Spring 5-29-2024

Connection-Saving Gate Assignment: A Computational Approach

Rob Mailley
Dartmouth College, rob.mailley.24@dartmouth.edu

Connection-Saving Gate Assignment: A Computational Approach

A Thesis
Submitted to the Faculty
in partial fulfillment of the requirements for the
degree of

Bachelor of Arts

in

Computer Science

by Rob Mailley

Faculty of Arts and Sciences
Dartmouth College
Hanover, New Hampshire

May 2024

ABSTRACT

The growth of the commercial aviation industry has yielded many interesting problems in the field of Operations Research, many of which are now able to be solved as both technology and mathematical optimization improve. A particularly interesting problem in airport operations research is the Aircraft Gate Assignment Problem (AGAP), which seeks to create a feasible matching between planes and flights at an airport. This problem is well-suited to modeling with Integer Programming, and has attracted research since the 1970s. Researchers of the AGAP have considered many different objectives, ranging from airline-focused objectives to more passenger-focused objective functions. In this paper, we build on the the work done by scholars of the AGAP, and present a new objective that focuses on reducing possible passenger misconnections, which we argue benefits both parties.

Missed connections cost airlines hundreds of thousands of dollars annually. While some of this is inevitable, there are cases where small adjustments would allow connecting passengers to get to the gate of their flight before the door closes. This paper focuses on those small adjustments that make all the difference. Specifically, we assign flights to gates such that the number of passengers that misconnect is minimized. In effect, this is accomplished by assigning flights with passengers on them with tight connections nearby gates. Special care is given to international flights and the effect of customs clearance and security screening, and our model is applied to a real day's operation for United Airlines at Dulles International Airport in Virginia. We found that our model has a statistically significant effect on reducing the number of passengers at risk of missing their connecting flights.

Acknowledgements

First, I want to thank my wonderful advisor Dr. Eugene Santos for his guidance throughout this thesis process. From taking me on for an independent study of Operations Research to continually answering all my questions, I could not have asked for a more supportive advisor. I would also like to thank Dr. Vikrant Vaze for encouraging me to take his graduate level Operations Research course, and helping me to refine this thesis into what you are reading today. I am especially indebted to him for his help in thinking through the mathematical formulation of the model.

I also want to thank my supervisors at United Airlines, Usma Johnson, Henry Bird, and Jim Schake, who gave me the ideas and experience to confidently write this thesis. Without the projects and experiences they gave me in the summer of 2023, I would not be in this position today.

Finally, I want to thank my friends both for putting up with and supporting me while I was writing this thesis. Rem, Ben, Kami, Taylor, Sarah, Maddie, Elise, Gannon, Ethan, Joel, Molly: enjoy your break from hearing me talk about airplanes.

And to my parents, Bob and Beth Mailley, for taking me down to Gravelly Point on Sunday afternoons, watching the planes land at DCA, and everything else you have done to encourage me to study what I am passionate about. Thank you. I am forever grateful.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

If you have flown, you have seen it. Travelers of all types, clutching their carry-ons, sprinting through the airport to make their connection to their next flight. Some airlines even will drive high-status passengers between gates in a luxury sports car, if the connection is tight enough [1]. There is even a Strava segment between two gates at Denver International Airport, in which connecting passengers turn on their fitness trackers and compete for the fastest sprint.

As much as passengers want to make their connections, airlines want their customers to make their connections even more. Where for passengers a missed connection can at worst cause a multi-hour delay, passengers missing connections cause bigger problems for an airline. These problems can be broken down into 3 major categories: financial, reputational, and operational.

Financially, rebooking customers on the next flight removes the possibility for an airline to book a last minute customer on the next flight. The opportunity cost here can be substantial, especially because these last minute flights tend to cost the most [2] and are therefore the most profitable for an airline. Accommodating passengers with missed connections creates more work for airline employees. Customer service agents have to rebook them, baggage handlers have to reroute their checked baggage, and misconnections create more headaches for operations managers. These are man-hours which would not have to be spent had the passengers made their returning flight. Airlines may also feel compelled to give out valuable air miles as compensation, so that flyers will

consider their airline the next time they fly.

This brings us to the next problem missed connections make - reputational. Airlines are dependent on their passenger's loyalty [3], and several missed connections may cause a drop-off in this loyalty, even if the problems that caused the delay were outside of the airline's control.

