ABSTRACT

Title of Dissertation: COMPUTATIONALLY TRACTABLE STOCHASTIC INTEGER PROGRAMMING MODELS FOR AIR TRAFFIC FLOW MANAGEMENT

Charles Nathan Glover, Ph.D., 2010

Dissertation directed by: Professor Michael O. Ball
Decision and Information Technologies,
Robert H. Smith School of Business

A primary objective of Air Traffic Flow Management (ATFM) is to ensure the orderly flow of aircraft through airspace, while minimizing the impact of delays and congestion on airspace users. A fundamental challenge of ATFM is the vulnerability of the airspace to changes in weather, which can lower the capacities of different regions of airspace. Considering this uncertainty along with the size of the airspace system, we arrive at a very complex problem. The development of efficient algorithms to solve ATFM problems is an important and active area of research. Responding to predictions of bad weather requires the solution of resource allocation problems that assign a combination of ground delay and route adjustments to many flights. Since there is much uncertainty associated with weather predictions, stochastic models are necessary.

We address some of these problems using integer programming (IP). In general, IP models can be difficult to solve. However, if "strong" IP formulations can

be found, then problems can be solved quickly by state of the art IP solvers. We start by describing a multi-period stochastic integer program for the single airport stochastic dynamic ground holding problem. We then show that the linear programming relaxation yields integer optimal solutions. This is a fairly unusual property for IP formulations that can significantly reduce the complexity of the corresponding problems. The proof is achieved by defining a new class of matrices with the Monge property and showing that the formulation presented belongs to this class. To further improve computation times, we develop alternative compact formulations.

These formulations are extended to show that they can also be used to model different concepts of equity and fairness as well as efficiency. We explore simple rationing methods and other heuristics for these problems both to provide fast solution times, but also because these methods can embody inherent notions of fairness. The initial models address problems that seek to restrict flow into a single airport. These are extended to problems where stochastic weather affects en route traffic. Strong formulations and efficient solutions are obtained for these problems as well.

COMPUTATIONALLY TRACTABLE STOCHASTIC INTEGER PROGRAMMING MODELS FOR AIR TRAFFIC FLOW MANAGEMENT

By

Charles Nathan Glover

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

Advisory Committee:
Professor Michael O. Ball, Chair
Professor David Lovell
Dr. Robert Hoffman
Professor Eric Slud
Professor Carol Espy-Wilson

Dedication

To my parents, Charles Hewlett Glover and Glenda Baskin Glover, who made all of this possible, for their endless support, encouragement and patience.

Acknowledgements

I would first like to thank my dissertation advisor, Michael Ball, whose guidance, insight, and belief in me helped me to reach this point. I would also like to thank Robert Hoffman and David Lovell, who have provided me with advice and support since I joined the NEXTOR community. I would also like to thank the other members of my committee, Eric Slud and Carol Espy-Wilson who have both provided great assistance in guiding me to this step.

The MATH and AMSC departments here at Maryland have been extremely helpful in my progression. I would like to thank Robert Johnson, Larry Washington and William Gasarch for their support and advice throughout my time here.

I was first introduced to research in mathematics by Duane Cooper at Morehouse College. He has remained a great friend and an asset throughout this process. I would also like to thank him along with Arthur Jones, Chung Ng, and Curtis Clark who all played important roles in my development at Morehouse.

I thank the PROMISE community here at Maryland, who have also had a significant contribution to my success. Both the support and encouragement I have received from Johnetta Davis, Renetta Tull, and Janet Rutledge, as well as the information and insight I have received from the many informative talks which they have sponsored have been very helpful towards this accomplishment.

I also could not have reached this point without the support and understanding of so many friends. The NEXTOR community here at Maryland – Andrew Churchill, Shin-Lai Tien, Moein Ganji, Kleoniki Vlachou, Prem Swaroop, Nasim Vakili, Kennis Chan, and Alex Nguyen – has been extremely helpful throughout this process. I would also like to thank my study partners, Patrice Gregory, Shanna Smith, Tina Ligon, Angel Miles, Shelby Wilson, Sean Barnes, Gleneesha Johnson, Nicole Nelson, Sophoria Westmoreland, Lenisa Joseph, Alexis Williams, Angela McRae, Jamie Chatman and many others. To my friend and roommate Paul Matthews, I thank you for your patience and help through both my ups and downs. To my mentors, Calandra Tate, Monica Jackson, Angela Grant, William Howell, Shaun Gittens, Eulus Moore, and Stacey Nichols, I thank you for providing me with so much direction and guidance throughout this process and helping me to realize that this is possible. To Alverda McCoy, who has just been a blessing in so many ways, I cannot say thank you enough for all the help you have provided.

Finally, I would like to thank my family, who has been a steady rock of support throughout this entire process. My sister, Candace Glover, has been a source of pride and inspiration. My mother and father, Glenda Baskin Glover and Charles Hewlett Glover, encouraged me to pursue mathematics and science from an early age. They have never wavered in helping me pursue this goal. I thank you from the bottom of my heart for your love and encouragement.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

We live in a world dominated by the need for on-time performance. This need is countered with unexpected and uncertain events that derail performance. We are then left with the question of how best to perform in an environment without perfect information. This is a fundamental challenge in Air Traffic Flow Management (ATFM) and Ground Delay Programs (GDPs) in particular.

Here, we give a general overview of Air Traffic Flow Management (ATFM). For more specific information, see (Ball et al., 2007), (Vossen, Hoffman and Mukherjee, 2011), or (FAA, 2006). The Federal Aviation Administration's (FAA) air traffic flow management specialists have set a priority on resolving instances in the National Airspace System (NAS) where the anticipated demand exceeds capacity. Whenever the FAA predicts that the number of flights arriving at an airport within a 15-minute interval exceeds the capacity of the current runway configuration, FAA directives mandate a response. One of the primary limitations on capacity is the finite number of airports that can be built and the constrained number of runways at each of these airports. One such way that a demand capacity imbalance can occur is by a significant increase in the traffic between these airports. This increase in air traffic, though, occurs over a long period of time and thus the FAA has more of an opportunity to prepare for it. A much more complex situation arises when bad weather occurs in airspace. Runway capacity is a limited resource under good weather conditions. Poor weather conditions over an extended period of time can reduce the existing capacity causing a situation to arise where the number of flights attempting to land during this time exceeds the capacity. The result is that some flights must be delayed and the question of which flights should experience delays, and how much delay should be assigned to each flight is resolved though a traffic flow management initiative. One of the most common such procedures is a GDP. These procedures are usually planned in the expectation of bad weather; flights are held on the ground before they depart from their origin airports. These are effective because delay is shifted from being airborne delay to ground delay, which is both less costly and less risky.

GDPs were initially implemented after the oil crisis of the 1970s and the air traffic controller strikes of the 1980s, which made it attractive to reduce airborne delays by holding flights on the ground. Since then, they have become a major part of the U.S. ATFM strategy. Initially the question of how much delay to assign to each flight was handled by a method called Grover-Jack. This is a method for assigning flights to arrival slots based on their estimated time of arrival (ETA). It was shown that this method can be abused by providing an inaccurate or out of date ETA. These methods are precisely what airlines resorted to when they felt they were being treated improperly by the Grover Jack methods. This comes from the fact that, if an airline is flying into an airport experiencing a GDP, then the airline is penalized for reporting delays or cancellations of flights. For example, consider a flight that has an ETA of 1:00 pm, but because of mechanical difficulties is unable to arrive until 1:30. If the flight reports this to the FAA, then they are assigned a slot based on this new ETA of 1:30 instead of the original ETA of 1:00. Airlines saw this as being penalized for being truthful about their delays and cancellations, and this was referred to as the “double penalty”. As a result, this led to inaccurate information relating to arrivals, which led to inefficient GDPs.

Inequitable and inefficient GDPs led to the need to reconsider how GDPs were implemented. At the core of this was the need to remove the penalty for voluntary submission of timely and accurate flight data. No party involved in air traffic management has complete information. The FAA has a published schedule, giving them knowledge of the arrivals and departures as well as the status of airborne flights. But this schedule is published well in advance of any GDP implementation. This means that it does not reflect changes in the departure times of flights due to mechanical problems, delays to inbound flights, etc. In order to efficiently implement a GDP, the FAA needs active participation from the airlines. On the other hand, airlines know information about their flights, and can make adjustments to their own schedules around weather reports, but do not have any information about the overall demand and capacity at airports. As a result, Collaborative Decision Making (CDM) emerged based on the philosophy that an increase in data exchange and collaboration between the parties involved will lead to more effective and efficient decisions in ATFM. In 1998, CDM procedures were used to plan GDPs, and CDM became the official policy of the FAA.

A primary component of CDM is the ration-by-schedule (RBS) algorithm. Unlike the Grover Jack method, where flights were ordered by ETA, RBS allocates slots using a priority rule based on published schedules and daily downloads of fresh flight data. This can be seen as changing the philosophy from the ‘first-come-first-served’ method of Grover Jack to a ‘first-scheduled-first-served’ method. This is seen as a more equitable procedure than Grover Jack as airlines no longer suffer the double penalty, and they are not penalized for reporting updated cancellations and delay information. Under RBS if an airline reports updated delay information about a flight that will be unable to utilize an arrival slot at an airport, that airline keeps control of the corresponding arrival slot and has the option of substituting another of its flights into that slot. Once the airline has finished the process of cancellations and substitutions, a type of inter-airline slot exchange is used to insure full utilization of all available slots.

1.1 Motivation for Problems Studied

The manner in which RBS is applied in practice for GDP planning involves certain added features. There are two sets of flights that are exempt from being assigned ground delay. The first set, flights that are airborne at the start of the GDP, obviously cannot be assigned ground delay as they have already taken off. The second set of exempt flights, though, involves a more complex motivation. An exemption radius is set around the airport experiencing the GDP. Flights outside this exemption radius are not included in the program, and thus are exempted from any ground delay. Delay is assigned only to those flights within the exemption radius. A primary reason for this second set of exempt flights is the uncertainty associated with the weather forecasts on which the GDP is based. Longer flights must serve their delays several hours in advance of their arrival at the airport. If a forecast predicts poor weather at an airport and that weather does not materialize, this could result in some longer flights receiving what, in hindsight, is unnecessary delay.

Once the duration of a GDP is set, one can easily determine the total amount of delay that must be distributed among all flights. A basic consequence of exempting fights is that the number of flights over which this total delay that is distributed has now been reduced. Thus some flights will have no delay (the exempt flights) and others will receive more delay than they would without exemptions.

