Robustness of Two Air Traffic Scheduling Approaches to Departure Uncertainty

Adrian Agogino, UCSC, NASA Ames Research Center, Moffett Field, CA
Joseph Rios, NASA Ames Research Center, Moffett Field, CA

Abstract

Linear programming methods and non-linear, evolutionary algorithm-based optimization techniques have been shown to be effective in managing large-scale air traffic flow problems. However, many of these algorithms assume perfect knowledge therefore the robustness of these algorithms in the presence of uncertainties is questionable. Since real-world application of these methods require them to be effective under uncertainty (i.e. produce few unexpected capacity violations), it is critical that they are tested in such conditions. In this paper we test the effectiveness in the presence of uncertainty of a binary programming approach and a novel, fast-learning evolutionary algorithm. Specifically we change the assumed takeoff times on which these algorithms are trained, and test the resulting solutions when takeoff delays that are consistent with historical data are incorporated. Experimental results show that without uncertainty, both sets of algorithms are able to quickly produce solutions with few to no violations. In the presence of uncertainty, the performance of the algorithms degrade with respect to the amount of delay added, but are still very good. Even when uncertainty is extremely high, the expected delay is never increased more than 30%.

1 Introduction

Traffic Management is defined by the FAA as "the craft of managing the flow of air traffic in the [National Airspace System] based on capacity and demand."[1] To pose this statement as a question: Given a set of capacities for various resources (e.g., en route sectors or airports) in the National Airspace System, which flights should be held where and for how long in order to satisfy all demand-capacity imbalances? The solution space for this problem is massive when one considers the number of flights (thousands in the air, thousands more scheduled to depart in the coming hours) and the number of shared resources (sectors, airports, merge points, fixes, etc.) within the NAS. The methods presented here and throughout the traffic flow management literature depart from these traditional ways of controlling flows of traffic including Ground Delay Programs, Airspace Flow Programs, or Miles-in-Trail Restrictions. Specifically, this paper uses the binary integer programming approach originally presented by Bertsimas and Stock-Patterson [2] to determine an optimal solution to the above question. In addition, we build upon the work of Rios and Lohn [3] in the use of evolutionary approaches to solving the same problem. The most current and complete overview of TFM in practice and in research is provided by Sridhar, Grabbe, and Mukherjee [4].

There are several studies in the literature examining traffic management under uncertainty with this subdomain being called "Probabilistic Traffic Flow Management" in the literature. The work is divided into uncertainty in capacity estimation [5, 6, 7], demand estimation [8, 9, 10, 11], or simultaneous demand and capacity uncertainty [12, 13]. The work presented here focuses on airport departure uncertainty as quantified by Mueller and Chatterji [14] and its intersection with the optimization approaches described above. The merging of optimization approaches with demand uncertainty measurements. Much of the work involving stochastic optimization of Traffic Flow Management focuses on capacity uncertainty (see, for example, Nilim and El Ghaoui [15] or [16]) rather than the uncertainties involved with flight trajectories and departure times. The approach presented here looks at how the performance of a solution found with deterministic demand inputs de-

grades as departure uncertainty increases.

The remainder of this paper is organized as follows. The next section describes the binary integer programming approach to solving demand-capacity imbalances. Then in Section 3 a new evolutionary algorithm approach to solving the same problem is presented. Following that, Section 4 describes the departure uncertainty model. Experiments and results are presented in Section 5 followed by conclusions in Section 6.

2 Optimization Approach to Traffic Flow Management

In this paper, a binary integer programming (BIP) model [2] is used to perform scheduling that minimizes delay costs. The model as presented by Bertsimas and Stock-Patterson is given here, but for a more detailed description of the model, the reader is directed to their original paper.

The decision variables, $w^j_{f,t}$, are valued 1 if and only if a flight f has entered sector j by time t and 0 otherwise. The set of variables for a given flight-sector pair can be seen as a step function from 0 to 1 with the change occurring at the first time the flight enters the sector which is enforced by Constraints 8. The objective function (1-3) is an expression for describing the sum of weighted air and ground holding costs (c^a_f and c^g_f, respectively) over all flights.Typically the problem is modeled with $c^g_f < c^a_f$ as the preference is to hold flights on the ground rather than in the air. Qualitatively, the objective function simply says to minimize the sum of air and ground delay over all flights where air delay is more costly.

