed 30 times and the mean value of the executions is recorded. Every repetition of the simulation iterates for 70 time steps[10].

IV. Results

Since there are two dependent variables in the experiments, the results are presented in two sets of charts. Each chart compares the three strategies for the 19 scenarios (route distance ratios) corresponding to a particular wait to fly cost ratio. Each scenario is repeated 30 times and the values shown in the charts are the mean of these repetitions. As stated before all the schedules are flat and contain 30 flights. Several experiments were made by changing the values of the push back delay to 0 (no delay), 3 slots (15 minutes) in average, and 5 slots (25 minutes) in average. The difference in the dependent variables is not significant and therefore only the results for 5 slots in average are shown here. Since the delays behave the same way as the utilization, but with different numerical values. So delay charts will not be shown here.

The first set of charts shows the unused route slots. The second set of charts shows that cost for the whole system.

A. Total unused route slots

The following charts show the average total unused route slots. Each chart compares the three strategies (behaviors) FCFS, MinCost, and MinCostFB as functions of the route distance ratio. The difference between the charts is the "Wait to fly cost ratio".

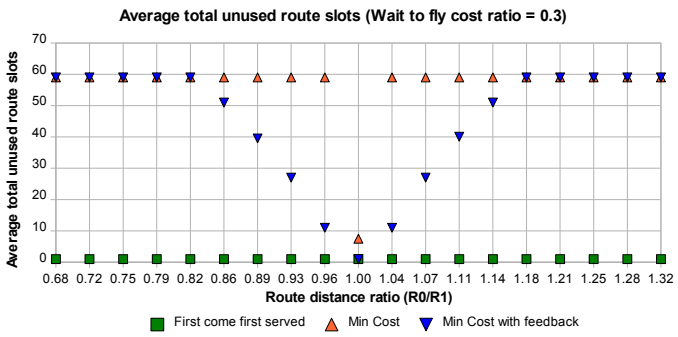

Fig. 4. Average total unused route slots (Wait to fly cost ratio = 0.3).

Figure 4 shows FCFS as the best strategy. The air traffic controller balances the utilization of the routes in the best possible way. This is because the controller does not consider any cost to make the decisions, but only the traffic. The other two strategies leave many route slots unused when the route distance ratio is very high or very low. This is because MinCost selects only the shortest route and leaves the longest one completely unused. That is also the reason for the improvement in utilization when the distance ratio is 1; where this strategy chooses randomly any route. The effect of the random choice is a better balance in the traffic. The performance of MinCost never reaches the level of FCFS.

When one route is 18% longer or shorter than the other, i.e. when the distance ratio is 0.82 or 1.18, MinCostFB starts using both routes and reducing the number of unused slots. In fact it linearly approaches the performance of FCFS and reaches it when the distance ratio is 1. This is because in these points the cost of waiting in the queue is high enough to make it profitable to fly the longest path instead of waiting for the shortest. Furthermore, as the distance ratio approaches one, the difference in cost of flying the two routes becomes smaller and the price of waiting in queue is more significant in these cases.

[10]The number of steps is actually a parameter of the simulation, but it is fixed to 70 for all these experiments.

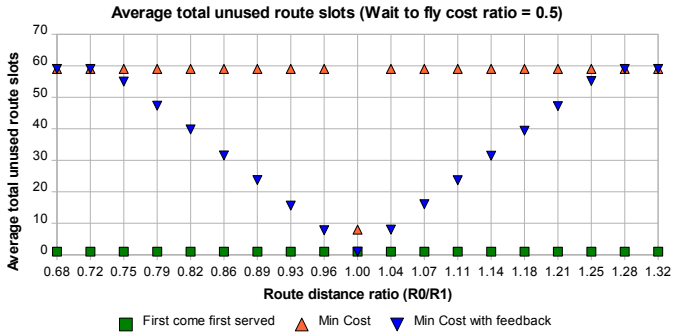

Fig. 5. Average total unused route slots (Wait to fly cost ratio = 0.5).

Figure 5 shows again first come-first served as the best strategy, balancing the utilization the best possible way. MinCostFB again leaves a route unused when the route distance ratio is too high or too low, but it starts its linear approach of the FCFS performance when the distance ratio is 0.72 or 1.28. In other words, when one route is 28% longer or shorter than the other. This shows the effect of a more costly wait time, because in the previous comparison the point were 0.82 and 1.18. Despite of this difference, MinCostFB reaches the performance of FCFS when the distance ratio is 1 as before. MinCost remains the same as in the previous comparison, because the wait cost is not considered in this strategy to make decisions.