Finally, missed connections cause an operational strain on the airport. Airlines must devote more of their resources at an airport to accomodate these passengers, creating customer service lines, unhappy passengers, and other issues.

Because airlines have so many reasons to minimize these misconnections, and recently, it has been a focus for airline operations professionals. Two notable cases of this innovation are American Airlines's usage of Machine Learning for gate assignment, and United Airlines's development of ConnectionSaver.

In 2023, American Airlines rolled out a new system for usage at their major hubs, which they named "Smart Gating" [4]. While there is scant technical information available, it appears as if their system utilizes some form of statistical learning to assign gates to aircraft, with the aim of reducing taxi time and ramp congestion. They purport that among the many benefits of this smart gating system is that passengers have more time to make connections by reducing the amount of time spent on the ramp. This may be the case, but this system likely fails to take into account gate locations and passenger transit time.

In 2019, United Airlines introduced ConnectionSaver, a tool to predict passenger misconnects and inform gate managers of their options to hold flights [5]. The idea of ConnectionSaver is that delaying the departure of a certain flight for a few minutes to allow passengers to make their connections might be acceptable. To this end, ConnectionSaver looks at the "downline effects" of delaying departure. If delaying the departure of one flight may delay the next flight operated by that aircraft, then ConnectionSaver does not recommend a hold. If the crew would time out, Then it does not recommend a hold. ConnectionSaver looks at various factors, which includes passenger walk time, to make this decision. However, it does not recommend gate changes, nor does it consider the impact of customs delays for international arrivals.

What this paper tries to accomplish is a in part a combination of the two systems developed by United and American. Like ConnectionSaver, it takes into account passenger walk time to free up minutes for passengers on tight connections. Like American's system, it attempts to save time through gate allocation. The main difference, however, is the utilization of mathematical optimization in our model to achieve fewer missed connections. By proactively assigning flights with passengers that have tight connections gates that are closer to each other, it saves passengers precious minutes and allows them to connect.

It is our opinion that this topic is even more important than the airlines realize. Minimizing the risk of misconnections is also a loyalty issue and an accessibility issue.

We have already described how airlines will occasionally drive high-loyalty passengers from gate to gate if they have a tight connection. But this is only available to customers at the highest tiers of the respective airline's loyalty programs. Instead of optimizing over all customers, airlines can only focus on high loyalty passengers in their schedules or otherwise weight those passengers higher. By doing this, airlines can effectively use computational techniques to drive loyalty and effectuate better business.

At the same time, airlines can also focus disabled customers. By doing this, they can make connecting distances for passengers with mobility or other issues shorter. These passengers often take the longest to transit the gates, so they stand to benefit the most from this improvement, and experience a more equitable travel experience.

In short, there are many advantages to optimizing gate assignments in this way. But this is far from the only way to optimize assignments, and ours builds on previous research done to find those other methods. We discuss those in the next chapter.

Chapter 2

Literature Review

Few problems in Transportation Operations Research have enjoyed as much coverage as the aircraft gate assignment problem. Serious study of the AGAP was started by Braaksma in 1977 [6]. Before this point, most of the research into aircraft gating was architectural in nature, but Braaksma offered a new way to minimize walking distance at already built airports. He also was the first to take into account the reccommended walking distance offered by IATA. The next mention of the AGAP in the literature comes from Babic's et. al.'s 1984 paper, where he was the first to model the AGAP as a problem in combinatorial optimization [7]. However, he uses a branch-and-bound backtracking algorithm, and he does not take into acoount connecting passengers, only distance to check-in and the baggage claim. However, their paper is the first to take the AGAP, model it, and compare it against random assignment. Babic et. al.'s results show that the optimal assignment can significantly reduce the number of passengers walking maximum distances compared to random assignments, though the impact on average walking distance is more modest.