The fact that predicted poor weather does not always materialize is a large factor in the distance based exemptions in RBS. This uncertainty of the weather can lead to more general inefficient utilization of the resources at an airport. Consider the example where poor weather is expected at an airport and consequently a GDP is planned for some set duration. Our knowledge of this bad weather, particularly of how long it will last, at best would take the form of a probability distribution. Thus, there is a significant possibility that the poor weather will not last for the planned GDP duration. If the time is longer than was initially expected, then the GDP can simply be extended and appropriate actions can be taken. Conversely, if the poor weather clears up earlier than anticipated, then the GDP will be cancelled early and the airport capacity will rise back to nominal conditions. However, the ability to take advantage of the possible increase in capacity at the airport due to the weather clearing up earlier than anticipated depends significantly on the manner in which GDPs are planned and controlled.

Vossen et al. (Vossen et al., 2003), (Vossen and Ball, 2006) showed that RBS without exemptions is an allocation method that meets three important metrics of equity. First, it minimizes total delay. Second, it lexicographically minimizes the vector giving the distribution of flight delays. This means that if *D* is the maximum number of minutes of delay assigned to any flight, and a_i is the number of flights receiving i minutes of delay, for i = 0, 1, 2, …, D, then RBS lexicographically minimizes (a_D, …, a_1, a_0). Also, for any flight k_1, the only way to decrease the amount of delay it receives from RBS is to increase the amount of delay given to another flight k_2 to a value greater than the amount of delay that k_1 receives. These can be seen as fundamental notions of equity (Young, 1994) applied within the ATFM context. It is also the case that the ATFM community has agreed that RBS produces a fair allocation. For these reasons, the "pure" RBS allocation (without any exemptions) will be used as the "ideal" allocation in terms of equity in our analysis.

The attempt to search for efficient solutions in the presence of weather uncertainty comes with the repeated question of how to ensure that such a solution remains equitable, or even how to define an equitable solution in such situations. Assuming the "pure" RBS solution is deemed the most equitable, defining equity metrics or objective functions remains a challenge. For example, should one seek to minimize the total deviation of all flights from their RBS allocations, or should one seek to minimize the maximum deviation of any flight from its RBS allocation? Other possibilities also exist.

There is a close relationship between the work on en route ATFM and the work on GDPs. The concern in both areas deals with the situation where demand exceeds capacity for an extended period of time. The FAA recently instituted airspace flow programs (AFPs) which use many of the GDP constructs to address en route congestion problems. AFPs restrict flow through a region of airspace, called a Flow

Constrained Area (FCA). With the bad weather occurring at an FCA instead of at an airport, flights have an additional option: routing around the FCA.

1.2 Integer Programming

Many ATFM problems involve discrete choices and thus can be modeled as combinatorial optimization problems. These are problems where the set of feasible solutions is a discrete set and the goal is to find the best solution in this set. Many combinatorial optimization problems have been shown to be NP-Hard, which means they are computationally difficult and polynomial time algorithms are unlikely (Garey and Johnson, 1979).

Bertsimas and Stock Patterson proved that the Air Traffic Flow Management Problem (TFMP), which considers the release times of aircraft as well as the optimal speed adjustments of aircraft while airborne for a network of airports taking into account the capacitated airspace, with all capacities equal to 1 is NP-Hard (Bertsimas and Stock Patterson, 1998). Much research is then given towards heuristics and approximation algorithms for NP-Hard problems. Approximation algorithms produce in polynomial time a feasible solution whose objective function is within a guaranteed factor of the optimal solution. This factor is called the approximation ratio (Vazirani, 2001).

Many combinatorial optimization problems can be formulated as integer programming (IP) problems. This formulation allows IP techniques to be used to develop algorithms and approximation algorithms. One important technique is the linear programming (LP) relaxation. The LP-relaxation of an integer program is a relaxation where the integrality constraints on the variables are removed. Linear programming problems have been proven to be solvable in polynomial time. An important class of IPs are those with totally unimodular (TU) constraint matrices. A matrix is TU if every square sub-matrix of A has determinant +1, -1, or 0. Minimum cost network flow problems, for example, have TU constraint matrices (Bertsimas and Tsitsiklis, 1997). LPs with TU constraint matrices will have integer optimal solutions as long as the right hand side vector is integer. The approximation ratio for an LP-relaxation, also called the integrality gap, is the supremum (infimum) of the ratio of the optimal integral and fractional solutions if it is a minimization (maximization) problem.

Another important feature of linear programming is duality theory. The dual of a linear program is a second linear program that finds a bound on the objective function of the original LP. This dual is formulated so that every feasible solution to the dual provides a bound on the primal objective function. The weak duality theorem says that the optimal objective function value for a minimization problem is always an upper bound for its dual. Correspondingly, the optimal objective function value for a maximization problem is always a lower bound for its dual. The strong duality theorem says that if the primal has a finite optimal, then the dual has a finite optimal with an objective function that matches the primal. These theorems can also be used to prove when the LP-relaxation of an IP formulation results in an integer solution.

In general, there can be many alternative formulations for the same IP, i.e. many different sets of constraints can define the same set of integer solutions. The strength of an IP formulation is a way of measuring how close the polyhedron for the constraint matrix is to the convex hull of the integer feasible solutions. Given a set of points $X \subseteq \mathbb{R}^n$, an inequality is called valid for X if it is satisfied by every member of X. Given a valid inequality of the polyhedron X, $\pi^T x \leq \pi_0$, the set $F = \{x \in X : \pi x = \pi_0\}$ is called a face of X, where π and x are both vectors in $\mathbb{R}^n$ and π_0 is a scalar. A facet of the polyhedron X is a face of X whose dimension is one less than the dimension of X. The facets of the convex hull of integer feasible solutions become very important because, if a formulation consists of enough facets of the convex hull of integer feasible solutions, it may be possible to solve the IPs using the LP-relaxation even with a constraint matrix that is not TU.

One technique used to model uncertainty in IPs is to formulate two-stage stochastic IPs. Here, there are two sets of decisions that are being made around some uncertain event. The second set of decisions is influenced by the uncertain event, while the first is not. In order to formulate the uncertainty, there are generally a set of possible scenarios, each with its own probability of occurrence. Ball et al. (Ball et al., 2003) and Richetta and Odoni (Richetta and Odoni, 1994) both used stochastic IPs to handle uncertainty associated with GDPs.

See (Bertsimas and Tsitsiklis, 1997), (Wolsey, 1998), or (Birge and Louveaux, 1997) for more general information on linear programming, integer programming and stochastic programming.

1.3 IP Approaches to the GDP

The GDP is a well studied problem in aviation research. The problem of assigning Ground Delay was first formulated as an IP by Odoni in 1987 (Odoni, 1987). Later, Vranas, et al. formulated a model which considered GDPs amongst multiple airports

(Vranas, Bertsimas and Odoni, 1994). This model was later extended to the full airspace by Bertsimas and Stock-Patterson (Bertsimas and Stock Patterson, 1998). These references only considered varying the timing of flights. A later formulation by the same authors also considered the option of rerouting aircraft (Bertsimas and Stock Patterson, 2000). These were all deterministic models, which do not take into account any uncertainty, like that brought about by the weather.

Richetta and Odoni (Richetta and Odoni, 1994) proposed the first IP model to solve stochastic GDPs. In this model, the goal was to minimize the cost of ground delay and the expected cost of airborne delay to all flights included in the GDP. Classes of flights are considered instead of individual flights. The model assumes that the cost of delaying two flights in the same class is equal. The airborne delay is assumed to be uniform for all flights. The random variable is assumed to be the airport capacity, and in each scenario there is an assumed Airport Arrival Rate (AAR), the number of flights the airport can handle for each arrival interval. The model returns the number of flights of each class that should receive ground delay and the expected number of flights that should receive airborne delay.

Ball et al. (Ball et al., 2003) then introduced a stochastic formulation which was a simplification of the Richetti-Odoni model. The model takes as input an AAR distribution, and produces a planned AAR (PAAR) vector, which is the number of flights that the airport should schedule to arrive in each time period, given the stochastic nature of the weather and the probabilities of different AARs. The authors showed that the special structure underlying their problem led to a totally unimodular (TU) constraint matrix. By only fixing planned arrival rates, their model allowed individual flight delays to be assigned (later) by CDM processes. Inniss and Ball (Innis and Ball, 2004) developed a procedure for deriving the AAR distribution which can be used as input in the Ball et al. model.

Kotynek and Richetta later showed that the Richetta and Odoni model (Richetta and Odoni, 1994), could also be used to determine the PAAR vector. They also answered some open questions about the Richetta-Odoni model, such as proving that its constraint matrix was not TU, but providing sufficient conditions for the IP to return integer solutions.

Both these models operate under the condition of weather uncertainty. Due to the excessive costs of airborne holding when compared to that ground holding, both papers try to avoid the situation where the airport has more flights seeking to land than it has landing slots available in a given time period. These two models though, are static-stochastic models, in the sense that once decisions are made on ground and airborne delays at the beginning of a GDP, the models do not consider the possibility of changing those decisions once the random variable is realized.

In contrast to the models described above, the first dynamic stochastic IP to model GDPs was formulated by Mukherjee and Hansen (Mukherjee and Hansen, 2007). This is a multi-stage model which takes into account the possible changes the weather can take throughout the duration of the GDP. The model is called dynamic because each possible change in weather, brings an opportunity to adjust the amount of delay given to flights. It generates a scenario tree to capture all the possible changes in weather outcomes. This scenario tree can grow large in size and can make the IP computationally inefficient.

In (Ball, Hoffman and Mukherjee, 2010), Ball et al. consider the problem of maximizing the throughput into the airport. Here, the Ration-by-Distance (RBD) algorithm is proposed. This algorithm is based on the principle of assigning longer flights to earlier slots. The authors prove that the RBD algorithm minimizes total expected delay if the GDP cancels earlier than anticipated, i.e. it allows for operators to reduce the amounts of ground delay some flights experience. Thus this model can be viewed as a dynamic stochastic model and is different from the static-stochastic models of Richetta-Odoni and Ball et al. Unlike the static-stochastic models it addresses the possibility of reassigning flights to the newly available slots once there is a change in the AAR. In their proof, the authors were able to compare the total expected delay of the RBD allocation with that of other allocations and show optimality.

1.4 Contents and Research Contributions

The RBD algorithm maximizes the expected utilization of an airport in the event of uncertain capacity increase. However, similar uncertainty affects the decision making for the entire airspace. For these areas, the proposal of an algorithm similar to RBD that has the same efficiency would be ideal, but may become a daunting task as each problem has its own individual assumptions and inputs. Instead, this dissertation builds on what the RBD algorithm brings to the table by providing IP formulations for the airport problem (treated by RBD) but then enhances these formulations to address more general problems.

