Multi-agent Simulation of Airline Travel Market

Track: Real-World Applications

Abstract. This paper explores the use of a learning classifier system variant, XCS, in learning effective airline decision rules within the context of a multi-agent team simulation. From this study a general approach to modeling an airline market has been developed based on a multi-agent, team-based concept. Additionally, several preliminary trials of the simulation have been executed and the results are reported within.

1. Introduction

Everyone who travels by air has experienced chronic delays and flight cancellations while traveling via the 'Friendly Skies.' Each day during the twin peak periods, which occur during the morning and evening rush hours, millions of passengers simultaneously descend upon our nation's airports. This leads to the long lines and delayed flight schedules to which we have all become accustomed. This research project is motivated by the Federal Aviation Agency's (FAA) interest in helping the Local Airport Authorities (LAA) mitigate these problems. Traditionally, the LAAs charge airlines a flat rate to land and depart from their airports regardless of a flight's arrival or departure time. In an attempt to encourage airlines to schedule their flights evenly throughout the day the LAAs are interested in experimenting with variable pricing policies. These policies would provide the proper economic incentives to airlines so that they will distribute their flights evenly over the course of the day.

This study investigates what the response of the individual airlines would be to a proposed pricing policy. The aim of this project is to model the decision making process of the airlines accurately using a multi-agent model, in which each airline is modeled by a cooperating set of teams (i.e., a "team-of-teams") that learn to work in concert to compete effectively against other similar airline teams. Discussed in this paper is a description of the task environment, multi-agent architecture, the learning algorithms, preliminary results, conclusions, and future directions of this project.

2. Task Environment and Problem Statement

The task environment is comprised of two relatively independent entities—passengers and local airport authorities. In this simulation, passengers are modeled by a "passenger demand allocation model" using utilities that are a function of the flight's time and fare. The LAAs have the ability to charge the airlines a fee for each flight departure and arrival. Currently, this is a fixed usage fee that remains the same over the course of the day (Figure 1). Historically, passengers have preferred overwhelmingly to travel during two periods of the day: morning and early evening. This phenomenon is also seen in the morning and evening ”rush hour” commutes of daily workers. Because the cost for an airline remains the same, regardless of the flight’s departure or arrival time, fares do not necessarily fluctuate according to its time slot. Therefore, passengers do not receive any benefits for traveling during off-peak hours. This has lead to the development of airport ”rush hours,” which are the source of flight delays and passenger frustration throughout the airline system.

3. Multi-agent Model Structure

This project aims to predict the behavior of the individual airlines when given an airport cost function, with values that vary with respect to the flight’s arrival/departure time; but once established does not change over the course of the simulation. A multi-agent simulation is developed to allow airlines to compete in a simulated market for passengers.

The proposed agent model consists of several teams comprised of learning agents that we call ”Flight agents,” each representing a single flight (departure or arrival). Flight agents are logically grouped into teams representing the specific routes that the airline services (route-teams). Each airline will consist of several route-teams of learning agents that must cooperate with each of its route-team members. Additionally, the set of route-teams as a whole must learn to cooperate with each other to optimize overall airline profitability. Furthermore, these airline teams must also learn to compete against similarly structured airline teams for passenger market share. As shown in figure 1 an Airline agent is defined as a static agent that merely serves as a bookkeeper; and provides an interface between the environment and the Flight agents. Each Flight agent contains its own schedule that consists of the Fare, Time, and Capacity of the flight. Collectively, the Flight agents represent the Airline’s flight schedule for that airport.

Each flight of an airline’s flight schedule is evaluated in a simulation environment by the Passenger Demand Allocator Model (PDAM). The PDAM assigns a utility value to each of the flights and allocates passengers to these flights based on their relative utility values.

4. Learning Algorithm

For this initial study each agent will use a Learning Classifier System [1] as its learning algorithm. An LCS was chosen because it is a rule based learner, and the resulting rule set can be used to determine the ”quality” of the rules evolved by the agents. Specifically, the LCS algorithm that will be used is XCS [2–5], a variant of Holland’s original LCS.