In 1985, Mangoubi et. al. build substantially on Babic, considering three sets of passengers, arriving, departing, and connecting, as opposed to only arriving and departing [8]. They also solve a linear relaxation of the IP. A limitation here is that Mangoubi et. al. use a uniform probability distribution to estimate the average walking distance, which is said to overstate the transfer distance. He also considers the effects of implementing a heuristic search, both as a bootstrap for the integer

program and as a standalone assignment, kicking off a long tradition of interesting heuristics in the AGAP. Then in 1990, Bihr was the first to formulate the problem as a strict 0-1 integer program that focuses on transfer passengers [9]. He uses matrix multiplication to generate a matrix C_{ij}, where the entries are the passenger distances traveled from flight i to gate j, and enters that into his model. His model also provides information on the cost penalties of sub-optimal solutions directly from the optimal basis. Bihr also discusses potential of using his model to establish initial gate schedules at major hub airports which can then be dynamically adjusted based on updated passenger loading information while utilizing the cost penalty data to minimize disruption to existing assignments.

According to Das and Gzara, as time went on, researchers placed more of their focus on researching heuristic solving methods to cope with the ever-growing size of the integer programming models [10]. Tabu search and Simulated Annealing are two especially popular heuristics, but simple heuristics, such as those that assign larger flights to closer gates, are popular as well.

Around the same time of the rise of heuristics, the objectives considered in AGAP research diverged into two separate categories: passenger-centric and airline-centric [10]. Up to this point, we have only discussed passenger-centric AGAP solutions.

One obvious avenue for researchers that focus on European airports are those that minimize the number of ungated flights. That is, attempting to reduce the number of flights that are assingned remote stands that passengers must be bussed to, which is generally not a common practice in the United States, where most mainline flights are assigned jetways. Dorndorf et. al. wrote a number of papers related to this objective, among others [11]. Popular topics also include minimizing taxi times, fuel burn, minimizing towing moves, and maximizing airline gate preference [10].

Until Tang and Wang in 2017, the effect of passengers transiting through customs was not widely studied in the literature. Tang and Wang consider four objectives: maximizing the number of departing flights assigned to gates near the airline's VIP lounge, maximizing the number of arriving flights assigned to gates near customs, maximizing the number of arriving and departing flights using the same aircraft assigned to the same gate, and minimizing the number of flights

assigned to remote gates [12]. The third objective here in effect minimizes tows, something that an expanded version of this thesis would encompass.

The only paper that explicitly mentions misconnects in its objectives is in a 2012 paper from Maharjan and Matis [13], who attempt to reduce some quantity of "passenger discomfort" in walking (or running) from gate to gate, but also have another objective which seeks to reduce aircraft taxi fuel burn. Our approach differs from theirs in its construction, and instead of minimizing a measure of "discomfort," we seek to reduce the actual number of passengers that are at risk of misconnecting.

We also assert that in the case of reducing misconnects, the dichotomy of passenger-centric and airline-centric objectives is not applicable. Reducing misconnects benefits both parties, as we have explained before.

Chapter 3

Approach

In this Thesis, we formulate a binary integer programming model which maps flights to gates. Formally, we seek to find a bipartite mapping $F \rightarrow G$ where F is the set of flights an airport sees in a day, and G is the set of gates at that airport.

First, we must obtain a dataset of flights. We obtain this from publicly available sources, and combine it into a table, indexed by the arrival time of the first flight. The schema is offered in table 3.1.

The columns presented in 3.1 comprise the minimal amount of data needed to run this model. Arrival and Departure times are needed to check for overlap and for customs delays, and inbound and outbound code are needed to check if the aircraft needs an international-capable gate, and type is needed for gate sizing.

Column	Data Type
Arrival Time	int
Departure time	int
Type	string
Origin	string
Destination	string
Aircraft Type	string
Inbound Code	int
Outbound Code	int

Table 3.1: Flight Data Schema

We preprocess this data using Python's pandas and dateutil libraries, for ease of dealing with timestamped data. We then obtain a list of gates at the airport, and append 'X' to the list to denote a remote stand. Then we import a set of US airports, to check which flights are international arrivals.

One thing that this paper does that is distinct from the previous literature is our conception of flight movements. Where other authors used sliding-window and other models of aircraft moving through space and time, we use 3-tuples to represent a movement of aircraft. The first entry represents where the passengers on the first flight deplane, the second where the plane sits before boarding, and the third where passengers on the second flight board. Paired with the arrival and departure data, we then have a record of where the aircraft is at all times. For movements comprised of the same gate, this is obvious. For movements where a tow happens, the tuple will be of the form (G1,X,G4). X denotes some remote apron stand. However, the plane does not necessarily have to be towed immediately following deplaning, nor does it have to be towed at the instant boarding begins. The ambiguity in the flight tuples allows us to be flexible on when the tows happen. Although it is beyond the scope of this thesis, the tuple method also allows constraints on the number of tows in use at one time. This is an important question for many airport operations teams, as the pushback tugs used for every flight are not suited to drive aircraft around an airport. The tugs used for towing are called supertugs, and they are expensive and much fewer in number, and would constrain the flight to movement assignment variables.