Chapter 2 describes an IP model that minimizes total expected delay in the case that a GDP ends earlier than anticipated. Once this model is constructed, the

RBD algorithm is shown to produce an optimal solution for its LP-relaxation. This is an important result because the problem is in general a multi-stage stochastic IP. Previous work that attempted to model this uncertainty formulated IPs that are larger in size, while not as strong. IPs that can be solved by their LP-relaxations are not common. One class of such IPs is those with TU constraint matrices. The formulations presented in Chapter 2 are shown not to have this property, thus belonging to an even smaller class of IPs. Other models which are equivalent in strength, but smaller in size are also provided and their performance is compared.

Chapter 3 models issues of equity in GDP planning and the potential trade-offs between equity and efficiency. The RBD solution may seem unfair to some airlines, particularly those with many short haul flights. If one considers the "pure" RBS allocation as the ideal allocation (perfect equity), then with the exception of a few extreme cases, the RBD allocation represents a deviation from this ideal allocation. This chapter shows that the IP models presented in Chapter 2 can be modified in various ways to address issues concerning equity and fairness. Heuristics are also developed which attempt to capture the essence of the RBD algorithm while also insuring a limit on the deviation from the most equitable solution. These heuristics provide near optimal solutions, with the guarantee of integrality.

Chapter 4 considers the problem of severe weather in other areas of airspace. The RBD algorithm was originally proposed to maximize airport throughput in the event of weather uncertainty. Similar questions are raised when the area of uncertainty is an FCA instead of an airport. A model is presented in Chapter 4, which builds upon the model from Chapter 2 as well as other models already in literature.

This model is then compared to the models in literature and shown to be stronger and smaller in size and thus able to handle more flights and a larger set of possible weather clearance times.

Chapter 5 presents conclusions and future work.

Chapter 2 Ground Delay Programs with Weather Uncertainty

A primary objective of ATFM is to ensure the safe and orderly flow of aircraft through airspace, while minimizing the impact of delays and congestion on airspace users. Much of this delay and congestion is caused by the vulnerability of the airspace to changes in the weather, which can lower the capacities of different regions of airspace. Combine this uncertainty with the size of the airspace system and the result is a very complex system. This makes the development of efficient algorithms to solve ATFM problems an important and active area of research.

Much of the delay in the airspace system is due to bad weather. Weather decreases the capacity of arrivals and departures that an airport or a region of airspace can handle. These lower capacities cause some of the flights whose route consists of the troubled area to experience delays. The increased delay can be served on the ground before the flights depart or in the air. When a GDP is instituted at an airport with reduced capacities, flights scheduled to arrive at this airport are given a delay in minutes to be served before they depart their origin airports. The inputs to these GDPs are the airport capacities over some pre-specified time period and the flight schedules. The flow of aircraft into the airport is then adjusted to meet the capacities for the duration.

A GDP must be planned several hours in advance. To accomplish this, weather forecasts are converted into profiles of AARs for 15 minute periods. These are the number of aircraft that can land at a particular airport in a period. This partitions each period into arrival slots of equal time length which are then assigned to the flights. Because these capacities are based on the weather forecasts several hours in advance, there is a high degree of uncertainty with these capacities. The uncertainty can be characterized by using a discrete AAR distribution represented by a set of AAR vectors $\left\{\left(A_q^1,...,A_q^T\right): q=1,...,Q\right\}$ with probabilities p_q for $q=1,...,Q$.

The CDM philosophy considers the allocation of capacity to be an allocation of airport arrival slots to airlines instead of an allocation of arrival slots to individual flights. This notion of slot ownership is one of the main tenets of the CDM paradigm. A general consensus among airlines was reached that RBS was indeed a fair method of rationing arrival capacity (Vossen et al., 2003) (Vossen and Ball, 2006). RBS orders flights according to increasing scheduled arrival times.

In (Ball, Hoffman and Mukherjee, 2010), Ball et al. consider the problem of maximizing the throughput into the airport. Here, the RBD algorithm is proposed. This algorithm is based on the principle of assigning longer flights to earlier slots. The authors prove that the RBD algorithm minimizes total expected delay if the GDP cancels earlier than anticipated. In their proof, the authors were able to compare the total expected delay of the RBD allocation with that of other allocations and are able to show optimality. RBD is structurally very similar to RBS. The only difference is that flights are ordered by increasing flight length rather than increasing scheduled arrival time.

Example 2.1

To illustrate the differences posed by these two approaches, consider the following example of how the RBS and RBD algorithms would allocate flights to slots in a ground delay program. Consider first, Table 2.1, which gives us the relevant flight information:

Flight(k)	Published Arrival Time, $arr(k)$	Length, $len(k)$	Published Departure Time, $dep(k)$
1	4:56	60	3:56
2	4:57	65	3:52
3	4:58	75	3:43
4	4:59	90	3:29
5	5:00	120	3:00

Table 2.1: Input Chart for Example 2.1

Suppose that the airport has a reduced number of landing slots, allowing a flight to land every five minutes. This amounts to arrival slots being available at 5:00, 5:05, 5:10, 5:15, 5:20, and later times. This is reduced from a nominal capacity where a flight is allowed to land every minute. Since it would be inefficient to allocate a flight to an arrival slot later than 5:20 in this GDP, there is only a need to consider these five arrival slots. Based on these assumptions, the RBS and RBD allocations are given in Figure 2.1, where the red lines are the assignments of the RBS algorithm and the blue lines are those of the RBD algorithm:

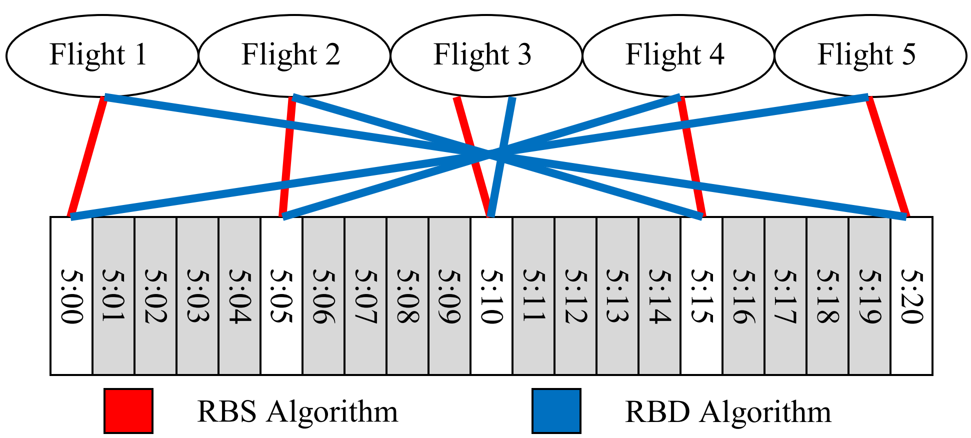

Figure 2.1: RBS and RBD Solutions to Example 2.1

Consider now the controlled departure times of flights under the different algorithms and consider how the formulations will perform if the GDP cancels earlier than anticipated. Table 2.2 lists the departure times of the flights under the different algorithms.

Flight	Length	*arr(k)*	RBS Arr	RBS Dep	RBD Arr	RBD Dep
1	60	4:56	5:00	4:00	5:20	4:20
2	65	4:57	5:05	4:00	5:15	4:10
3	75	4:58	5:10	3:55	5:10	3:55
4	90	4:59	5:15	3:45	5:05	3:35
5	120	5:00	5:20	3:20	5:00	3:00

Table 2.2: Arrival and Departure Times for RBS and RBD Algorithms

During GDPs, it is usually the case that the duration of the bad weather (lower airport arrival rates) is not known with certainty. If the weather suddenly clears, then the GDP will be cancelled. However, it can be difficult to take advantage of a capacity increase at an airport since this is done by releasing flights currently on the ground and such flights must travel (usually an hour or more) before they can reach the destination airport. The efficiency of the RBS and RBD solutions under different GDP cancellation times can now be considered. Consider the following GDP cancellation times: 3:00, 3:15, 3:30, 3:45, 4:00, 4:15, and 4:30.

Table 2.3 shows how the formulations perform under the different cancellation times, where capacity for the arrival slots in Figure 2.1 is increased to 1 after the GDP is cancelled (i.e. every slot after the cancellation time has its capacity rise to 1), and each column of Table 2.3 measures the total delay if the GDP is cancelled at the mentioned time.

If the GDP is cancelled at 3:45, then the following transpires.

- By Table 2.2, we can see that Flights 4 and 5 have departed under both algorithms. This implies that there can be no change in the originally assigned ground delays. Thus the RBS algorithm will assign these flights 16 and 20 minutes of delay respectively, whereas the RBD solutions will give these flights 0 and 6 minutes respectively. Because the GDP was cancelled, Flight 3 can depart immediately and land at 5:00 under the RBS algorithm, and at 5:01 under the RBD algorithm because flight 5 is already arriving at 5:00 under the RBD algorithm. Flights 1 and 2 do not receive any delay under either algorithm in this cancellation time.
- The total delay for other cancellation times is computed through similar measures.

	3:00	3:15	3:30	3:45	4:00	4:15	4:30
RBS	0	15	21	38	60	60	60
RBD	0	0	2	9	32	56	60

Table 2.3: Total Delay for the RBS and RBD Algorithms

Table 2.3 gives the total delay achieved under each algorithm under various cancellation times.

Although this is a simple example, the RBD solution has equal or less delay than the RBS solution in all seven scenarios presented here. This illustrates the result of Ball et al., who showed that the RBD solution minimizes the total expected delay if a GDP cancels earlier than anticipated (Ball, Hoffman and Mukherjee, 2010). In this chapter, this same problem of maximizing expected throughput (i.e. minimizing total expected delay) into an airport during a GDP that has an uncertain cancellation time is considered. This will be modeled as an integer program with hopes that this IP will allow us to consider more complex problems that address similar issues as allocating scarce resources subject to a possible increase in capacity at some later uncertain time.

2.1 Formulation

A general approach to modeling weather uncertainty IPs is to use a multi-stage scenario tree that tracks weather changes over time. The scenario tree represents points in time and states of nature. For example, the storm might move, get worse, or change in forecast. Each node in the scenario tree would represent a decision point in time when the decision of how to reassign flights to arrival slots needs to be considered, given the updated weather forecast or weather conditions. This sets the problem up as a multi-stage stochastic program e.g. as done in (Mukherjee and Hansen, 2007).