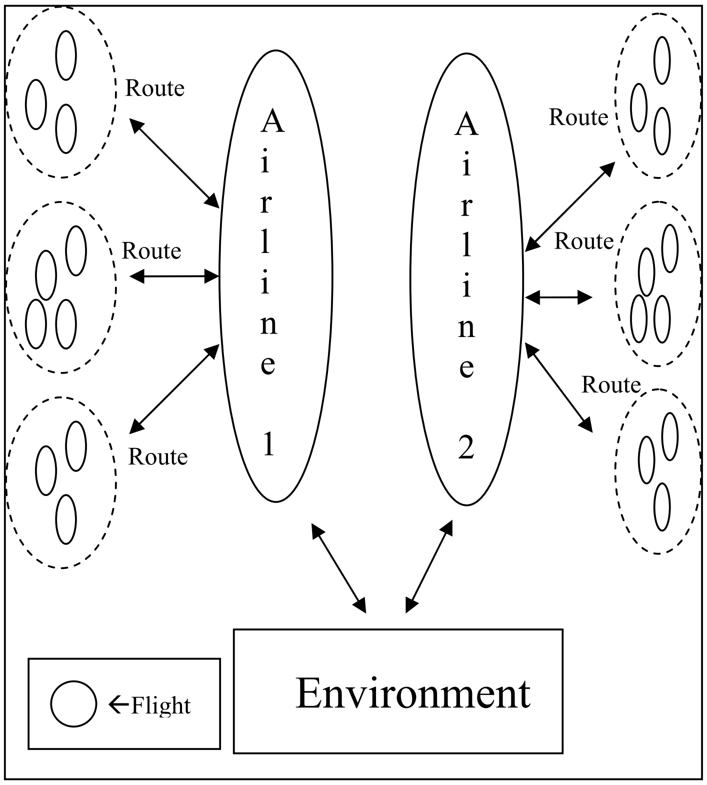

Fig. 1. *Multi-agent architecture for airline scheduling*

4.1 Representation

LCSs use a set of rules in which each rule consists of a condition-action pair. The condition of each rule is matched to the current state of the environment and competes to have its action executed in the environment. The conditions can be represented as a simple conjunction of terms, or as complex structures such as trees. In this study a simple conjunction of attributes will be used. The attributes that will form the representation of the rule's condition will are listed below (Note: each field contains a single discrete value).

Definitions

Pax →	Passengers
Seats →	Seats available (capacity)
Seat Mile →	Seats/Mile
Load Factor (LF) →	Flight Load Factor (LF_f) and Airline Load Factor (LF_a)

$$LF_f = \frac{Pax}{Seats\ Available} \qquad LF_a = \frac{\sum_{\forall f} \left(LF_f \times D_f \right)}{\sum_{\forall f} D_f} \tag{1,2}$$

Revenue per Available Seat Mile (RASM) → "Revenue per Seat per Mile"
Cost per Available Seat Mile (CASM) → "Cost per Seat per Mile"
Yield → "Revenue per Passenger per Mile"

$$RASM = \frac{Revenue}{Seats \times Distance} \qquad CASM = \frac{Cost}{Seats \times Distance} \tag{3,4}$$

$$Yield = \frac{Revenue}{Pax \times Distance} \tag{5}$$

f, *r*, *a*, subscripts → Flight, Route, and Airline respectively.

4.2 Condition Attributes

The attributes that will comprise the conditional for each rule can be divided into five categories: Profitability, Load Factor, Market Share, Price, and Time.

Profitability

Profitability is measured by the difference between the revenue per seat mile (*RASM*) and the cost per seat mile (*CASM*). These attributes are quantized into Boolean values of true (1) or false (0) in which a true value is assigned for a positive result; otherwise, a false value is assigned.

- Is the airline profitable? (true/false) [$PASM_a = RASM_a – CASM_a$]
- Is the route profitable? (true/false) [$PASM_r = RASM_r – CASM_r$]
- Is the flight profitable? (true/false) [PL = (Pax)(Fare) – Costs]
  - *Costs* = Airport fees and static fee to operate a plane of the given capacity.

Passenger Load Factor

The load factor will range continuously from –1 to 1 when the difference between two load factors is computed. Then each attribute value is quantized into integer values between –2 and 2. The first comparison of load factors considers the difference between the flight load factor and the aggregate load factor for all of the flights within the same route and airline as the current flight. If the difference is positive the flight's load factor is higher than the average load factor for that route. The rest are listed below.

- How much greater(or less than) is the load factor for the flight, LF_f, compared to the aggregate load factor across the market for this route, $LF_{r,m}$?
- How much greater(or less than) is the load factor for the flight, LF_f, compared to the aggregate load factor across the entire market, LF_m?

Market Share

Market Share is quantized into integer values ranging from 1 to 5.

- The percentage of the airline's total passengers that is provided by a specific route. [0, 1]
- The share of the market that airline, *a*, has in route, *r*. [0, 1]
- The share of the market that airline, a, has with respect to the total airline market, m. [0, 1].