After the movement variables are generated, they are filtered to retain the valid movements generated above. After that, we iterate through the flights in the given dataset, and create valid assignments of flights to movements. In this step, we give special care to only instantiate movement variables that are valid. That means that a widebody aircraft can never be at a gate that cannot fit it, nor can an international arrival ever be assigned a movement that starts at a domestic-only gate.

Then, we iterate through each pair of flights and see which flights overlap with each other, generating a list of pairs of flights. To see which flights overlap with each other, we use a simple conditional: if $\max(f_1^a, f_2^a) < \min(f_1^d, f_2^d)$, then the flights overlap at some point.

Then for each pair in this list, we check which segments of their movements would overlap. We

divide up the segments using passed in parameters for the time desired for deplaning and boarding time; these parameters can be altered based on aircraft time and flight destination.

This filtering generates a conflict set, that is, all possible pairs of valid assignments that would overlap for at least one gate for at least one minute.

We use these to generate the other assignment variables in the model, and then generate a set of valid connections. A pair of flights f_1, f_2 is designated as a valid connection if $f_1^a < f_2^d$, without loss of generality. We take these and generate a dictionary of connection times.

Then, we needed to generate a dictionary of gate transit times. For large airports with many terminals, the best way to do this would to be to generate a directed graph $G = (V, E)$ of the airport where V is the set of gates, and E is the set of edges weighted by transit time between them. Then we would run a shortest paths algorithm on the set of possible connecting gates, storing the times in a dictionary. We however, do not have to generate this graph, for reasons discussed in chapter 5.

However, this graph construction would only be valid for domestic flights, as passengers would have to transit through customs for international gates. Here is where the parameter computation would encounter some complexity. On one hand, it would be easy to satisfy the connection requirement by adding a dominator node $v_1$in the graph with an in-vertex from every international gate, only one out vertex to another dummy node, v_2. The edge (v_1, v_2) would set the customs wait in that hour time block, and you could again run a shortest paths algorithm such as Djikstra's.

Obtaining customs time is also an interesting problem in and of itself, that lends itself to interesting solutions in the field of queueing networks that we discuss later. Fortunately, CBP keeps track of the average wait times at the FIS, and we utilise that data in this project. An interesting application of machine learning would be to look at all the aircraft movements and the load factors[1], as well as weather and staffing data, and construct a regression model that forecasts wait times in customs.

However, if we didn't have that, this thesis would have taken on a new sub-project that itself needs research. The movement of passengers through an airport would have had to been modeled

[1]Load factor is the percentage of seats filled on an aircraft.

as a queuing network, as in Ebert [14]. Although queuing networks are beyond the scope of this paper, they would lead into even more accurate models, as forecasting time in customs is often more reliable than using historical data.

Using this data, we obtain our dictionary of gate transit times. This is where we include deplaning time and the cutoff time before boarding, both assumed to be 15 minutes for simplicity.

The final step in preprocessing before model construction is generating a dictionary of connecting flights to quantity of passengers. This is a closely guarded trade secret, and airlines use this connection data to price flights and set their schedules. As such, we are not privy to such information, so we randomly generate it.

Chapter 4

Model Formulation

Objective:

$$\text{Minimize} \sum_{i \in F} \sum_{j \in F \setminus \{i\}} \sum_{a \in g} \sum_{b \in F \setminus \{a\}} w_{ijab} \cdot c_{ij}$$