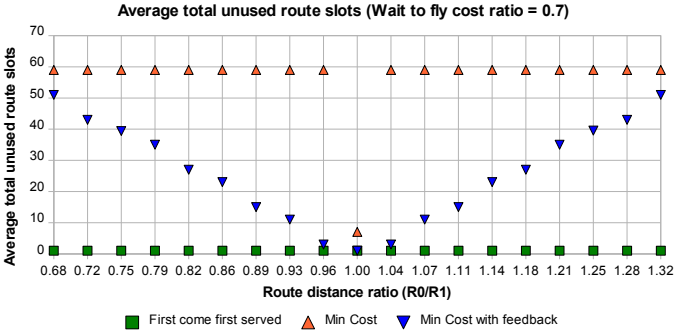

Fig. 6. Average total unused route slots (Wait to fly cost ratio = 0.7).

Figure 6 shows again first come-first served as the best strategy, balancing the utilization the best possible way. The results of MinCost are the same as in the other two comparisons due to the irrelevance of the wait cost for this strategy. MinCostFB behaves as it did in the other two comparisons. The linear approach starts at distance ratio values that are out of scale because of the effect of higher wait costs.

Table II shows the confidence intervals of a paired t-test of these results. Each cell of the table contains the 95% confidence interval for the comparison (difference in the values) between two strategies for a particular wait to fly cost ratio. Since none of the intervals includes 0, the test indicates that each strategy is statistically different to the other two strategies. This result means that using different strategies effectively gives different results.

TABLE II
PAIRED T-TEST FOR THE DIFFERENCE IN UTILIZATION OF ROUTES WITH A CONFIDENCE OF 95%.

Wait to fly cost ratio	**95% confidence intervals of the difference (α = 0.05, 19 data points)**		
	FCFS-MinCost	**FCFS-MinCostFB**	**MinCost-MinCostFB**
0.3	[-59.99, -50.59]	[-51.50, -35.83]	[5.00, 18.25]
0.5	[-59.97, -50.66]	[-42.62, -26.83]	[13.32, 27.86]
0.7	[-60.01, -50.52]	[-31.56, -18.64]	[23.74, 36.59]

The summary of these results is that MinCost is never the best strategy in terms of utilization. It only achieves an acceptable performance when the routes are of equal length. MinCostFB is either equal or better than MinCost. FCFS is better or equal to MinCostFB. When the route distance ratio is too high or too low MinCostFB leaves many many slots unused. But there is a point, defined by the wait to fly cost ratio, from which the performance of MinCostFB starts to linearly improve with the distance ratio until it reaches the FCFS performance at a distance ratio of 1.

B. Total airline cost of the system

The following charts show the average total airline cost. The cost for an airline is the summation of the cost of push back delays, plus the cost of waiting in the queue, plus the cost of flying a route. All costs are normalized with respect to the cost of flying a mile. It is assumed that all aircraft are the same size (number of passengers) and have the same type and number of engines. The costs of waiting and flying are the same for all airlines. The total airline cost is the summation of cost across airlines at the end of each execution. The average is obtained across the 30 repetitions of each scenario.

Each of the following charts compares the three strategies FCFS, MinCost, and MinCostFB as functions of the route distance ratio. The difference between the charts is the "Wait to fly cost ratio".

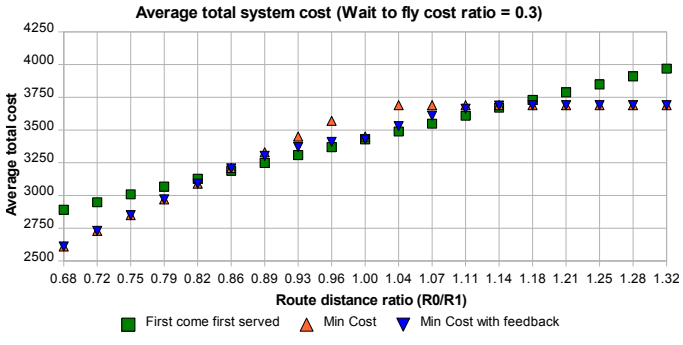

Fig. 7. Average total airline cost (Wait to fly cost ratio = 0.3).