For each flight, the model is able to decide if the flight needs to be held anywhere (and for how long) in order to satisfy the airport departure, airport arrival, and sector capacity constraints (constraints (4), (5), and (6), repsectively). Airports are denoted by k and sectors by j. Each of the capacities is a function of time, t. Each flight in the set of flights, $\mathcal{F}$, is described as an ordered list of distinct sectors, j, from a set of sectors, J, with earliest and latest feasible entry times for each of those sectors. For modeling purposes, airports are considered a subset of sectors with associated arrival and departure capacities. A sector in a flight path is denoted by $P(f, y)$, where f is the flight and y is the ordinal representing its place in the flight path. For ease of notation, $P(f, \text{last})$ is used to represent the last sector (usually an airport) in f's path.

$$\text{Min} \quad \sum_f \Bigg[(c^g_f - c^a_f) \sum_{t,k=P(f,1)} t(w^k_{f,t} - w^k_{f,t-1}) \tag{1}$$

$$+ c^a_f \sum_{t,k=P(f,\text{last})} t(w^k_{f,t} - w^k_{f,t-1}) \tag{2}$$

$$+ (c^a_f - c^g_f)d_f - c^a_f r_f \Bigg] \tag{3}$$

Subject to:

$$\sum_{f:P(f,1)=k} (w^k_{f,t} - w^k_{f,t-1}) \leq D_k(t), \tag{4}$$

$$\forall\, k \in \text{Airports}, t \in \text{Time}$$

$$\sum_{f:P(f,\text{last})=k} (w^k_{f,t} - w^k_{f,t-1}) \leq A_k(t), \tag{5}$$

$$\forall\, k \in \text{Airports}, t \in \text{Time}$$

$$\sum_{\substack{f:P(f,i)=j,\\ P(f,i+1)=j'}} (w^j_{f,t} - w^{j'}_{f,t}) \leq S_j(t), \tag{6}$$

$$\forall\, j \in \text{Sectors}, t \in \text{Time}$$

$$w^{j'}_{f,t+\mu(f,j)} - w^j_{f,t} \leq 0 \tag{7}$$

$$\forall\, f \in \mathcal{F}, j = P(f,i), j' = P(f,i+1)$$

$$w^j_{f,t} - w^j_{f,t-1} \geq 0 \tag{8}$$

$$\forall\, f \in \mathcal{F}, j \in (f\text{'s flight path})$$

$$w^j_{f,t} \in \{0,1\} \tag{9}$$

$$\forall\, f \in \mathcal{F}, j \in (f\text{'s flight path}), t \in \text{Time}$$

The problem is also constrained by the physical and temporal limitations of the flights. Specifically, each flight spends, at least, the specified minimum sector time, $\mu(f, j)$, in each of its sectors, j as described by constraints (7). Coupled with the set of constraints enforcing the logic of the variables, i.e. all 0's before all 1's as discussed above (constraints (8)), physical and temporal logic is satisfied.

3 Evolutionary Algorithm Approach

Evolutionary algorithms [17] are inspired by nature, where organisms of low fitness do not survive and organisms with high fitness thrive and reproduce. Over

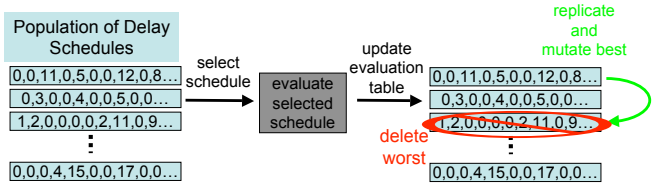

Figure 1: Simple Evolutionary Algorithm. A population of delay schedules is evaluated based on lateness and constraint violation penalties. Afterward the worst performing member of the population is eliminated, and replaced with a mutated version of the best member of the population.