In order to achieve a more compact scenario tree, we employ a fairly simple model of weather states and decision dynamics. We assume the weather has only two possible states: clear and not clear. This is actually generally consistent with how GDPs are handled in practice, where a GDP is not cancelled until the weather clears.

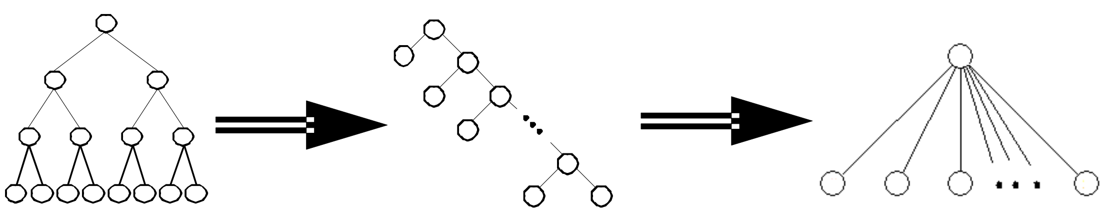

Figure 2.2: Two-Stage Structure of Problem

Figure 2.2 shows how this assumption turns the problem from having a multi-stage scenario tree into one with a two-stage scenario tree. Each node in the far left tree represents the condition of weather at a given time in the day. The assumption of weather having only two states changes the structure of the scenario tree from a general multi-stage structure to a skewed multi-stage structure. This skewed multi-stage scenario tree can then be replaced by a two-stage scenario tree by changing the random variable from the condition of the weather at a given time period to the time when the weather clears. This collapses the scenario tree and allows the problem to be formulated as a two-stage stochastic IP instead of as a multi-stage stochastic IP.

A second assumption is that there is no lag between weather clearance time and the time the airport goes back to nominal capacity. The assumption is that this happens immediately. This is to mimic the practice of cancelling a GDP, where once the GDP is cancelled the capacity at the hosting airport is increased.

A third assumption is that the possible weather clearance times and the times we can change our decision coincide. Thus, we do not change our decisions based on changes in the forecast. More generally, we also assume that the distribution does not change, e.g. due to a forecast change.

The input to the model comes from two sources: flight-based input and airport-based input. The flight-based input includes a set of flights, *Flights*, with the following provided for each flight $k \in Flights$:

- The stage length of the flight *k*, *len(k)*
- The published arrival time of the flight *k*, *arr(k)*
- The arrival slot that the flight *k* would receive in the RBS allocation, RBS(*k*).

The airport-based input includes:

- The maximum duration of the GDP
- The reduced capacity of the airport, $cap_1(i)$ for each initial (stage one) slot i, i.e. the number of flights that can land in time period i when the capacity is reduced.
- There are T possible GDP endings (cancellations). Each cancellation $t = 1, \ldots, T$, has an associated time, $\tau(t)$. The GDP end time $\tau(t)$ will be referred to as scenario t.
- The nominal capacity of the airport, $cap_2(j,t)$ for each slot j in scenario t. (We assume that for each slot j in each scenario t, $cap_2(j, t) \geq cap_1(j)$).
- A probability p_t for each scenario $t = 1, \ldots, T$.

Slot i	$time(i)$	$cap_1(i)$	$cap_2(j, t)$
1	6:00	1	1
2	6:01	0	0
3	6:02	1	1
4	6:03	0	0
5	6:04	1	1
			GDP Cancelled at 6:05
6	6:05	0	**2**
7	6:06	1	**2**
8	6:07	0	**2**
9	6:08	1	**2**

Table 2.4: An Example of Stage One and Stage Two Capacities under a Given GDP Cancellation Time

Table 2.4 gives a possible situation where a GDP is cancelled and associated capacities. Each of slots 6, 7, 8, and 9 now have their capacity increased to 2. This capacity can now be utilized by reducing the ground delay of flights.

This problem will be formulated as a two-stage stochastic IP. The slots in stage one will be labeled by the set $Slots_1(i)$, generally indexed by i. Likewise, the slots in scenario t of stage two will be labeled by the set $Slots_2(j, t)$, generally indexed by j and t. There will also be a time associated with each slot (in stage one or stage two). The functions $time(i)$ will indicate the start time of the slot i, and $time(j, t)$ will indicate the start time of slot j in scenario t. In a GDP, every flight must initially be assigned to a slot, and the first stage models these actions. What follows next is a description of this first stage.

Let $x_{k,i}$ be the binary variable which is one if flight k is initially assigned to the arrival slot i. Similarly, the variable $x_{s,i}$ is the integer variable which is the amount of unused capacity for slot i. Then the following three constraint sets model the stage one restrictions. These constraints are very similar to the model proposed by Odoni (Odoni, 1987), where $\mathbb{Z}$ is the set of integers.

$$\sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} x_{k,i} = 1 \text{ for each flight } k \tag{2.1}$$

$$\sum_{\substack{k \in Flights \\ arr(k) \leq time(i)}} x_{k,i} + x_{s,i} \leq cap_1(i) \text{ for each arrival slot } i \tag{2.2}$$

$$\sum_{i \in Slots_1} x_{s,i} = \sum_{i \in Slots_1} cap_1(i) - |Flights| \tag{2.3}$$

$$x_{k,i} \in \{0,1\} \text{ for all } k,\ i$$

$$x_{s,i} \geq 0, x_{s,i} \in \mathbb{Z}$$

Each flight has a scheduled arrival time, $arr(k)$, and constraint set (2.1) ensures that each flight is assigned to some arrival slot after its scheduled arrival time.

Constraint set (2.2) ensures that no slot is utilized in excess of its capacity during the GDP. Constraint set (2.3) ensures that every arrival slot has its demand met by having the slack flight supply these slots. This completes stage one of the formulation.

Example 2.2

Consider the following three flight input for a GDP at an airport with a reduced capacity of one flight every three minutes, i.e. *time*(1) = 7:30, *time*(2) = 7:33, and *time*(3) = 7:36.

Flight(k)	$arr(k)$	$len(k)$
1	7:29	30
2	7:30	45
3	7:32	60

Table 2.5: Input Table for Example 2.2

Under this input, stage one of this IP would be as follows:

$$x_{1,1} + x_{1,2} + x_{1,3} = 1$$
$$x_{2,1} + x_{2,2} + x_{2,3} = 1$$
$$x_{3,2} + x_{3,3} = 1$$
$$x_{1,1} + x_{2,1} = 1$$
$$x_{1,2} + x_{2,2} + x_{3,2} = 1$$
$$x_{1,3} + x_{2,3} + x_{3,3} = 1$$
$$x_{s,1} + x_{s,2} + x_{s,3} = 0$$

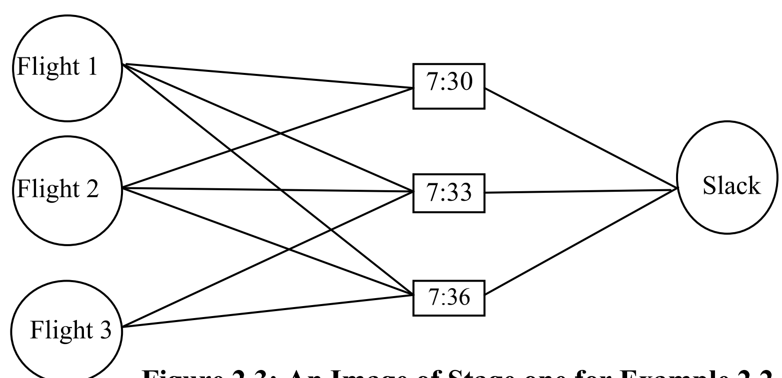

Figure 2.3: An Image of Stage one for Example 2.2

What follows next is a presentation of stage two of the formulation. Here, some flights have already been given delay, but the amount of delay a flight actually experiences is determined by both the slot to which the flight is assigned in stage one and the time of weather clearance, $\tau(t)$.

Each scenario is constructed as an assignment problem on a bipartite graph. Each arc of stage one (k, i) (except those from the slack flight) will become a node in every scenario t of stage two. These nodes, (k,i,t), represent the reallocation possibilities that the corresponding stage one arc provides. There are also nodes for each slot available in each scenario of stage two. There is also a slack flight (s_t,t) in each scenario of stage two to ensure that supply equals demand and a slack slot (d_t,t) to ensure that nodes (k,i,t) do not send flow unless the corresponding arc (k,i) in stage one receives flow.

Each stage two node (k,i,t) representing a stage one arc (k, i) has arcs connecting it to the slack slot (d_t,t), as well as the possible slots to which it can be reallocated. This is the set of non-slack stage two slots that are no earlier than both the flight k's original arrival time, $arr(k)$, as well as the earliest slot the flight can reach by departing immediately at time $\tau(t)$ if the flight had not yet departed, or i if the flight had already departed by time $\tau(t)$. More precisely, the set of stage two slots to which flight k can be reassigned under scenario t, assuming it was initially assigned to slot i is given by:

$$Feas(k,i,t) = \left\{ j \in Slots_2 \mid time(j,t) \geq arr(k) \text{ and } time(j,t) \geq \min\left\{i, t + len(k)\right\}\right\}$$

In the formulation, there are binary variables, $w_{k,i,j,t}$, that are one if and only if the flight k was initially assigned to slot i in stage one and then the flight k is reassigned to slot j in stage two under the scenario that the GDP is cancelled at time t. The $w_{s_t,j,t}$ variables are integer variables that are nonzero if and only if the slot j does not have all its capacity met by the non-slack flights k in scenario t. This means that some of the demand of this slot must be met by the slack flight s_t. The $w_{k,i,d_t,t}$ variables are binary variables that are one when the flight k is not initially assigned to the slot i.

Constraint set (2.4) says that each stage two node (k,i,t) representing the stage one arc (k, i) in scenario t must be assigned to a stage two slot in the same scenario. Constraint set (2.5) says that in each scenario, t, each stage two slot (j, t) must have enough flights assigned to it to meet its capacity in that scenario. These can either be a typical flight (k,i,t) or the slack flight (s_t, t). Constraint set (2.6) says that each stage two slack flight must meet the demand of the stage two slots of that scenario that are not met by the flights in that scenario. Constraint set (2.7) says that each stage two slack slot, denoted by d_t, has a demand equal to the total number of arcs in the stage one network minus the number of flights. These constraints make stage two into T distinct simple transportation problems.