Figure 7 shows that MinCost and MinCostFB are truly minimum cost only when the route distance ratio is very high

or very low and when the distance ratio is 1. In other cases, FCFS is better in terms of cost. The distance ratio values in which the cost of FCFS equals the cost of MinCost and MinCostFB is between 0.82 and 0.86 for one side and 1.14 and 1.18 for the other side of the chart, i.e. when one route is about 16% shorter or longer than the other. The reason for this alternating behavior can be explained as follows. When the route distance ratio is 1, the cost of flying the routes makes no difference. In this case the cost of waiting is very low and, since all the routes are utilized, no long wait queues are expected. Then the total cost of waiting in queue should be very low. In these conditions all the strategies are actually selecting the route randomly and the total cost is the same for all of them. When the distance ratio is not 1 the cost of flying the route is determined by the shortest route in the case of MinCost. This explains why this strategy grows linearly, from 0.68 toward 1.0: the shortest route is becoming larger. It also explains why the strategy results in a horizontal line for the cost from 1.0 to 1.32, because the shortest route is constant at 57 slots. The cost in this case includes the cost of waiting, but this cost is the same across distance ratios because all the aircraft fly the same route and the length of the queue grows the same way regardless of the length of the routes. The case of MinCostFB is more complex. In the extremes it follows the results of MinCost. But when one route becomes less than 16% longer or shorter than the other, MinCostFB becomes better than MinCost, but still not as good as FCFS. MinCostFB achieves better results in these distance ratios because it compensates the waiting costs by selecting longer routes sometimes; the cost of waiting is significantly reduced. FCFS shows a linearly growing cost with respect to the distance ratio because it just distributes flights among the rotes and the total distance to fly grows with the distance ratio. There are low wait costs because the traffic is very well balanced.

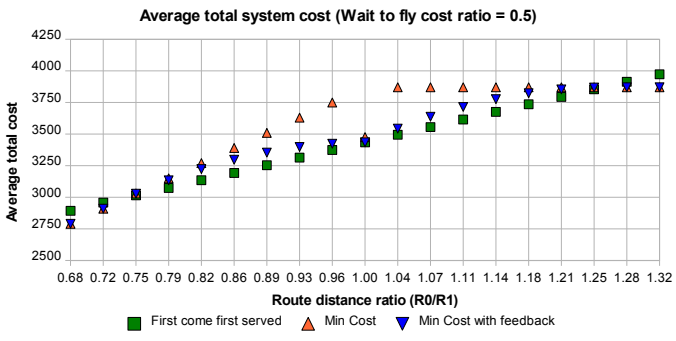

Fig. 8. Average total airline cost (Wait to fly cost ratio = 0.5).

Figure 8 shows that the FCFS strategy maintains the same linear behavior as before, even the numerical values are the same. This is because the cost of waiting has a minor or no effect: there are no waiting queues. The curve for the MinCost strategy has the same shape as before, but it is displaced upward (higher values) because the cost of waiting is higher in this case. When the distance ratio is 1, the cost for the MinCost strategy is only a little higher than in the previous comparison, but it still close to the costs of MinCostFB and FCFS. This follows from reducing the waiting queues by randomly selecting the route. MinCostFB is also displaced upward. It equals the curve from MinCost until the distance of one route is about 23% shorter or larger than the other. Then MinCostFB starts to achieve better results than MinCost, but not as good as FCFS. It only equals the cost of FCFS when the distance ratio is 1.

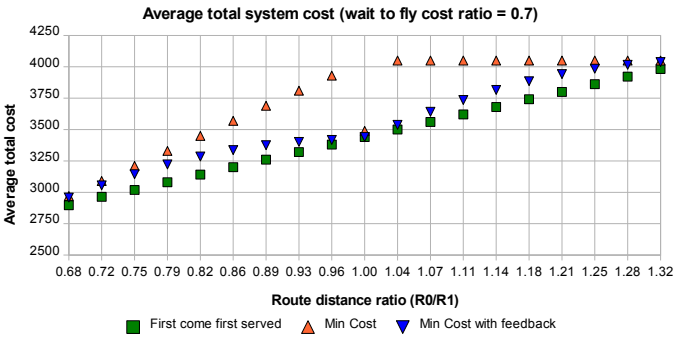

Fig. 9. Average total airline cost (Wait to fly cost ratio = 0.7).