time the expectation is to have a population with an increasing number of high-fitness organisms. In this paper, instead of evolving organisms, we evolve a population of delay vectors, which we call chromosomes. We then assign a fitness to each chromosome with a fitness evaluation function. There are many possible fitness evaluation functions, but in general we are interested in ones that penalize chromosomes representing delay schedules that lead to high overall delay and violate constraints. Using an evolutionary algorithm we can then eliminate chromosomes with low fitness evaluations and replicate chromosomes with high fitness evaluations (see Figure 1). In principle with enough time, the evolutionary algorithm will produce a population of chromosomes that will have high fitness, therefore each chromosome will represent a delay schedule that has low overall delay and violates few constraints. If not violating constraints is critical, we can set the fitness penalties for constraint violations to very high levels.

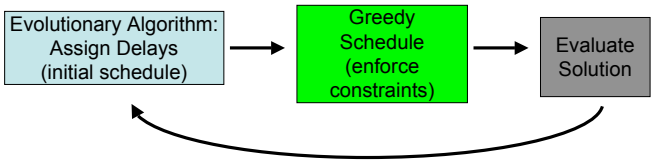

Figure 2: Evaluation using greedy scheduler.

Full Chromosome Evolution with Greedy Scheduler

While in principle an evolutionary algorithm could produce a solutions that met all the constraints, in practice it would take a very long time as most solutions would violate the constraints. As an alternative we take the solution provided by the algorithm and then use a greedy scheduler to maintain the constraints [3]. The greedy scheduler simply takes each aircraft in order of departure time and delays it the minimal amount to avoid overloading any constraints. With this system, later flights are more likely to be delayed than earlier flights since there is less capacity available by the time the later flights get their "turn." We then evaluate a chromosome based on the final schedule produced by the greedy scheduler (seeded with the chromosome's solution). Therefore, an initial schedule represented by chromosome, x, is evolved using the evaluation function:

$$G(x) = \sum_i S(x_i) \ , \tag{10}$$

where, $S(x_i)$ is the delay assigned by the greedy scheduler for aircraft i, given the initial delay of x_i.This process is shown in Figure 2. While the output of the greedy scheduler does not violate constraints, the greedy scheduler will add delay. The task of our evolutionary algorithm is to choose initial delay schedules, such as to minimize the delay of the schedule produced by the greedy scheduler.

Even though with the use of the greedy scheduler, the task of the evolutionary algorithm is simpler, it can still suffer from scaling issues. In the application used in this paper, the scheduler has to schedule more than 5,000 aircraft at one time. This results in a chromosome with more than 5,000 elements. Since mutations are performed randomly across the chromosome, a mutation will likely add delays to some aircraft that are helpful to the system, but at the same time will add delays to other aircraft that are unhelpful. In fact, since the vast majority of aircraft do not have to be delayed at all, most delay assignments will be harmful. Given the size of the chromosome, it will take many rounds of evolution, before a new chromosome is produced that is significantly better than existing ones.

Component Evolution

We propose to address the issue of problem size by having a separate population of delays for each aircraft (see Figure 3). Each member of each population contains a chromosome of length 1, that specifies the delay only

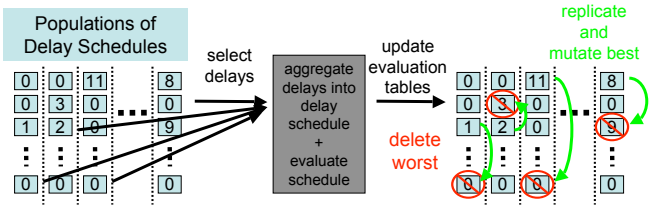

Figure 3: Multi-Population Evolutionary Algorithm. Each aircraft has its own population of delays. Each delay is selected independently and aggregated to form a delay schedule. Based on the performance of this schedule the delays are then evaluated independently and each population is updated independently.

for a single aircraft. Each population has its own evolutionary algorithm that selects members independently from the other populations, and evaluates each member of its population separately. The main issue that now has to be resolved is how do we evaluate each individual delay, when the values of these delays only have meaning within the context of the entire delay schedule? The simplest way to handle this issue is to simply evaluate each individual delay using the evaluation function for the entire delay schedule. However, this approach has similar scalability issues that the simple evolutionary algorithm has. Suppose a delay choice for one aircraft out of 5,000 happens to be good, if it receives an evaluation that is a function of all 5,000 delays, it is likely that many of the other delay choices will be poor, and this good choice will not get proper credit. While on average good choices should get slightly higher evaluation than bad choices, it will take a very long time to converge.