Subject to the following constraints:

$$y_{ia} + y_{jb} + \mu_{ijab} - 2 \leq w_{ijab} \quad \forall i, j \in \mathcal{F}, \forall a, b \in \mathcal{G} \tag{4.1}$$

$$z_k + z'_k \leq 1 \quad \forall (k, k') \in \mathcal{K}_2 \tag{4.2}$$

$$z_{ijm} \leq y_{ia^1_m} \quad \forall m \in \mathcal{M}_{ij}, (i, j) \in \mathcal{P} \tag{4.3}$$

$$z_{ijm} \leq y_{ja^2_m} \quad \forall m \in \mathcal{M}_{ij}, (i, j) \in \mathcal{P} \tag{4.4}$$

$$z_{ijm} \geq y_{ia^1_m} + y_{ja^2_m} - 1 \quad \forall m \in \mathcal{M}_{ij}, (i, j) \in \mathcal{P} \tag{4.5}$$

$$\sum_{a \in \mathcal{G}_i} y_{ia} = 1 \quad \forall i \in \mathcal{F} \tag{4.6}$$

$$\sum_{m \in \mathcal{M}_i j} z_{ijm} = 1 \quad \forall i, j \in \mathcal{C} \tag{4.7}$$

$$y_{ia} \in \{0, 1\} \quad \forall i \in F, a \in \mathcal{G} \tag{4.8}$$

$$w_{ijab}, x_{ijab} \in \{0, 1\} \quad \forall (i, j) \in C, a, b \in \mathcal{G} \tag{4.9}$$

$$z_{ijm} \in \{0, 1\} \quad \forall i, j \in F, m \in \mathcal{M}_{i,j} \tag{4.10}$$

We seek to minimize w_{ijab}, which in (4.1) is set to 1 if and only if flight i is assigned to gate a and j to b, and if μ_{ijab} is one. Note that the necessity of this condition is enforced by the objective, because w being set to 1 will necessarily increase the objective function. We can divide our constraints into three sections: Connection timing constraints (4.1), Gate conflict constraints (4.2-6), and binary constraints (4.8-10).

	Arr	Dep	Movement
UA1	0700	0900	(C1, C1, C1)
UA2	1400	01800	(C1, C1, C1)

Table 4.1: Non-Conflicting Pair of z_{ijm} Variables

	Arr	Dep	Movement
UA3	1200	1900	(C1, X, C3)
UA4	1715	01845	(C3, C3, C3)

Table 4.2: Conflicting Pair of z_{ijm} Variables

Next, we have the gate assignment constraints. As I have discussed before, we take a different tack here than the standard literature on the topic does. In preprocessing, we have generated a list of conflicting movements, which is a list 2-tuples of 3-tuples. Essentially, the list contains every set of flight-movement pairings that would cause a conflict. Note that these are time dependent. Constraints (4.4-6) works with the y_{ia} flight-gate constraints and the z_{ijm} movement variables to ensure that if a flight pair i, j is assigned the movement $m = (g, g, g)$ where $(i, j) \in \mathcal{P}$ and $g \in \mathcal{G}_i \cap \mathcal{G}_j$, or $z_{ijm} = 1$, then $y_{ig} = y_{jg} = 1$. This condition holds the other direction as well. (4.7) assigns each flight $f \in \mathcal{F}$ to a gate in $g \in \mathcal{G}$.

Finally, we have the binary assignment constraints, which simply constrict each decision variable $w_{ijab}, y_{ia}, z_{ijm}$ to taking values in $\{0, 1\}$.

Table 4.3: Parameters

$\mathcal{P} = \mathcal{I} \times \mathcal{D}$	Set of all flight-pairs operated consecutively with the same aircraft, indexed by departure time
$\mathcal{F}$	Set of all scheduled flights that arrive and depart at the airport, $\mathcal{I} \cup \mathcal{D}$
$\mathcal{C}$	Set of flight pairs (i, j) that have connecting passengers
$\mathcal{I}$	Set of arriving flights
$\mathcal{D}$	Set of departing flights
$\mathcal{K}_2$	Set of all pairs of conflicting assignments
$\mathcal{G}$	Set of all gates
$\mathcal{G}_i \subseteq \mathcal{G}$	Set of all gates that can be assigned to to flight i
$\mathcal{M}$	Set of all movements possible at an airport. Note that each movement is a complete space-time trajectory that can be assigned to an aircraft for its entire time starting from the arrival at the first gate and until the pushback at the second gate (which can be the same as the first gate).
M	Arbitrarily large constant
$\mathcal{M}_{ij} \subseteq \mathcal{M}$	Subset of all movements that are compatible with incoming flight i and outgoing flight j.
international_arrivals	Array indexed by flights such that International_arrivals[i] $= 1$ iff flight i arrives from an international destination and requires passengers to clear customs
international_gates	Array indexed by gates such that International_gates[a] $= 1$ iff gate a has an entry to the sterile corridor to customs
t_{iab}	Time needed to transit between gates $a, b \in G$ (including any wait times), at the moment of arrival of flight i
s_{ij}	Time available to connect between flight i and flight j
μ_{ijab}	Parameter that is set to 1 if the time it takes to transit from gate a to gate b is less than the amount of time between the arrival of flight i and the departure of flight j $\forall (i, j) \in C$
c_{ij}	Number of connecting passengers between flights $i, j \in F$