Figure 9 shows a curve for the FCFS strategy that is equal to the curve of the two previous comparisons, for the same reasons as before. The curve for MinCost is further displaced upward, to the level that it never performs better than FCFS. This is because the cost of waiting is very significant in this case and the waiting queue is very long for one of the routes. When the distance ratio is 1, it approaches the performance of FCFS and MinCostFB, but its value is higher there also. In this case the strategy is selecting the route randomly and therefore the traffic is balanced and the waiting queues are short. The curve for MinCostFB is also further displaced upward due to the higher waiting costs. It only reaches the same performance of FCFS when the distance ratio is 1. This is because in this point the strategy is actually selecting the route randomly. It is better than MinCost except when the one route is 32% shorter or larger than the other. This is because MinCostFB generates shorter waiting queues than MinCost, but as short as FCFS.

Table III shows the confidence intervals of a paired t-test of total airline cost. Each cell of the table contains the 95% confidence interval for the comparison (difference in the values) between two strategies for a particular wait to fly cost ratio. Only one of the intervals includes 0: the test indicates that FCFS is not statistically different to MinCost when the wait to fly cost ratio is 0.3. But the strategies are different in all the other cases. This result means that using different strategies effectively gives different results, but in one case, it is not possible to differentiate between two strategies.

The summary of these results is that FCFS performs the same regardless of the wait to fly cost ratio. It is only affected by the route distance ratio: its total cost grows linearly with

TABLE III
PAIRED T-TEST FOR THE DIFFERENCE IN TOTAL AIRLINE COST WITH A CONFIDENCE OF 95%.

Wait to fly cost ratio	**95% confidence intervals of the difference (α = 0.05, 19 data points)**		
	FCFS-MinCost	**FCFS-MinCostFB**	**MinCost-MinCostFB**
0.3	**[-26.32, 97.57]**	[17.44, 113.21]	[8.80, 50.64]
0.5	[-192.65,-68.25]	[-63.32, -9.38]	[49.53, 138.67]
0.7	[-361.75, -229.48]	[-115.18, -80.71]	[130.42, 264.91]

the distance ratio. When the wait to fly cost ratio is greater than 0.5, FCFS performs better than MinCost and MinCostFB with the exception of the distance ratio of 1. In that ratio MinCostFB and FCFS are equal and MinCost is only a little more costly. Whenever the wait to fly cost ratio is 0.5 or smaller, MinCostFB and MinCost can perform better than FCFS when one of the routes is considerably shorter or larger than the other; the exact percentage is determined by the wait cost ratio: 16% for a wait to fly ratio of 0.3, 23% for a wait to fly ratio of 0.5, and more than 32% for a wait to fly ratio of 0.7. The difference in cost among the strategies is statistically significant in all the cases except one (highlighted in the table), when the wait to cost ratio is 0.3 and the strategies are FCFS and MinCostFB.

V. CONCLUSIONS AND FUTURE WORK

Balancing the traffic in the routes by using a first come-first served (FCFS) strategy gives the best performance in terms of utilization. But, using cost and feedback information could help other strategies approach the performance of FCFS. The wait to fly cost ratio and the route distance ratio define the conditions with which MinCost and, especially, MinCostFB can approach the FCFS performance.

Using cost information as criterion to assign routes can lead, under certain conditions, to superior results when compared to the costs obtained using traffic balancing. The conditions are defined by the wait to cost ratio and the route distance ratio.

For this reason, future research work is proposed to create strategies that obtain cost reductions and acceptable utilization performance by combining strategies or using more information to make decisions. For instance, a good strategy could exploit the knowledge about the wait to fly cost ratio and the route distance ratio to use either FCFS or MinCostFB strategy in search for better global performance. A change in the objective function of the airlines, perhaps including information about the current traffic, cost, utilization situation, can also result in better general performance. Finally, an agent that can learn from history and adapt is the long term goal of the this study.

The fundamental contribution of this paper is that the global performance of the system can be tweaked by changing only the behavior of the individual agents. The goals of the traffic controller (traffic balance) conflict with the goals of the airlines (profit) and the strategies must try to optimize both simultaneously. Figure 1 shows performance achieved by the strategies used in the study and where the ideal situation is. The proposed future work is intended to go to the ideal region.