Because stage two can be seen as the reallocation stage, it also must be ensured that no stage one arc is reallocated unless it is used in stage one. This is achieved by the slack slot which is added to each scenario. Every node (k,i,t) will have an arc connecting it to the slack slot, but constraint (2.8) will force the flow to this slack slot to depend on the flow the corresponding arc received in stage one.

$$\sum_{j \in Feas(k,i,t)} w_{k,i,j,t} + w_{k,i,d_t,t} = 1 \quad \text{for each } k,i,t \tag{2.4}$$

$$\sum_{k \in Flights} \sum_{\substack{i \in Slots_1 | \\ j \in Feas(k,i,t)}} w_{k,i,j,t} + w_{s_t,j,t} = cap_2(j,t) \tag{2.5}$$

for each stage-two slot j and scenario t

$$\sum_{j \in Slots_2} w_{s_t,j,t} = \sum_{j \in Slots_2} cap_2(j,t) - |Flights| \text{ for all scenarios } t \tag{2.6}$$

$$\sum_{k \in Flights} \sum_{i \in Slots_1} w_{k,i,d_t,t} = \left| \left\{ (k,i): \begin{array}{c} k \in Flights, i \in Slots_1 \\ \text{and } time(i) \geq arr(k) \end{array} \right\} \right| - |Flights| \tag{2.7}$$

$$w_{k,i,d_t,t} + x_{k,i} = 1 \text{ for each } (k,i) \text{ feasible to stage one and each scenario } t \tag{2.8}$$

$$w_{k,i,j,t}, w_{k,i,d_t,t} \in \{0,1\} \text{ for all } k,\ i,\ j,\ t$$

$$w_{s_t,j,t} \geq 0, w_{s_t,j,t} \in \mathbb{Z}$$

Example 2.3

An illustration of stage two for the Example 2.2 is produced in Figure 2.4. The solid arcs represent feasible reallocations.

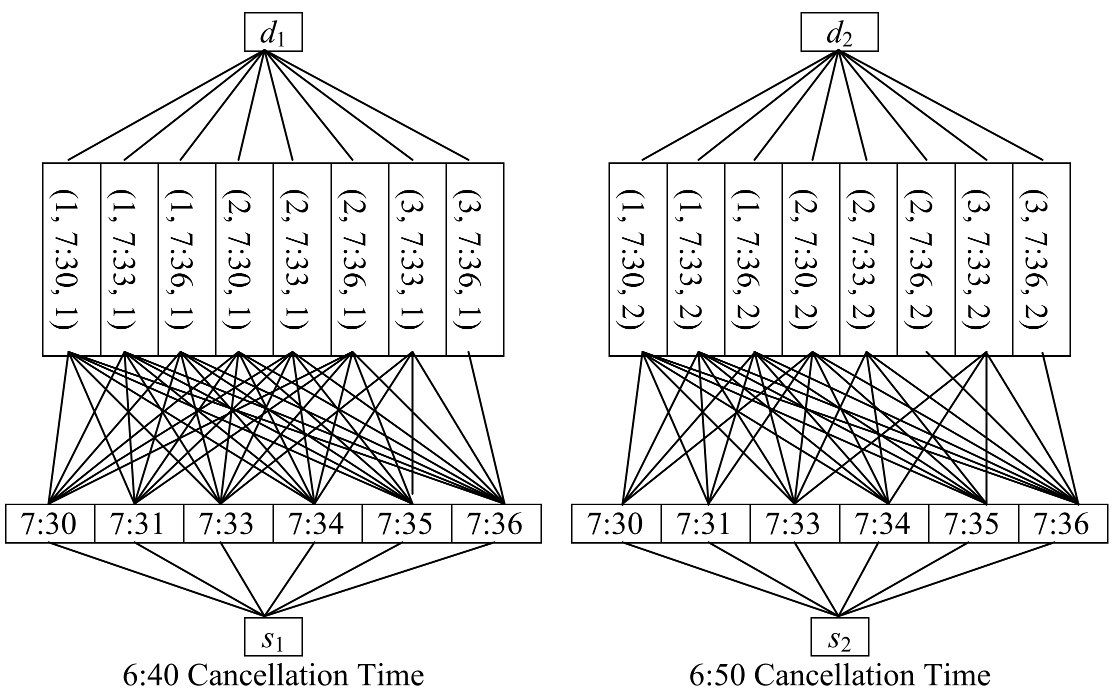

Figure 2.4: Example of stage two with two scenarios

The objective is to minimize the total expected delay. This can be measured as the amount of delay incurred by each flight in each stage two scenario multiplied by the probability of that scenario occurring, q_t, summed over all flights and all scenarios, which can be written as

$$\sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{j \in Slots_2} \sum_{t \in Scenarios} \left(p_t \left(time(j,t) - arr(k) \right) w_{k,i,j,t} \right) \tag{2.9}$$

2.2 Proof of Optimality of the RBD Algorithm

The results of (Ball, Hoffman and Mukherjee, 2010) imply that the RBD algorithm generates a stage one optimal solution to the IP defined in section 2.3, which we will refer to as the Two-Stage Stochastic Dynamic GDP (TSDG). Here, we will show that the RBD solution also solves the LP-relaxation of this model. We will do this by showing that a solution inspired by the RBD algorithm satisfies conditions on optimality given by Linear Programming theory. First, consider the following theorem from linear programming.

Theorem (**Weak Duality**): If X_0 is a feasible solution to the primal minimization problem, minimize $z = C^T X$ subject to $AX \geq B$, $X \geq 0$ and W_0 is a feasible solution to the dual maximization problem, maximize $z = B^T W$ subject to $A^T W \leq$ C, W ≥ 0, then $C^T X_0 \geq B^T W_0$.

A simple corollary to this theorem is that if X_0 and W_0 are feasible solutions to the primal minimization problem and dual maximization problem respectively, with

$C^T X_0 = B^T W_0$, then X_0 and W_0 are optimal solutions to the primal and the dual respectively. (Bertsimas and Tsitsiklis, 1997).

One obvious fact about this formulation is that stage one and each scenario of stage two (when considered independently of one another) is a transportation problem. These transportation problems are linked together by constraint set (2.8), which prevents the problem from being a large set of disjoint transportation problems. However, it is easy to see that this formulation has complete recourse, i.e. for any solution feasible to stage one, there exists a feasible solution to stage two (Birge and Louveaux, 1997). For example, given a solution to stage one, the only stage two constraint set where the variables from stage one appear is in constraint set (2.8). These constraints ensure that the nodes (k,i,t) representing the arcs (k, i) of stage one in scenario t are not reallocated unless they were used in stage one. Since we assume $cap_2(j,t) \geq cap_1(j)$, the solution to stage one immediately provides a feasible solution to stage two. For each scenario, once constraint (2.8) is satisfied, the remaining nodes that still have supply and demand are the nodes representing stage one arcs that receive flow and the stage two slots, as well as the slack flight.

2.2.1 **Stage two Dual Feasible Solution**

If the vector x is a solution to stage one, then define w_x as the optimal solution to the stage two problem generated by x. If a stage one solution is an RBD solution, define the vector (x, w_x) an *RBD-inspired solution*. This RBD-inspired solution is used to construct a dual solution and show its feasibility.

The dual of this LP is:

$$\max\left(\begin{array}{l}\sum_{k\in Flights} u_k + \sum_{i\in Slots_1} cap_1(i)v_i + \left(\sum_{i\in Slots_1} cap_1(i) - |Flights|\right)u_s + \\ \sum_{t\in Scenarios}\left(\sum_{k\in Flights}\sum_{i\in Slots_1}\lambda_{k,i,t} + \sum_{j\in Slots_2} cap_2(j,t)\delta_{j,t}\right) + \\ \sum_{t\in Scenarios}\left(\left(\sum_{j\in Slots_2} cap_2(j,t) - |Flights|\right)\lambda_{s_t,t} + \left(|Arcs_1| - |Flights|\right)\delta_{d_t,t} + \sum_{k\in Flights}\sum_{i\in Slots_1} y_{k,i,t}\right)\end{array}\right)$$

subject to

$u_k + v_i + \sum_{t\in Scenarios} y_{k,i,t} \le 0$ for each feasible arc (*k*, *i*) (2.10)

$u_s + v_i \le 0$ for each arc (*s*, i) (2.11)

$\lambda_{k,i,t} + \delta_{j,t} \le q_t\left(time(j,t) - arr(k)\right)$ for each feasible arc ((k,i,t), (*j*, *t*)) (2.12)

$\lambda_{k,i,t} + \delta_{d_t,t} + y_{k,i,t} \le 0$ for each arc ((k,i,t), (d_t, *t*)) (2.13)

$\delta_{j,t} + \lambda_{s_t,t} \le 0$ for each arc ((s_t, *t*), (*j*, *t*)) (2.14)

A dual feasible solution will be constructed through spanning trees in each scenario of stage two. In order to accomplish this though, the allocation nodes in stage two need to be classified into two different types of allocations in a scenario. A similar distinction must be made between the different types of slots in stage two. The stage two node (k,i,t), which represents the stage one arc (*k*, *i*) is called a *supply 1 node* if $x_{k,i} = 1$ in the primal (where *k* is not the stage one slack flight). Otherwise (k,i,t) is *supply 0 node*. A stage two slot (*j*, *t*) is a *fully used stage two slot* if $\sum_{k\in Flights}\sum_{i|time(i)\ge arr(k)} w_{k,i,j,t} = cap_2(j,t)$ in the primal solution, where *k* is not the stage one slack flight. A stage two slot (*j*, *t*) is a *partially used stage two slot* if

$\sum_{k \in Flights} \sum_{i | time(i) \geq arr(k)} w_{k,i,j,t} > 0$ and $\sum_{k \in Flights} \sum_{i | time(i) \geq arr(k)} w_{k,i,j,t} < cap_1(j,t)$ in the primal solution. Otherwise (j, t) is an *unused stage two slot*. Denote the set of fully and partially used stage two slots in scenario t by $used(t)$.

Define an *RBD-arc* as a stage one arc (k, i) such that $x_{k,i} = 1$ in the RBD solution. Define an *RBD-Inspired arc* for an RBD arc (k, i) as a stage two arc (k, i, j, t) with $w_{k,i,j,t} = 1$.

Given a stage one solution, e.g. the RBD solution, there are T disjoint stage two transportation problem networks. For an RBD-Inspired solution, define a *stage two comp* as a set of used slots in the scenario t of stage two such that if (j_1, t) and (j_2, t) are in the same comp, then every slot (j', t) such that $time(j_1,t) \leq time(j',t) < time(j_2,t)$ is a fully used stage two slot. Any feasible solution will generate a set of comps, but we are only interested here in those generated by an RBD-Inspired solution. Suppose that there are $Comps_2(t)$ such comps in the scenario t, indexed by $\beta_t = 1, \ldots, Comps_2(t)$, and let $last(\beta_t)$ be the latest slot in comp β_t (i.e. $time(last(\beta_t),t) > time(j,t)$ for all other j in comp β_t). Note that all fully or partially used stage two slots are in a comp and an unused stage two slot is not in any comp. Notice also that a stage two comp can end in a fully or partially used stage two slot, but the only place that a partially used stage two slot can be in a comp is at the end of that comp. The node (k,i,t) representing the stage one arc (k, i) in scenario t is in comp β_t if the earliest slot that (k,i,t) can be reassigned to is in comp β_t. This is the same as saying that this node is assigned to a slot in this comp.