Fare Factor (Price)

The fare for each flight will be calculated on an available seat mile basis (FASM), and quantized into integer values of 1 through 5. The attributes pertaining to fare are as follows:

$$FASM_f = \left(\frac{RASM_f}{Pax_f} \right) \qquad \frac{FASM_f}{FASM_{r,m}} = FareFactor_{f|r,m} \tag{6,7}$$

- Is the fare for the flight greater (less) than the average fare for all flights on this route across the market?
- Is the fare greater (less) than the fares across the entire market?

Time

The travel day is one continuous 20 hour time block beginning at 5 a.m. and ending at 1 a.m. the next morning. This interval will be initially divided into eighty 15 minute time slots during which an airplane can depart.

- What is my current time slot? [1 – 80]
- How many time slots away is the next flight departure? [0 – 79]
- How many time slots away was the last flight departure? [0 – 79]

Internal Flight State

Each Flight agent has an internal state that the actions listed below modify. The internal state of the Flight agent is its fight schedule: time, fare, capacity, and whether it will fly or not.

4.3 Action Set

Each rule will have one macro-action that is a composite of a set of actions chosen from the following action categories.

- Increase fares by a certain percentage (ex. +20%, or – 15%)
- Shift time slot by ± N Time Slots
- Increase/Decrease Plane Capacity
- Flight occurs? (Yes/No)

For example, one possible macro-action might consist of: Fare = Increase; Time Slot = Unchanged; Capacity = Decrease; and Fly? = True

Increase Fare and Time Slot Actions

An individual flight can increase its current fare by a predetermined average percentage. The mean percentage change and standard deviation are specified prior to starting the simulation. If the fare increase (decrease) action is chosen a random value is then drawn from the Gaussian distribution centered around the desired mean fare percentage change, and the current fare is changed by the random amount. It is believed that this fare modification algorithm will allow for fares to exist over a continuous range; instead of a range of discrete fares.

Time Slot modification actions, although integer valued, operate in the same manner as the fare modification algorithms. However, the mean distance (in number of time slots) to move and standard deviation are specified, instead of the percentage increase.

Flight Capacity Action

A flight's capacity is dictated by the type of plane it is assigned. For example if the flight is assigned a 737 it can hold 150 passengers. If a flight decides it needs more capacity, a bigger plane, it can request the next largest plane size, such as a 757. To add additional realism each airline is assigned a static fleet of airplanes from which to choose. Meaning, that if a flight wants more capacity and there are no larger planes available in the fleet then the flight's request is denied, and it remains at the current capacity.

Will Fly? Action

This action will remove the flight from the simulation and it will not be included in the airline's schedule and evaluation. Potentially, an airline can stop servicing a route if it cannot find a profitable way to compete against it's competitors in that market.

4.4 Reward

The reward given to a particular flight will be a linear combination of the profitability of the individual flight, the route, and the airline. The coefficients will be pre-determined and in future studies could be learned as well.

$$Reward_f = C_f(PASM_f) + C_r(PASM_r) + C_a(PASM_a) \tag{8}$$

This reward system is designed to encourage cooperation between flights within a route and across the airline. Suppose an individual flight is unprofitable, and the availability of that flight is key to achieving higher profits across the entire airline then the reward system will properly reward that flight and keep it operational.

5. Experimental Design

Task Environment

The objective of the airline teams in this experiment is to maximize the aggregate airline profit, as measured by $PASM_a$. Passengers are randomly assigned to each flight in proportion to the utility it provides to passengers, $U_{flight}(fare,\ time)$. A flight's utility is a function of both its fare and time of departure. In this simulation all passengers have the same objective, which is to pay the lowest fare, and fly as close to the 'peak' time slots as possible. Additionally, all flights are viewed as commodities such that passengers are indifferent to the airline on which they travel. The passenger's utility function is simply a linear combination of the flight's normalized fare and time slot.

$$U_{flight}(fare, time) = C_f \times U_{fare}(fare) + C_t \times U_{time}(time) \tag{9}$$

The relationship between the flight's fare utility is inversely proportional to the actual fare; and the time slot utility is also inversely proportional to the time interval separating the flight's departure time and the closest peak time. The values of C_f and C_t for this study are both 1.0; equally weighting the contributions of the fare and time slot utilities.

When these two functions are combined they create a multi-dimensional passenger utility search space. Flights must find the proper time slot and fare that will maximize the utility it provides to its passengers and maximize its own profitability.

It is expected that by varying the airport costs as a function of a flight's departure time it will allow the airlines to offer flights at time slots that fall within the "valleys" at lower prices. The utility gain from the lower fare will offset the loss of utility for departing at a less than desirable time (from the passengers' perspective).

Passenger Demand Allocation

Because the objective of each passenger is to maximize his or her expected utility, passengers are assigned to flights stochastically in proportion to the flight's utility.