Instead of evaluating each delay using the system evaluation, we propose to evaluate them using the difference evaluation:

$$D_i(x) = G(x) - G(x - x_i) \ , \qquad (11)$$

where $x - x_i$ is the delay schedule for all the aircraft with the delay of aircraft i set to zero [18, 19]. Intuitively the difference evaluation measures the contribution of delay i to the system as a whole. The main advantage of this evaluation is that the subtraction removes a lot of the uncertainty caused by delay assignments from aircraft that are not relevant to aircraft i. The main disadvantage of this approach is that the counterfactual $G(x - x_i)$ has to be computed for each aircraft. Luckily for most aircraft there is no delay ($x_i = 0$). Therefore, the value of D_i can be trivially computed as 0, since in this case $G(x)$ is equal to $G(x - x_i)$.

4 Uncertainty Model

In this paper our algorithms produce a delay vector x assuming an exact departure schedule z. However, in general the exact time that aircraft will depart in the future will not be precisely known to the algorithms. Instead the true departure times will be related to their schedule departure time plus some amount of uncertainty. If our algorithms produce a delay vector x assuming a schedule of z when the actual schedule is z', this delay vector may be suboptimal and could result in producing a schedule that has significant violations.

To test the robustness of the BIP solutions and the evolutionary algorithm solutions to this uncertainty, we perform a number of tests were we add uncertainty to the departure schedule. This uncertainty model comes from analysis of historical departure data done by Mueller and Chatterji [14]. Using this model we add uncertainty to each departure time, using the Mueller and Chatterji's normal distribution with mean 3.91minutes and a standard deviation of 16 minutes. Therefore for each flight i, the new departure time is equal to:

$$z'_i = z_i + N(3.91, 16) \ . \qquad (12)$$

To produce the final departure schedule the times from the uncertain departure schedule are added to the delays.

5 Experimental Results

Deterministic Performance

We test two different algorithms to evolve the chromosome. The goal of the algorithms is to produce the highest performance delay schedule, e.g. the schedule that minimizes the total delay. The first algorithm is a simple evolutionary algorithm with no cross-over and a population of ten chromosomes. In the second algorithm each individual component evolves separately with a simple learning algorithm that tries to maximize a component specific evaluation function. The evaluation function for component i is in the following form: $D_i(x) = G(x) - G(x'_i)$ where x'_i is produced by replacing the delay value for component i with zero. This evaluation produces a "difference evaluation" measuring the

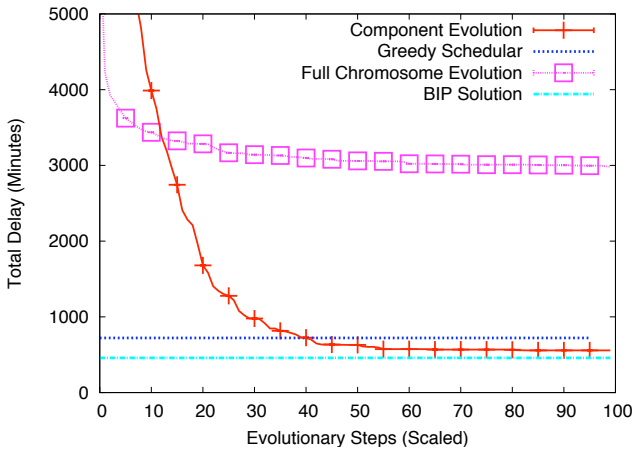

Figure 4: Performance of Evolutionary Algorithms. Component evolution performs better than full chromosome evolution, though still short of optimal. Note number of evolutionary steps is "scaled" so that at each step, each algorithms has called the greedy scheduler the same number of times, so at each step each algorithm has used approximately the same amount of total computation.

difference between the performance of system achieved by its choice of delay and the performance achieved by choosing a delay of zero. This can be seen as measuring the contribution to system performance for that component [19].