Table 4.4: Decision Variables

w_{ijab}	Binary decision variable that is 1 iff flight i is assigned to gate a and flight j is assigned to gate b, and there is not sufficient time to make the connection from i to j, $\forall(i,j) \in C$
y_{ia}	Binary decision variable that is 1 iff flight i is assigned to gate a, $\forall a \in G_i, i \in \mathcal{F}$
z_{ijm}	Binary decision variable that is set to 1 iff flights i and j are part of the same turn, that is, operated by the same aircraft, and are assigned a movement m. Note that every unique combination of i, j, m is together also called a unique assignment.

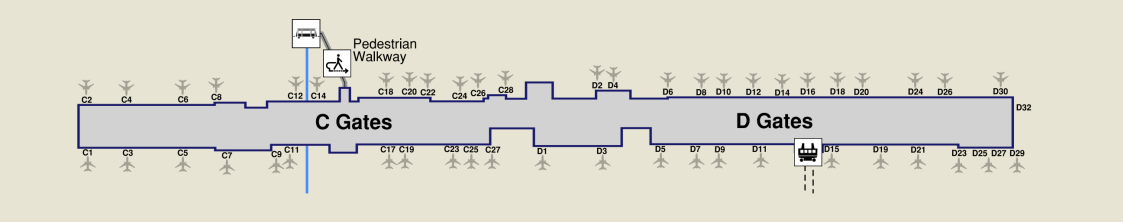
Chapter 5

Results

We ran this model on a MacBook Pro with an M1 CPU and 16 GB of RAM, as to simulate the equipment gate managers at an international airport would have. The preprocessing was done with Python, and the Integer Program model was developed with the Gurobi API.

We ran this model on a single day of United's schedule at IAD, and the data was derived from AeroBox's Flights API. Some flight codes were changed as to remove duplicates from the dataset, but those flights were not removed from the dataset. We also removed all flights not operated on United mainline aircraft – that is, United Express branded regional jets operated by contractors.

Figure 5.1: Dulles Concourses C and D



We chose IAD because of the physical plant, which is simple and lends itself to this model well. As in figure 5.1, United's Mainline operations take place in the C and D concourses, and they do not compete with other airlines for gates at those locations. Additionally, the terminal setup is linear, and the numbering system for the gates is simple. Compared to Denver international airport, which has multiple linear concourses in parallel, or Newark, which utilizes finger piers, Dulles's

simple topology makes it an ideal candidate. To calculate the walk time, we simply say that it takes a minute to transit between subsequently numbered gates. It also is the longest concourse without a moving walkway in the world [15], which simplifies walk time calculation.

Finally, researchers up to this point have given very little attention to the effect of customs delays on passenger connection times. In the United States, all passengers arriving from international locations must clear Customs and Immigration, regardless of their final destination. This stands in contrast with the system used in Europe, where flights departing to and arriving from locations outside the Schengen zone are given their own terminal. At Dulles, passengers arriving from outside the US, who are connecting to united flights make use of a midfield customs station called the Federal Inspection Station, or FIS. These passengers are routed through a "sterile corridor," where they first go through immigration. This can be a large bottleneck. Dulles's banked schedule has four banks of flights[1], and third bank arrivals start around 1400 and last until 1630. This is also when most European flights arrive in the US, so that the planes can be turned around and sent back in the early evening, arriving in Europe again in the early morning. Overnight flights from South America and East Asia also arrive in the early morning.

Accordingly, there is great demand for the FIS in the early morning and afternoon, and backups in the line can be severe. US Customs and Border protection keeps data on average wait times when the FIS is open, and these data is implemented in the walk time dictionary as detailed before.