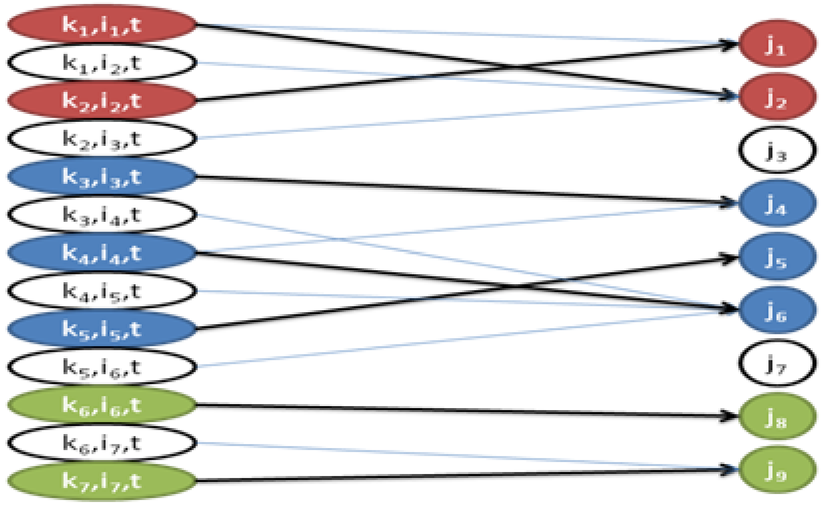

Figure 2.5: Stage Two Comps for a particular Stage One Solution

A dual feasible solution is defined by constructing trees in each scenario that span every node except the node (d_t, t). This construction begins by adding the arcs connecting the nodes (k,i,t) representing the stage one arc (k, i) in scenario t to their earliest reallocation slots in this scenario. This connection is made for both the supply 1 and supply 0 nodes. The arcs connecting the supply 1 nodes (k,i,t) to the slot (j, t) to which they are reassigned (i.e., such that $w_{k,i,j,t} = 1$) are also added to the spanning tree. This will partition the set of used stage two slots into sets of consecutively used slots, or comps. Next, an arc is added connecting the last slot in each comp, ($last(\beta_t)$, t), to the slack flight for scenario t, (s_t, t). The slack flight for each scenario can then be set equal to zero and the following stage two dual solution is obtained via the complementary slackness conditions.

$$\lambda_{s_t,t} = 0 \text{ for all scenarios, } t. \tag{2.15}$$

$$\delta_{j,t} = p_t\left(time(j,t) - time(last(\beta_t))\right) \quad \text{if } (j,\, t) \text{ is in comp } \beta_t \tag{2.16}$$

$$\delta_{j,t} = 0 \text{ if } (j,\, t) \text{ is an unused slot.} \tag{2.17}$$

$$\lambda_{k,i,t} = p_t\left(time(last(\beta_t)) - arr(k)\right) \text{ if } (k, i, t) \text{ is in comp } \beta_t \tag{2.18}$$

$$\delta_{d_t,t} = 0 \text{ for all } t \tag{2.19}$$

$$y_{k,i,t} = -\lambda_{k,i,t} = p_t\left(arr(k) - time(last(\beta_t))\right) \text{ if } (k, i, t) \text{ is in comp } \beta_t \tag{2.20}$$

Next, the feasibility of this stage two dual solution is checked in each constraint of the stage two-dual.

Lemma 2.1. The constraint $\lambda_{k,i,t} + \delta_{j,t} \le p_t(time(j,t) - arr(k))$ is satisfied by this solution.

Proof:

Case 1: (j,t) is an unused or partially used slot:

Then the above solution implies that $\lambda_{k,i,t} = p_t(time(j_\beta, t) - arr(k))$ and $\delta_{j,t} = 0$, where (k,i,t) is in comp β. Then $\lambda_{k,i,t} + \delta_{j,t} = p_t(time(j_\beta, t) - arr(k))$. The satisfaction of this constraint then depends on the relationship between j and j_β. If $j < j_\beta$, then this constraint is violated, but it also implies that there is a slot that the node (k,i,t) can be reallocated to that is earlier than any slot in its comp. This is not possible if (k,i,t) is a supply 1 node since the slot j would then be a used slot and the comp would change accordingly.

If (k,i,t) is a supply 0 node and the slot i' that the flight k is assigned to is before i, then (k,i',t) is a supply 1 node. The departure times of (k,i',t) and (k,i,t) are $time(i') - len(k)$ and $time(i) - len(k)$ respectively. Because i' is an earlier slot than i, this implies that the departure time of the arc representing (k,i',t) is before that of (k,i,t). In scenarios that are before either (*k, i*) or (*k, i*') have departed, they can be rescheduled to the same set of slots and so the earliest slot that the arc representing (k,i',t) can be reallocated to will be used by the arc representing (k,i,t). In scenarios that are after (*k, i*') has departed, (k,i,t) will not be able to be rescheduled to slot *i*', so (k,i',t) will be in an earlier comp than (k,i,t).

If (k,i,t) is a supply 0 node and the slot *i*' that the flight *k* is assigned to is after *i*, then consider the supply 1 nodes (k',i,t) and (k,i',t) with *k'* a longer flight than *k* and *i'* later than *i*. These nodes must exist because the RBD algorithm says that a longer flight *k'* must use the slot *i* and hence the flight *k* will use a later slot *i'*. The departure times of the arcs representing (k',i,t), (k,i,t), and (k,i',t) are $time(i)-len(k')$, $time(i)-len(k)$, and $time(i')-len(k)$ respectively. Because *k*' is a longer flight than *k*, the departure time of the arc representing (k',i,t) is before that of (k,i,t). Likewise, because *i* is an earlier slot than *i'*, the departure time of the arc representing (k,i,t) is before that of (k,i',t). In scenarios that are before the arcs representing (k,i,t) and (k,i',t) have departed, they can be rescheduled to the same set of slots and so the earliest slot that the arc representing (k,i,t) can be reallocated to will be used by the arc representing (k,i',t). In scenarios that are after the arc representing (k,i,t) has departed, the arc representing (k',i,t) will also have departed and both will be rescheduled to the slot *i* (implying that $j = i$). If the slot *j* is before j_β, then *j* will be a fully used slot, which contradicts this case. If $j \geq j_\beta$, then $\lambda_{k,i,t} + \delta_{j,t} \leq q_t(time(j,t) - arr(k))$ and this constraint is satisfied.

Case 2: (*j, t*) is a fully used slot:

Then the stage two dual solution implies that $\lambda_{k,i,t} = p_t(time(j_{\beta_{1,t}}, t) - arr(k))$ and $\delta_{j,t} = p_t(time(j,t) - time(j_{\beta_{2,t}}, t))$. Then $\lambda_{k,i,t} + \delta_{j,t} = p_t(time(j_{\beta_{1,t}}, t) - arr(k)) + p_t(time(j,t) - time(j_{\beta_{2,t}}, t))$. The validity of this constraint then depends on the relationship between $\beta_{1,t}$ and $\beta_{2,t}$. $\beta_{2,t}$ cannot be before $\beta_{1,t}$ by the definition of a comp (it would imply that (*k, i, t*) connects to a slot earlier than its earliest slot). So $\beta_{2,t}$ is a comp that is equal to or after $\beta_{1,t}$. This implies that $j_{\beta_{1,t}} - j_{\beta_{2,t}} \leq 0$, which implies that $\lambda_{k,i,t} + \delta_{j,t} \leq p_t(time(j,t) - arr(k))$, so this constraint is satisfied.

Q.E.D.

Constraints (2.13) and (2.14) follow immediately from the definition of the stage two dual solution. This establishes a dual feasible solution for stage two. Next this stage two dual solution will be used to help define a dual solution for stage one. Then the remaining dual constraints will be shown to be satisfied by this solution.

The objective function for the stage two dual can now be simplified with the values given to the stage two dual variables. Notice that $\partial_{p,t} = 0$ and $\lambda_{s_t,t} = 0$, so the corresponding terms in the stage two dual objective function can be eliminated. Then because $y_{k,i,t} = -\lambda_{k,i,t}$, the terms will cancel in the stage two dual, leaving the stage two dual objective function value as: $\left(\sum_{t \in Scenarios} \left(\sum_{j \in Slots_2} cap_2(j,t)\delta_{j,t} \right) \right)$

Since $\lambda_{k,i,t} = p_t \left(time(j_{\beta_t}, t) - arr(k) \right)$ if (k,i,t) is in comp β_t, define $c_{k,i} = \sum_{t \in Scenarios} \lambda_{k,i,t}$ as the *expected delay cost* for the allocation (k, i). With these new costs, a new stage one assignment problem can be formulated, where an arc (k, i) is feasible to this new problem if it is feasible to stage one of the original problem, and the cost of the arc (k, i) is $c_{k,i}$. Call this new problem the expected stage one problem (ESOP)**:**

ESOP: $\min f(x,w) = \sum_{k \in Flights} \sum_{i \in Slots_1} c_{k,i} x_{k,i}$

subject to

$$\sum_{i \in Slots_1} x_{k,i} = 1 \text{ for each flight } k \tag{2.21}$$

$$\sum_{k \in Flights} x_{k,i} + x_{s,i} = cap_1(i) \text{ for each slot } i \tag{2.22}$$

$$\sum_{i \in Slots_1} x_{s,i} = \sum_{i \in Slots_1} cap_1(i) - |Flights| \tag{2.23}$$

$$x_{k,i} \in \{0,1\}$$
$$x_{s,i} \ge 0, x_{s,i} \in \mathbf{Z}$$

2.2.2 Stage One Monge Matrix

A matrix is *Monge* if there exists an ordering of the rows and columns such that $c_{i,a} + c_{j,b} \le c_{i,b} + c_{j,a}$ whenever $i < j$ and $a < b$ (Bein et al., 1995). These matrices are named for the 18th century French mathematician Gespard Monge, who first discovered them. Table 2.6 provides a small example of a Monge matrix.

$$\begin{bmatrix} 10 & 17 & 13 & 28 & 23 \\ 17 & 22 & 16 & 29 & 23 \\ 24 & 28 & 22 & 34 & 24 \\ 11 & 13 & 6 & 17 & 7 \\ 45 & 44 & 32 & 37 & 23 \end{bmatrix}$$

Table 2.6: An example of a Monge matrix

Monge matrices have a long history in mathematics and computer science. Hoffman (Hoffman, 1963) showed that transportation problems with Monge bi-adjacency matrices (the adjacency matrix of a bipartite graph) could be solved to optimality by the Northwest Corner Method. This is an algorithm for determining feasible solutions to the transportation problem that operates by iteratively selecting the most “northern” cell in the “west-most” column of the remaining bi-adjacency matrix, setting that variable to the highest feasible value, removing those rows and columns from consideration and repeating until either there are no remaining rows or columns. For the assignment problem, this will yield the diagonal of the matrix as the optimal solution. Wilber (Wilber, 1988) showed that dynamic programming algorithms can often be solved more rapidly if their underlying weight matrix is Monge.