We perform experiments showing the performance of component based evolution and traditional evolution. In all experiments, an initial set of flight plans for 5,910 aircraft is used, covering a 60 minute time period. Both evolutionary methods produce a set of initial delays that are then processed by the same greedy scheduler to produce a final solution. Note that even though the initial solution only presents ground delays, the greedy scheduler may add air delay for aircraft already in the air. Final solutions are then evaluated based on the amount of delay generated where air delay is penalized twice ground delay. To keep higher values better, the evaluation function is the negative of the total delay penalty. In the experiments, approximately 70,000 total evaluations were performed to create the final solution (representing 70,000 runs of the greedy scheduler). The vast majority of computation is spent in these evaluations.

The results show that evolving the full chromosome at once produces poor results (see Figure 4). This is not surprising, because the search space is so large. Since relatively little delay is needed to meet the constraints, most solutions produce vastly too much delay. In fact a solution is not even found that is better than a chromosome composed of zeros, which ultimately creates a final delay penalty of 722 once it is run through the greedy scheduler (we also tested the case when a member of the initial population set to zero, but no solution better than the zero vector was ever produced). In contrast, component evolution performs well. Component evolution evolves quickly and produced a final solution within 20% of optimal. This algorithm is able to perform better, since each component is able to optimize its own best delay using an evaluation function that highlights its contribution to the system evaluation. By the end of evolution, most components find that their best solution is to have zero delay, while the rest are able to optimize their delay choices.

Robustness to Uncertainty

While both the BIP approach and evolutionary algorithm perform well when the departure times are exactly known, we would like to know how well their solutions work when the departure times are slightly different than the algorithms expected them to be. To analyze this we run a series of experiments where we add uncertainty (using the model discussed in Section 4) to the departure schedule. In these tests, we look at the difference in the performance of the BIP solution and the evolutionary algorithm solution when they are used in a deterministic environment compared to when they are used in an uncertain environment. In all experiments thirty trials are run with different random number seeds use for adding uncertainty. The results are all an average of these thirty trials, with the error bars in figures representing one standard deviation from the mean.

In our first test we ran the BIP solution and the evolutionary algorithm solution, on uncertainty coming from the distribution:

$$z_i' = z_i + aN(3.91, 16) \ , \tag{13}$$

where a controls the scaling. At 100%, a equals 1 and the uncertainty distribution is equal to the uncertainty coming from historical data (see Equation 12). While the solutions provided by the algorithms produce zero capacity violations when demand is deterministic, these

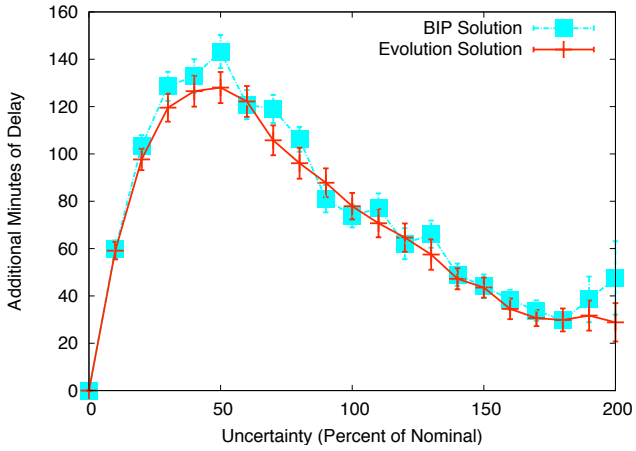

Figure 5: Performance Sensitivity of BIP and Evolutionary Algorithm to Uncertainty. Unlabeled graphs represent alternative runs of evolutionary algorithm with different random number seeds.