Taking into account all these factors, we built the model, and the results are as follows.

5.1 Results

To test this model on real life conditions, we used the United Mainline schedule for August 1, 2023. By Washington, DC, standards, August 1 was a perfect day for operations at Dulles. There were no afternoon thunderstorms, the wind was out of the north at about 10kt, and a high temperature of 81 degrees. This means that there were no ramp closures, and there should not have

[1]Banks of flights refer to clusters of flights that arrive and take off around the same time, allowing for uniform connection time.

been any abnormal backups in the FIS. Accordingly, this would be an ideal day upon which to test the model, and that is what we have done.

As a baseline, we set the objective function to zero and generated a random feasible assignment of flights to gates. This assignment is given in Table 5.1.

Table 5.1: Bank 1: Randomized assignment

Incoming Flight	**Outgoing Flight**	**Gate**
783	2021	D29
2626	1154	D5
584	1964	D16
2050	380	D32
1395	1600	D12
73	700	C8
2046	277	C4
1562	618	C1
997	1744	C2
860	2058	C3
1325	641	D6
2280	260	D11
379	1949	C11
2492	1203	D14
1228	1406	D26
1657	667	D1
493	1443	D15
2487	381	C25
1274	2336	D7
745	2628	C4
1785	602	D19
2247	1661	C17
1599	2314	D21
2132	1809	C26

Although it may not seem so, these randomized assignments are a good approximation of what would actually happen at an airport. Oftentimes, gate managers will choose the first feasible solution they manually discover.

Taking these assignments, we can calculate the number of passengers at risk of missing their connection. For this model run, we obtained that there were 251 passengers at-risk of missing their connection.

Then, we brought back in the original objective function, and re-ran the model. The results are displayed in Table 5.2.

Table 5.2: Bank 1: Optimized assignment

Incoming Flight	**Outgoing Flight**	**Gate**
783	2021	D32
2626	1154	D5
584	1964	D12
2050	380	C5
1395	1600	D21
73	700	C9
2046	277	C4
1562	618	C6
997	1744	C7
860	2058	C8
1325	641	D16
2280	260	D15
379	1949	C26
2492	1203	C28
1228	1406	C17
1657	667	C2
493	1443	D3
2487	381	D6
1274	2336	D4
745	2628	D2
1785	602	D1
2247	1661	C23
1599	2314	C19
2132	1809	C20

There is a substantial difference in the value obtained by the objective function: only 24 passengers are at risk of missing their connections.

This is substantial, and shows real improvement over random assignment. To further prove the model's impact, we ran the model 10 times on different slices of the schedule, and did a t-test. The numbers of passengers at risk of missing their connections in the randomized group of 10 model runs($M = 165, \sigma = 146.6$) was much higher than the same numbers for the optimized group ($M = 23.4, \sigma = 31.6$), $t(18) = 2.84$, $p = .005$. This result is extremely statistically significant.

Althought this model was intended to be small enough to run on a laptop, we would like to run

the model on a computer that is large enough to run on an entire day's schedule, across multiple days, to get even more results, and further prove the efficacy of the model.

This model could be even more useful in larger airports, as well. We have noted that IAD is simple in comparison to other US international airports such as Dallas-Fort Worth, Newark-Liberty, and Chicago-O'Hare, among others. At these airports with longer distances, moving walkways, and people movers, there are even more gains to be made from an introduction of a model of this type.

Also, although we built in the capability to allow for tows, we decided not to use it in the model runs presented. The size of the conflict sets became too large, and the computer used to run the simulation ran out of memory. This is something that would be possible with a larger computer.

Chapter 6

Further Study and Conclusions

In this paper, we prepossessed the data for and built an integer programming model that minimizes the number of passengers that are at risk of missing their connections. Airports are not getting any less busy, the number of flights are increasing, and the need for mathematical optimization in planning is as great as it has ever been. Our solution contributes to this field, and has the potential to have a real impact on the efficiency of airline operations, and positively impact an airline's bottom line.

One area we did not discuss here is schedule recovery. As climate change drives up the frequency of storms, ground stops are becoming all too common at US airports. Our models and algorithms need to be resilient to such issues, and we are convinced that ours, with some reinforcement, is. Over small sets of flights, such as the size of a single bank, our model can be run in under a minute on a standard laptop. By editing the flights dataset on the fly, perhaps from real time schedule data, our model can quickly reassign gates to save customers precious minutes to make their flights.