Consider also the traveling salesman problem (TSP). In this problem, a salesman is to visit a set of cities, $\{c_1, \ldots, c_m\}$ and return home. The input to the problem is a matrix consisting of the distance of the cities from one another; i.e. $d(c_i, c_j)$

represents the positive distance between cities c_i and c_j. The goal of the TSP problem is to find an ordering of the cities that minimizes the total distance that the salesman will have to travel.

Although the TSP is NP-Hard, when the distance matrix is Monge, the TSP problem can be solved in polynomial time (Burkard, Klinz and Rudiger, 1996).

An implicit assumption of the preceding discussion is that the matrices were dense and that the graphs were complete (or complete bipartite graphs in the case of transportation problems). We now extend the Monge property to a class of sparse bipartite graphs and matrices. Define a bipartite graph *lower-Monge* if the bi-adjacency matrix of this graph can have its rows and columns ordered such that:

1. For every row i and each pair of columns a, b with $b > a$, if (i, a) is defined then (i, b) is defined.
2. If the matrix entries (i, a), (i, b), (j, a), (j, b) are all defined with $i < j$ and $a < b$, then $c_{i,a} + c_{j,b} \le c_{i,b} + c_{j,a}$

$$\begin{bmatrix} 4 & 9 & 7 & 3 \\ - & - & 6 & 1 \\ - & 11 & 7 & 1 \\ - & - & - & 3 \end{bmatrix}$$

Table 2.7: An example of a lower-Monge matrix

The transportation problem with unit supply (TPUS) is as follows:

TPUS: $\min f(x) = \sum_{a \in Rows} \sum_{i \in Columns} c_{a,i} x_{a,i}$

subject to

$$\sum_{i \in Columns} x_{a,i} = 1 \text{ for each row } a \tag{2.24}$$

$$\sum_{a \in Rows} x_{a,i} = D_i \text{ for each column } i \tag{2.25}$$

$$x_{a,i} \in \{0,1\}$$

Proposition 2.2: If a feasible solution to TPUS with a lower-Monge constraint matrix exists, then the Northwest Corner Method finds a feasible solution.

Proof:

Let $G = (R, C, A)$ be the corresponding bipartite graph for TPUS, where R is the set of rows, C is the set of columns and $(a,i) \in A$ if the corresponding element of the cost matrix is defined. Then TPUS can be converted into a standard assignment problem by formulating a new bipartite graph $G' = (R', C', A')$, where $R' = R$ and C' is defined as follows: for each column i in TPUS with demand D_i, there are D_i nodes: $i_1,...,i_{D_i}$. An arc (a,i_α) for $\alpha \in 1,...,D_i$ is in A' if $(a,i) \in A$. Each of the rows in G' has a supply of 1, and each of the columns in D' has a demand of 1. A feasible solution to TPUS will correspond to a perfect matching on G'.

Hall's theorem (Halmos and Vaughan, 1950) states that a bipartite graph $G = (X,Y,E)$ has a perfect matching if and only if for every subset S of X, $|Adj(S)| \geq |S|$, where $Adj(S)$ denotes the set of vertices adjacent to some vertex in S.

Let C be the set of columns of the constraint matrix and R the rows. $|Adj(S)|$ is the number of rows eligible for the set of columns $S \subseteq C$. Hall's theorem says that an assignment with a lower-Monge matrix has a feasible solution if and only if for every subset *S* of columns, the number of rows eligible for this subset is at least the cardinality of *S*.

Then the NWC rule orders the columns in an order $i_0,...,i_m$ and the sets $S_\alpha = \bigcup$ (this has no relationship to S_i in (2.24)). Because the problem is assumed to have a feasible solution, $|Adj(S_\alpha)| \geq |S_\alpha|$ for $\alpha = 0,...,m$. This means that i_a can be assigned to any of the $|S_\alpha|$ rows and the condition still holds on the remaining sets so the procedure can be iterated and the result is a feasible solution.

Q.E.D.

Proposition 2.3: If a feasible solution exists, the Northwest Corner Method finds an optimal solution for TPUS with lower-Monge matrices.

Proof:

This proof will be constructed by way of contradiction. Suppose then that no solution obtained by the Northwest Corner Method is optimal. Let $\hat{x}$ be an integer optimal solution to this problem. Then $\hat{x}$ is not obtained by the Northwest Corner Method and has a lower objective function value than any Northwest Corner Method solution. Let *i* be the first column such that the row chosen by $\hat{x}$, *b*, is not the row that the Northwest Corner Method says to choose, *a* (i.e. $\hat{x}_{i,b} = 1$ and $\hat{x}_{i,a} = 0$). Since the Northwest Corner Method says that (*i*, *a*) should be chosen, this implies that *a* < *b*. Since this is an equality constrained transportation problem, there must be another column, *j*, which supplies this row *a* in the solution $\hat{x}$ (i.e. $\hat{x}_{j,a} = 1$). Then we have that (*i*, *b*) and (*j*, *a*) are both chosen by $\hat{x}$. We have already shown why the entry (i, a) is defined. Because we assume the matrix has the lower-Monge property, if an element is defined in a row of a matrix, then every element after this column is defined. If $j < i$, then $\hat{x}$ is the same as the NWC solution, so by assumption $i < j$. Then since we assumed the matrix is lower-Monge, (j, b) is defined. Let x be the solution which agrees with $\hat{x}$ everywhere except in columns i and j. In these respective columns, instead choose cells (i, a) and (j, b). Then we have that the chosen arcs of $\hat{x}$ have a sum of $c_{i,b} + c_{j,a}$. By the lower-Monge property, $c_{i,a} + c_{j,b} \le c_{i,b} + c_{j,a}$. This says that x has a lower objective function value than $\hat{x}$, which contradicts that $\hat{x}$ is an optimal solution.

Q.E.D.

We can also define the weak transportation problem with unit supply (WTPUS) as follows:

WTPUS: $\min f(x) = \sum_{a \in Rows} \sum_{i \in Columns} c_{a,i} x_{a,i}$

subject to

$$\sum_{i \in Columns} x_{a,i} = 1 \text{ for each row } a \qquad (2.26)$$

$$\sum_{a \in Rows} x_{a,i} \le D_i \text{ for each column } i \qquad (2.27)$$

$$x_{a,i} \in \{0,1\}$$

Proposition 2.3a: If a feasible solution exists, the Northwest Corner Method finds an optimal solution for WTPUS with lower-Monge matrices and non-decreasing objective function values.

Proof:

This proof will be constructed by way of contradiction. Suppose then that no solution obtained by the Northwest Corner Method is optimal. Let $\hat{x}$ be an integer optimal solution to this problem. Then $\hat{x}$ is not obtained by the Northwest Corner Method and has a lower objective function value than any Northwest Corner Method solution. Let *i* be the first column such that the row chosen by $\hat{x}$ is not the row that the Northwest Corner Method says to choose.

Case 1: A row is chosen by NWC in column *i.* Then Proposition 3 applies to this case.

Case 2: The Northwest Corner Method does not select a row in this column. Let *a* be the row chosen by $\hat{x}$ in column *i* and let *j* be the column chosen by NWC in row *a*. If *j* is unused in $\hat{x}$ then consider the solution x which agrees with $\hat{x}$ everywhere except in row *a*, where instead the entry (*j*, *a*) is chosen. Because of the assumption that the cost matrix is non-decreasing (from left to right) and the fact that the NWC method will always choose a column to the left of the column *i*, this implies that $j < i$, and consequently $c_{j,a} \le c_{i,a}$. This says that x has a lower objective function value than $\hat{x}$, which contradicts that $\hat{x}$ is an optimal solution.

If *j* is used in $\hat{x}$, then because the NWC says to choose (*j*, *a*) we must have that $j < i$. We assumed that *i* was the first column such that the row chosen by $\hat{x}$ is not the row NWC says to choose. So this case contradicts our assumption.

Q.E.D.

Proposition 2.4: The bi-adjacency matrix for ESOP is a lower-Monge matrix when the flights are ordered by decreasing length and the slots by increasing time.

Proof:

Property (1) for lower-Monge matrices holds since for each flight, there is a scheduled arrival time, *arr*(*k*), and the flight can be scheduled to any slot *i* such that *time*(*i*) ≥ *arr*(*k*), and the flight *k* cannot be scheduled to any slot *i* with *time*(*i*) < *arr*(*i*). It needs to be shown that for all cells such that (*i*, *a*), (*i*, *b*), (*j*, *a*), and (*j*, *b*) are real values, with *i* < *j* and *a* < *b*, $c_{i,a} + c_{j,b} \le c_{i,b} + c_{j,a}$. Since the rows correspond to flights of (possibly) different lengths, suppose that *i* corresponds to a long flight, which will referred to as *LONG* and *j* corresponds to a short flight, which will be referred to as *SHORT*. Since the columns correspond to time slots, suppose that *a* corresponds to an early slot, which will be referred to as *EARLY* and *b* corresponds to a late slot, which will be referred to as *LATE*. To prove that the inequality holds, it can be shown to hold in each stage two scenario *t*. It is clear that, if *departure*(*k*, *i*) represents the controlled departure time of the flight *k* when assigned to slot *i*, then:

- *departure*(*LONG*, *EARLY*) ≤ *departure*(*SHORT*, *EARLY*)
- *departure*(*LONG*, *EARLY*) ≤ *departure*(*LONG*, *LATE*)
- *departure*(*LONG*, *EARLY*) ≤ *departure*(*SHORT*, *LATE*)
- *departure*(*LONG*, *LATE*) ≤ *departure*(*SHORT*, *LATE*)
- *departure*(*SHORT*, *EARLY*) ≤ *departure*(*SHORT*, *LATE*)

Case 1: $\tau(t)$ is before all controlled departure times.