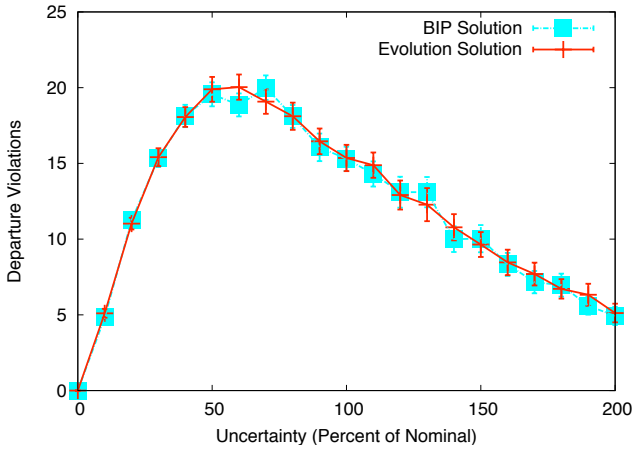

Figure 6: Departure Violations Caused by Uncertainty for the BIP and Evolutionary Algorithm.

solutions may produce results that lead to capacity violations when used with uncertain departure times, since these departure times are slightly different than what the algorithms produced their solutions for. In our first set of tests we eliminate all of the capacity violations by running the solution through the greedy schedular. While the greedy scheduler removes all the violations, it also adds delay to the system. Figure 5 shows the additional delay added to the system as a result of the greedy scheduler removing violations caused by the uncertainty. The results show that both the BIP approach and the evolutionary algorithm handle uncertainty remarkably well. Given the implemented uncertainty model, the amount of delay added is 30% in the worst case. However, with uncertainty values consistent with historical data the amount of additional delay falls to 18%. In fact, surprisingly the amount of delay actually goes down when the uncertainty level become extremely high. This can be attributed to the high level of uncertainty smoothing the system, as more uniformly random departure times produce less conflicts than true departure times since natural peaks in the departure schedule are removed. This result shows that we can have confidence in all of the solutions, even if the actual departure times for aircraft are not exactly what we expect. A similar phenomenon has been noted by Gilbo and Smith [9] in their study of uncertainty affecting sector violations.

We also test the performance of our algorithms where the greedy scheduler is not used to remove the capacity violation. This models the situation where global schedules are less flexible and, therefore, less likely to be altered once implemented. In this case, since no new delay is added to the system, we are concerned with how many violations the delay schedules lead to in the presence of uncertainty. The results show that there are very few violations, even in the presence of large amounts of uncertainty. In Figure 6, we see that the maximum number of departure rate violations for the entire nation is about 20 and falls to 15 under uncertain conditions consistent with historical data. Figures 7, and 8 show that the number of sector capacity violations and the number of arrival rate violations is even lower. These low violation numbers in these last two results can be attributed to sector capacity and arrival rate being a binding constraint less often than departure rate. Note that the experiment measuring arrival rate violations is the only one where the robustness of the BIP solution differs from the evolutionary algorithm solutions. This robustness difference here could be attributed to the BIP solution optimizing air delays in addition to ground delays, creating a more complex schedule that is less robust. However, in all cases the number of violations is small.

Evolutionary Solution Stability

In addition to randomness intrinsic to the problem domain, evolutionary solutions have their own source of uncertainty in how the solutions themselves are pro-

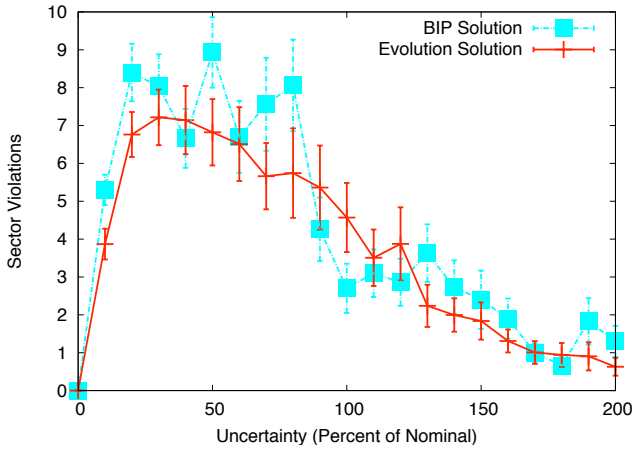

Figure 7: Sector Violations Caused by Uncertainty for BIP and Evolutionary Algorithm.