ACKNOWLEDGMENT

Thank you for technical assistance to Maria Consiglio, Brian Baxley, Kurt Neitzke (NASA), George Hunter, Huina Gao (Sensis), Joe Post, Kimberly Noonan, Stephanie Chung (FAA), Midari Tancho (FAA), Terry Thompson, Mike Brennan, Mark Klopfenstein (Metron Aviation), George Donohue, John Ferguson, Jianfeng Wang (CATSR - GMU), Sean Luke (GMU).

Thank you for support to Michael Landis, Harry Swenson (NASA).

REFERENCES

[1] J.E. Dieudonne, H.L. Crane, J. Gonda, and S.R. Jones. Neo (nextgen tbos) provided by multi-agency surveillance soa (sdn). *Digital Avionics Systems Conference, 2007. DASC '07. IEEE/AIAA 26th*, pages 1.E.3–1–1.E.3–13, Oct. 2007.

[2] Jacques Ferber. *Multi-Agent Systems. An Introduction to Distributed Artificial Intellegence*. Adison-Wesley, United Kingdom, 1999. ISBN 0-201-36048-9.

[3] Eric Edwards George L. Donohue, Russel D. Shaver III. *Terminal chaos: why US air travel is broken and how to fix it*. American Institute of Aeronautics and Astronautics, Inc, USA, 2008. ISBN 978-1-56347-946-6.

[4] Daniel P. Greenbaum Leonard A. Wojcik Keith C. Campbell, Wayne W. Cooper Jr. Modeling distributed human decision making in traffic flow management operations. *Air Transportation Systems Engineering*, 193:227–237, 2001. ISBN 1-56347-474-3.

[5] Dave Knorr James Wetherly Mike Wambsganss Michael O. Ball, Robert L. Hofman. Assessing the benefits of collaborative decision making in air traffic management. *Air Transportation Systems Engineering*, 193:239–250, 2001. ISBN 1-56347-474-3.

[6] Richard De Neufville and Amedeo Odoni. *Airport Systems. Planning, Design, and Management*. McGraw-Hill, USA, 2003. ISBN 0-07-138477-4.

[7] M. Matsumoto T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simul.*, (3), 1998.

[8] Nicolas Bouge Sophie Vial Peter Martin, Alison Hudgell. Improved information sharing: A step toward the realization of collaborative decision making. *Air Transportation Systems Engineering*, 193:2161–175, 2001. ISBN 1-56347-474-3.

[9] Hayley J. Davidson R. John Hansmann. Effect of shared information on pilot/controller and controller/controller interactions. *Air Transportation Systems Engineering*, 193:205–223, 2001. ISBN 1-56347-474-3.

[10] Stuart Russel and Peter Norvig. *Artificial Intelligence. A Modern Approach*. Prentice Hall, USA, 2003. ISBN 0-13-790395-2.

[11] Liviu Panait Keith Sullivan Sean Luke, Claudio Cioffi-Revilla. Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 SwarmFest Workshop*, 2004.

[12] Clare J. Tomlin Steven L. Waslander. Efficient market-based air traffic control flow with competing airlines. *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 2843–2848, December 2006.

[13] Leonard A. Wojcik. Airline personalities and air traffic flow management: A simple agent-based model. In *AIAA 4th Aviation Technology, Integration and Operations Forum*, pages 1–16, Chicago, Illinois, USA, September 2004.

AUTHOR BIOGRAPHY

Lance Sherry is Associate Research Professor of System Engineering

and Operations Research and is Executive Director of the Center for Air Transportation Systems Research (CASTR) at GMU. Dr. Sherry is a system engineer with over 20 years of practical experience in air transportation operations and the design/flight-test/certification of commercial avionics. Dr. Sherry has served as control engineer, system engineer, lead system engineer, avionics flight test engineer, and program manager, has also served as Principal Investigator on research projects for FAA, NASA, NSF, DOT, DOE, airports, airlines, aircraft manufacturers and avionics vendors and has published over 100 papers and articles. He holds several patents and has won several awards for his work.

Guillermo Calderón-Meza is a PhD student at GMU conducting research in multi-agent-based simulations. He obtained his BS in Electronics at the Instituto Tecnológico de Costa Rica and an MSc in Electronic Systems at the Bolton Institute of Higher Education from England and the Universität Paderborn from Germany. He also took classes for an MSc in Computer Science at Instituto Tecnológico de Costa Rica and at the Universität Kaiserslautern from Germany. He has over twelve years of experience as a software engineer, software developer, and software project manager. He has worked for several Costa Rican and US companies and for the Fraunhofer Institut für Experimentelles Software Engineering in Germany.