We also identified avenues for further research on this project, which we will detail further here. We talked about the extension into queuing theory, and generally, the need for more simulation in airport planning. This is a burgeoning problem in airport operations research, with the problem of "gate lice," or people that stand before the gate podiums early and block the concourse,

reaching fever pitch [16]. Walking simulation applied to the gate assignment problem would not only provide interesting forecasts for customs, but potentially to allocating gates as to minimize terminal congestion.

Additionally, we did not cover robustness in this paper, which has been a frequent topic of AGAP researchers. We would suggest adding the average delays of incoming flights to the S_{ij} dictionary, to plan for the expected time of arrival, as opposed to the scheduled time of arrival.

We also discussed the applications of machine learning, and the combination of optimization and machine learning is a powerful tool to solve operational issues at airports. Airlines are slowly starting to catch up with the amount of data they generate every day. While airlines have always been pioneers when working on pricing and revenue management, they are just starting to master the large quantities of operational data they hold. For example, creating machine learning models that map gate assignments to total waits in customs could be extremely useful in generating parameters for optimization models.

However, optimization and machine learning are not the ultimate solution. There is much to be learned from the people who work at airports and see things every day that we as academics do not. It is only when we combine real world expertise and practice with computer science techniques that we gain a real handle on these issues.

Bibliography

[1] United Airlines. United and Jaguar Launch the First All-Electric Gate-to-Gate Airport Transfer Service in the U.S.

[2] Stefanie Waldek. 10 Tips to Help You Score Cheap Last-minute Flights. Section: Travel + Leisure.

[3] Steven Leon and Sonoma Dixon. Airline satisfaction and loyalty: Assessing the influence of personality, trust and service quality. *Journal of Air Transport Management*, 113:102487, October 2023.

[4] American Airlines. Straight to the gate this holiday season, December 2023.

[5] United Airlines. United Airlines Makes Connecting the World Easier Than Ever with ConnectionSaver, June 2019.

[6] J. P. Braaksma. REDUCING WALKING DISTANCES AT EXISTING AIRPORTS. *AIRPORT FORUM*, 7, August 1977.

[7] Obrad Babic, Dusan Teodorovic, and Vojin Tosic. Aircraft Stand Assignment to Minimize Walking. *J. Transp. Eng.*, 110(1):55–66, January 1984.

[8] Rami S Mangoubi and Dennis FX Mathaisel. Optimizing gate assignments at airport terminals. *Transportation Science*, 19(2):173–188, 1985. Publisher: INFORMS.

[9] Richard A. Bihr. A conceptual solution to the aircraft gate assignment problem using 0, 1 linear programming. *Computers & Industrial Engineering*, 19(1):280–284, 1990.

[10] Galesin Sena Das, Fatma Gzara, and Thomas Stutzle. A review on airport gate assignment problems: Single versus multi objective approaches. *OMEGA-INT J MANAGE S*, 92:102146, 2020. Place: OXFORD Publisher: Elsevier Ltd.

[11] Ulrich Dorndorf, Andreas Drexl, Yury Nikulin, and Erwin Pesch. Flight gate scheduling: State-of-the-art and recent developments. *Omega*, 35(3):326–334, June 2007.

[12] Ching-Hui Tang and Wei-Chung Wang. Airport gate assignments for airline-specific gates. *Journal of Air Transport Management*, 30:10–16, July 2013.

[13] Binod Maharjan and Timothy I. Matis. Multi-commodity flow network model of the flight gate assignment problem. *Computers & Industrial Engineering*, 63(4):1135–1144, 2012.

[14] Anthony Ebert, Ritabrata Dutta, Kerrie Mengersen, Antonietta Mira, Fabrizio Ruggeri, and Paul Wu. Likelihood-Free Parameter Estimation for Dynamic Queueing Networks: Case Study of Passenger Flow in an International Airport Terminal. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 70(3):770–792, June 2021.

[15] Oppendahl, Carl. Dulles's C and D concourse, October 2017.

[16] Sofia Andrade. Why do âgate liceâ line up early for a flight? Psychologists explain. *Washington Post*, December 2023.