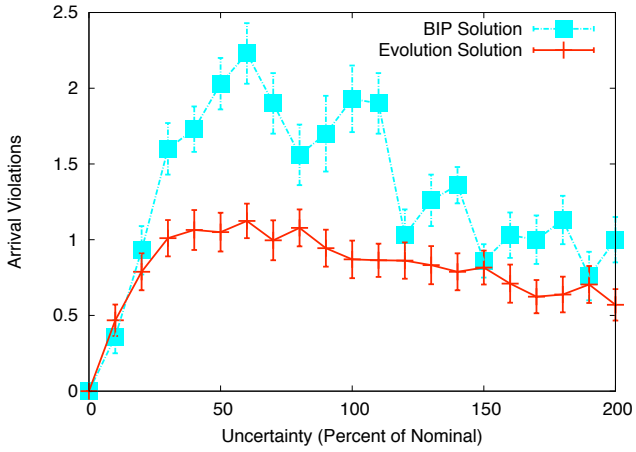

Figure 8: Arrival Violations Caused by Uncertainty for BIP and Evolutionary Algorithm.

duced. Since evolutionary algorithms are a form of random algorithm, each time the algorithm is run, a different solution can be formed. In this section, we test the stability of the evolutionary algorithms with respect to different solutions that are generated. We do this by testing five different evolutionary algorithm solutions produced using different random number seeds. Our goal is to see how well different solutions may handle uncertainty, so that we will have confidence that a particular solution produced by an evolutionary algorithm will behave as expected. The results in Figure 9 show that the evolutionary algorithms do indeed provide consistent results independent of the the particular solution instance.

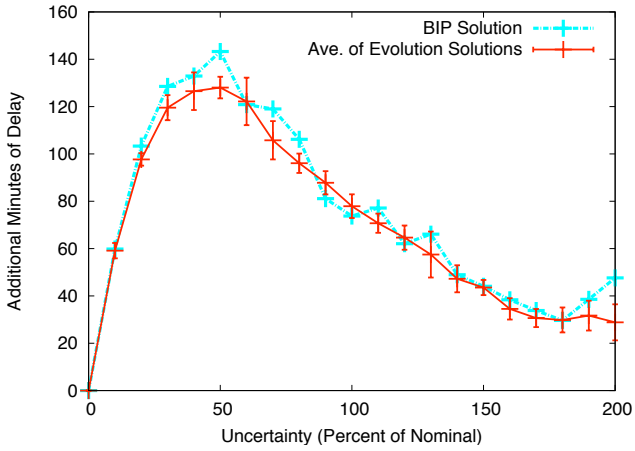

Figure 9: Performance Sensitivity of BIP and Evolutionary Algorithm to Uncertainty. Experiments are performed with alternative runs of evolutionary algorithm with different random number seeds. Error bars for the evolutionary performance represent standard deviation in performance between the evolution solutions with different number seeds. The different evolutionary solutions show similar performance.

All five solutions have about the same performance, for all levels of uncertainty present in the system. This result gives us confidence that the evolutionary algorithm can reliable produce a good solution.

6 Conclusions

The result of varying the departure times of aircraft after determining an optimal schedule for those departures is an increase in the number of departure, arrival, and sector violations in the National Airspace System, which is not a surprising result. However, by quantifying the number of violations, it is surprising to see the number of violations decrease as the amount of uncertainty added to the departure times increases. This is a similar effect noticed by Gilbo and Smith [9] as they studied the effect of uncertain sector entry times. Their analysis implied that as entry time uncertainty increases, the effect of an additional unexpected arrival of a flight in a sector is more likely offset by the unexpected *lack* of arrival by some other flight. While this analysis is not performed here, it seems a reasonable hypothesis worthy of further investigation.