Once a flight reaches its capacity, the remaining passengers are assigned to other flights based on the relative utilities of the remaining flights. This form of roulette wheel selection provides a fairly linear mapping from passenger utility to expected passenger demand for a particular flight.

Flight Profitability

The profitability of a flight is defined in the normal manner; revenues minus costs. For the airline domain all profit, revenue, and cost values are given in terms of dollars per available seat mile ($/ASM) as described above. A flight's revenues are defined as the product of the total number of passengers onboard and the fare; and its costs are separated into fixed and airport related costs. The two cost categories are described below.

Fixed Costs

A flight's fixed costs are a function of the plane's capacity. In this simulation a schedule of fixed costs versus capacity was created and is described below. Additionally, three classes of carriers were defined as well: low-cost, average-cost, and high-cost. The airlines used in this experiment are all average cost carriers.

Carrier Fixed Cost Schedule

Plane Type	Capacity (Pax)	Low Cost Carrier	Avg. Cost Carrier	High Cost Carrier
717	100	7,500	10,000	12,500
737	150	11,250	15,000	18,750
757	225	16,875	22,500	28,125
767	300	22,500	30,000	37,500
777	350	26,250	35,000	43,750
747	400	30,000	40,000	50,000

Airport costs

In this experiment the airport costs are held constant at 0.03$/ASM. At this rate it will cost an airline $1,350 to operate one 737 in the airport's facilities.

Flight Reward

The reward that each flight receives after selecting its flight schedule is calculated according to the equation (8) in section 4.4. The coefficients chosen for this experiment are listed below. Note that C_a is set to 0.0 because there is only one route, so there will not be a contribution from additional routes.

Reward Coefficients

Coefficient Parameter	Value

C_f	1.0
C_r	0.25
C_a	0.0

Agent Model Experiment Structure

For this experiment, two airlines were created that compete in the same 200 mile route from La Guardia Airport (LGA) to Reagan-National Airport (DCA) with a maximum of two flights per airline. Each airline has an identical fleet and fixed cost structure. The fleets are composed of 2 of each type of aircraft from the 717 to the 747; and each airline is an average cost carrier (as defined above). This allows the flight agents to choose any combination of capacities they desire. In future experiments additional airlines and routes will be added as well as varying each airline's cost structure.

XCS Learning Algorithm Parameters

The individual flight agents have at their center an XCS (rule) learning algorithm. Using the representation previously defined, each flight uses the same set of parameters defined below. Definitions of these parameters can be found in [2].

Learning Parameters

Parameter	Value
Maximum rule set numerosity (N)	5,000
Initial rule set state	Empty
Rule set deletion threshold (Θ_{del})	15
Rule set delta (δ)	0.1
Rule set minimum # of actions (Θ_{mna})	25
Probability of "Don't Care" ($P_{\#}$)	0.20
Probability of mutation (μ)	0.02
Probability of crossover (χ)	0.95
Probability of selecting a random action (p_{explr})	0.25
Initial fitness (f_I)	2.5
Initial prediction (p_I)	2.3
Initial error (ε_I)	0.5
Initial experience (exp)	0
Initial action set size (as)	1.0
Initial numerosity	1
Discount factor (γ)	0.71
Do GA subsumption?	No
Do action set subsumption?	No
Subsumption threshold (Θ_{sub})	N/A
Learning rate (β)	0.3
Error threshold (ε_0)	0.1

Power parameter (ν)	5
α	0.1
GA Threshold (Θ_{GA})	10

Market/Environment Parameters

In the table below the remaining simulation parameters are listed.

Parameter	Value
Total Passenger Demand	500
Time Slot Peak Times	16 and 52 (9am and 6pm)
Number of simulation epochs	30,000
Number of experiment trials	1
Minimum Fare	$100
Maximum Fare	$750
Change Fare action mean % change	5.0%
Change Fare action stand. dev ($\sigma_{fare\ action}$)	0.5
Change Time Slot action mean slot change	4
Change Time Slot action std. dev ($\sigma_{time\ action}$)	0.5

6. Results

It was expected that the flights would migrate to one of the two ridges and find the optimal fare along the peak of one of the ridges. Additionally, it was expected that the profitability and market share of the two airlines would be approximately equivalent. This, however, was not the case. The charts below indicate that Airline 1 gained 80% of the market share and was almost twice as profitable on an available seat mile basis; and three times as profitable in absolute terms. Of additional concern is that both airlines chose to fly at very unattractive time slots in the early morning and during the last time slot available. Airline 1 did schedule one flight in time slot 13, which is only 3 slots away from a peak at slot 16 (56 is the other peak time slot).

Aggregate Airline Statistics

	Airline 1	Airline 2
Total Capacity	450	250
Total Passengers	400	100
Market Share	80%	20%
Load Factor	0.888	0.5

Available Seat Miles	180,000	100,000
PASM	0.83974	0.494
Profit	$150,295.33	$49,400.00
Average Fare (2 flights each)	$496.71	$750.00
Times Slots Flights Flew	13, 79	5, 79

From a qualitative perspective a detailed look at a random sampling of the rules produced by the four flights indicate that very few of the rules are actually being selected to be part of an action set, which is indicated by the very low figures for the rules' experience values. The average rule has an experience of about 2; and the few rules that did get selected frequently as members of action sets were unable to improve significantly on its initial prediction, error, and fitness. The existence of rules such as these indicates the need to include GA and Action Set subsumption in the algorithm to help condense the population of rules and quickly force rules like this one out of the population. XCS has a bias towards driving inaccurate (less fit), experienced rules out of the population. Meaning that in order to be deleted from the population a rule most be sufficiently experienced and unfit relative to the mean fitness of the population [6,7]. This is accomplished by using a fitness sharing scheme to calculate a rule's fitness.

A rule's numerosity is increased whenever the XCS determines that one rule's conditional is more general than another rule – and prescribes the same action. The more specific rule is deleted from the population and the more general rule's numerosity is increased [2]. When the fitness of a rule is calculated it is divided by its numerosity which effectively shares the fitness across all of the rules it has subsumed. This will work to drive the fitness of general, inaccurate rules rapidly out of the population. In this experiment subsumption was not turned on as it is not needed in all applications [2]. However, after studying the rule sets generated both action set subsumption and GA subsumption would help to increase performance.

7. Conclusions

An experiment of this nature involves a great deal of moving parts and parameters. The most immediate concern is to fine-tune the XCS learning algorithm to find a suitable combination of settings. Improving the ability of the agents to perform well in this environment will allow for larger, more complex simulations. Preliminary runs with as many as 3 airlines, 3 routes, and 4 flights in each route for each airline (36 agents) showed similar results as above, and took several days to complete. Because no simulation has ever converged to an 'optimum' solution it is unclear whether the simulation requires more training epochs or that the parameters are set incorrectly. Once, the answer to that question has been answered scaling this simulation in complexity will be possible.

8. Future Work

Future work to follow this study includes investigating various parameters in the XCS algorithm in order to see how these changes affect performance. Once acceptable performance is realized on the simplified environment in this study a more complex passenger demand model and airport cost function will be integrated into the environment.

Additional learning algorithms other than learning classifier systems must be investigated for various elements of the simulation. A long term goal is to not only predict airline behavior in response to a given airport cost function, but to also learn the optimum airport cost function that will maximize any one (or more) of a set of objectives (e.g. reducing delay at airports).

As additional airports, airlines, routes, and flights are added scalability will become a critical issue. An advantage of this autonomous multi-agent architecture is that as more agents are required additional processing components can be added as well. However, research needs to be done on how best to distribute the population of agents over the computing resources. Of particular interest is the emerging Grid Computing standard that would allow agents in our model to be efficiently distributed across a wide range of processing elements.

9. References

[1] J. H. Holland. Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems. Reprinted in G. F. Luguer, editor, *Computation & Intelligence: Collected Readings*, pages 275 – 304, Cambridge, Mass. 1995. MIT Press.

[2] Wilson, S. W., Butz M. V. An Algorithmic Description of XCS. *Lecture Notes in Computer Science*. Vol. 1996, pp. 253+. 2001.

[3] Butz, M. V., et al. How XCS Evolves Accurate Classifiers. In Spector, L. et al (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference ({GECCO}-2001)*. pp 927 – 934. San Francisco: Morgan Kaufmann. 2001

[4] Tharakunnel, K. K., Butz, M. V. XCS: Accuracy and Successful Generalizations. IlliGAL Report No. 2002022. University of Illinois at Urbana-Champaign. December 2002.

[5] Butz, M. V., et al, Theory of Generalization and Learning in XCS. IlliGAL Report No. 2002011. University of Illinois at Urbana-Champaign. May 2002.

[6] Wilson, S. W. Classifier Fitness Based on accuracy. *Evolutionary Computation,3* (2), 149 – 175. 1995.

[7] Wilson, S. W. Generalization in the XCS Classifier System. In Koza, et al (Eds.), *Genetic Programming 1998: Proceedings of the Third Annual Conference* pp. 665-674. San Francisco: Morgan Kaufmann. 1998.