The results presented here also show while uncertainty does impair the solutions discovered using deterministic bounds on capacities, the violations resulting from uncertain departure times may be within reasonable operational bounds. Further analysis would be necessary to confirm if this is true. Overall, the use of deterministic bounds within a traditional optimization approach or an evolutionary approach can produce solutions that perform well in the presence of operationally observed departure uncertainty. Incorporation of other uncertainties (airport capacities, sector capacities, etc.) would be an important inclusion to future studies along these lines.

Acknowledgments: This research was partially supported by NSF grant CNS-0931591.

References

[1] FAA, "Traffic Flow Management in the National Airspace System," Educational Booklet, 800 Independence Avenue, SW, Washington, D.C. 20591, October 2007.

[2] Bertsimas, D. and Patterson, S. S., "The Air Traffic Flow Management Problem with Enroute Capacities," *Operations Research*, Vol. 46, No. 3, May-June 1998, pp. 406–422.

[3] Rios, J. and Lohn, J., "A Comparison of Optimization Approaches for Nationwide Traffic Flow Management," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, August 2009.

[4] Sridhar, B., Grabbe, S., and Mukherjee, A., "Modeling and Optimization in Traffic Flow Management," *Proceedings of the IEEE*, Vol. 96, No. 12, December 2008, pp. 2060–2080.

[5] Mitchell, J. S. B., Polishchuk, V., and Krozel, J., "Airspace Throughput Analysis Considering Stochastic Weather," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Keystone, CO, August 2006.

[6] Krozel, J., Mitchell, J. S. B., Polishchuk, V., and Prete, J., "Capacity Estimation for Airspaces with Convective Weather Constraints," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Hilton Head, SC, August 2007.

[7] Zou, J., Krozel, J. W., Krozel, J., and Mitchell, J. S. B., "Two Methods for Computing Directional Capacity given Convective Weather Constraints," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Chicago, IL, August 2009.

[8] Gilbo, E. P. and Smith, S. B., "A New Model to Improve Aggregate Air Traffic Demand Predictions," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Hilton Head, South Carolina, August 2007.

[9] Gilbo, E. P. and Smith, S. B., "Probabilistic Predictions of Traffic Demand for En Route Sectors Based on Individual Flight Data," Tech. Rep. VNTSC-TFM-10-01, Volpe National Transportation Systems Center, January 2010.

[10] Wanke, C. R., Callaham, M. B., Greenbaum, D. P., and Masalonis, A. J., "Measuring Uncertainty in Airspace Demand Predictions for Traffic Flow Management Applications," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Austin, Texas, August 2003.

[11] Chen, N. and Sridhar, B., "Management-Action-Embedded Sector-Demand Prediction Models," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 6, November-December 2010, pp. 1892–1898.

[12] Hunter, G. and Ramamoorthy, K., "Evaluation of the National Airspace System Aggregate Performance Senstivity," *Proceedings of the Digital Avionics Systems Conference*, October 2007.

[13] Rios, J. L., "The Value of Reduced Uncertainty in Air Traffic Flow Management," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, Portland, Oregon, August 2011.

[14] Mueller, E. and Chatterji, G., "Analysis of Aircraft Arrival and Departure Delay Characteristics," *Proceedings of the AIAA Aircraft Technology, Integration, and Operations Conference (ATIO)*, Los Angeles, California, October 2002.

[15] Nilim, A. and Ghaoui, L. E., "Algorithms for Air Traffic Flow Management under Stochastic Environments," *Proceeding of the 2004 American Control Conference*, Boston, MA, USA, June 30 - July 2 2004.

[16] Mukherjee, A., *Dynamic Stochastic Optimization Models for Air Traffic Management*, Ph.D. thesis, UC Berkeley, 2004.

[17] Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st ed., 1989.

[18] Tumer, K. and Wolpert, D., editors, *Collectives and the Design of Complex Systems*, Springer, New York, 2004.

[19] Tumer, K. and Agogino, A., "Distributed Agent-Based Air Traffic Flow Management," *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Honolulu, Hawaii, May 2007.