Then because neither (*LONG*, *EARLY*) nor (*LONG*, *LATE*) has departed, they can both depart at the same time and be reallocated to the same set of slots in scenario *t*. This means that they will be reassigned to the same slot and thus be in the same comp. Likewise (*SHORT*, *EARLY*) and (*SHORT*, *LATE*) will be in the same comp. Hence

$$\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} = \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}.$$

Case 2: $\tau(t)$ is before the controlled departure times of only (*SHORT*, *EARLY*), (*LONG*, *LATE*), and (*SHORT*, *LATE*).

Then (*LONG*, *EARLY*) can be reassigned to *EARLY*, which is a slot to which (*LONG*, *LATE*) cannot be reassigned, so it is in an equal or earlier comp than (*LONG*, *LATE*). Once again (*SHORT*, *EARLY*) and (*SHORT*, *LATE*) will be in the same comp. Hence

$$\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} \le \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}.$$

Case 3: $\tau(t)$ is before the controlled departure times of only (*LONG*, *LATE*) and (*SHORT*, *LATE*).

Then (*LONG*, *EARLY*) and (*SHORT*, *EARLY*) have both departed and can be reassigned to the same slot, *EARLY*. Similarly neither (*LONG*, *LATE*) and (*SHORT*, *LATE*) have departed and can thus depart at time $\tau(t)$. However since *SHORT* is a shorter flight than *LONG*, (*SHORT*, *LATE*) can be reassigned to an earlier slot than (*LONG*, *LATE*), and thus may be in an earlier comp. Hence

$$\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} \le \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}.$$

Case 4: $\tau(t)$ is before the controlled departure times of only (*SHORT*, *EARLY*) and (*SHORT*, *LATE*).

Then (*LONG*, *EARLY*) can be reassigned to *EARLY* and (*LONG*, *LATE*) can be reassigned to *LATE*. Since *EARLY* is an earlier slot than *LATE*, (*LONG*, *EARLY*) may be in an earlier comp than (*LONG*, *LATE*). Since neither (*SHORT*, *EARLY*) nor (*SHORT*, *LATE*) have departed, they are eligible for the same slots and will be in the same comp. Hence $\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} \leq \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}$.

Case 5: $\tau(t)$ is before the controlled departure time of only (*SHORT*, *LATE*)

Then (*LONG*, *EARLY*) can be reassigned to *EARLY* and (*LONG*, *LATE*) can be reassigned to *LATE*. Since *EARLY* is an earlier slot than *LATE*, (*LONG*, *EARLY*) may be in an earlier comp than (*LONG*, *LATE*). Similarly, (*SHORT*, *EARLY*) will have departed be eligible for the slot *EARLY*, whereas (*SHORT*, *LATE*) will not. So (*SHORT*, *EARLY*) will be in an equal or earlier comp than (*SHORT*, *LATE*). Hence

$$\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} \leq \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}$$

Case 6: $\tau(t)$ is before all controlled departure times.

Then both (*LONG*, *EARLY*) and (*SHORT*, *EARLY*) will be in the same comp. Likewise (*LONG*, *LATE*) and (*SHORT*, *LATE*) will be in the same comp. Hence

$$\lambda_{LONG,EARLY,t} + \lambda_{SHORT,LATE,t} = \lambda_{LONG,LATE,t} + \lambda_{SHORT,EARLY,t}$$

Because the inequality holds in each case, the result follows from summing over all scenarios to arrive at the desired inequality. This shows that the expected stage one problem has a constraint matrix that is lower-Monge.

Q.E.D.

$$\begin{bmatrix} - & 1 & 4 \\ 0 & 0 & 3 \\ 1 & 1 & 1 \end{bmatrix}$$

Table 2.8: An example of a cost matrix for an ESOP problem.

Corollary 2.5: The Northwest Corner Method provides an optimal solution to the expected stage one problem, and this optimal solution is the RBD Solution.

The RBD Algorithm iteratively selects the longest remaining unscheduled flight for each slot (in increasing order of slot times). ESOP can be formulated as a transportation problem with equality constraints by noting a result from (Vossen and Ball, 2006), which states that the same set of slots will be used by any optimal solution to the Odoni model of the GDP. ESOP can then be re-stated with capacities to match these values. Similarly, ESOP has a non-decreasing objective function as the columns go from left to right. This means that Proposition 3 or Proposition 3a imply that the Northwest Corner Method will give the optimal solution to ESOP. When the Northwest Corner Method is then run on the ESOP, it will give the same arcs as the RBD algorithm. Thus this corollary shows that the RBD arcs will solve the expected stage one problem. This expected stage one assignment can be viewed as an updated primal, where the stage one costs of an arc (k, i) is the expected comp delay, $c_{k,\,i}$. An optimal solution to this problem will give a set of stage one arcs that use the earliest comps in each scenario in stage two. By LP Duality, there exists a dual feasible solution (u, v) to this problem with equal objective function value as the primal. This stage one dual can be combined with the stage two dual already obtained (λ, δ, y) to get a dual solution (u, v, λ, δ, y) to the overall problem.

Lemma 2.6: The solution (u, v, λ, δ, y) is dual feasible to the overall problem.

Proof:

Chapter 2.3.1 shows that (λ, δ, y) is a feasible dual solution. The fact that (u, v) is the optimal dual solution to the expected stage one problem, which contains the constraints $u_k + v_i \le c_{k,i} = \sum_{t \in Scenarios} \lambda_{k,i,t}$ and $u_s + v_i \le 0$, shows that the solution (u, v, λ, δ, y) is a dual feasible solution to the overall problem.

Q.E.D.

Theorem 2.7: The RBD-Inspired solution is optimal to the overall problem.

Proof:

This proof will be constructed by showing that the dual feasible solution (u, v, λ, δ, y) has an equal objective function value as the RBD-Inspired primal solution. The dual objective function is $\sum_{(k,i) \in NWC} c_{k,i} + \sum_{t \in Scenarios} \sum_{j \in used(t)} cap_2(j,t)\delta_{j,t}$

$= \sum_{(k,i) \in RBD} c_{k,i} + \sum_{t \in Scenarios} \sum_{j \in used(t)} cap_2(j,t)\delta_{j,t}$ because the NWC arcs are the same as the RBD arcs.

$$= \sum_{(k,i)\in RBD} \sum_{t\in Scenarios} \lambda_{k,i,t} + \sum_{t\in Scenarios} \sum_{j\in used(t)} cap_2(j,t)\delta_{j,t}$$ by the definition of $c_{k,i}$

$$= \sum_{(k,i)\in RBD} \sum_{t\in Scenarios} p_t(time(j_\beta,t) - arr(k)) + \sum_{t\in Scenarios} \sum_{j\in used(t)} cap_2(j,t)\, p_t(time(j,t) - time(j_\beta,t))$$ by the definition of $\lambda_{k,i,t}$ and $\partial_{j,t}$.

$$= \sum_{(k,i)\in RBD} \sum_{t\in Scenarios} \left(p_t(time(j_\beta,t))\right) + \sum_{(k,i)\in RBD} \sum_{t\in Scenarios} \left(p_t(-arr(k))\right) + \sum_{t\in Scenarios} \sum_{j\in used(t)} \left(cap_2(j,t)\, p_t(time(j,t))\right) + \sum_{t\in Scenarios} \sum_{j\in used(t)} \left(cap_2(j,t)\, p_t\left(time\left(-j_\beta,t\right)\right)\right)$$

$$= \sum_{(k,i)\in RBD} \sum_{t\in Scenarios} p_t(-arr(k)) + \sum_{t\in Scenarios} \sum_{j\in used(t)} cap_2(j,t)\, p_t(time(j,t))$$, because $|used(t)|$ = |RBD| and each used stage two slot is used by an RBD arc in each scenario *t.*

$$= \sum_{t\in Scenarios} \left(\sum_{k\in Flights} p_t\left(-arr(k)\right) + \sum_{j\in used(t)} cap_2(j,t)\, p_t(time(j,t)) \right)$$

$$= \sum_{k\in Flights} \sum_{t\in Scenarios} \sum_{j\in used(t)} p_t(cap_2(j,t) time(j,t) - arr(k))$$, because each flight k uses a single slot $j \in used(t)$.

Since this last line is equal to the objective function value for the RBD-Inspired solution, the Weak Duality Theorem implies that both the primal and dual solutions are optimal.

Q.E.D.

This shows that the RBD-Inspired solution is an integer optimal solution to the LP-relaxation of this integer program. Hence the LP-relaxation solves the IP to optimality.

2.3 Other Formulations

The previous section shows that the Ground Delay Problem with Weather Uncertainty can be solved by an LP. The speed at which these LPs can be solved is determined in large part by the size of the formulation. Here these results are extended by providing new formulations, which are equal in strength to this formulation, but smaller in size. The reduction in size will result in the availability to solve larger instances of these and similar problems.

2.3.1 Flight-Based Formulation

In the initial formulation, which now will be referred to as the *Allocation-Based Formulation*, careful attention was paid to the slots where flights were initially allocated. This was useful to help determine which initial allocations could be reallocated to certain slots and which ones could not. The problem with this formulation is that the stage two variables require four subscripts, which causes the problem sizes to grow very large very fast. In the next formulation, it will be shown that equivalent or nearly equivalent formulations can be obtained with fewer variable subscripts and much smaller formulation sizes.

What is necessary, though, is a set of constraints that ensures that no arc in stage one is reallocated to a slot in a scenario of stage two that it cannot actually supply. In the Allocation-Based Formulation, this was done through the definition of the set $Feas(k, i, t)$. The new formulation will eliminate the subscript i, so the corresponding set cannot be defined. Instead, the question of whether an arc (k, i)

from stage one can supply the slot j in scenario t will be checked by the function $last(k, j, t)$. This records the latest slot, i, in stage one that the flight k can be initially allocated to and still be reallocated to the slot j in scenario t of stage two. This function will serve the desired purpose because if $i = last(k, j, t)$ then for $i' < i$, the arc (k, i') of stage one can also be reallocated to the stage two slot j in scenario t. The determination of this function is a simple measure of pre-processing based on two cases:

- If the slot j requires the flight k to have already departed in scenario t, i.e. $time(j,t) - len(k) < \tau(t)$, then $last(k, j, t)$ is the latest stage one slot i such that $arr(k) \le time(i) \le time(j,t)$
- If the slot j does not require the flight k to have departed in scenario t, then the flight k can be initially assigned to any stage one slot i and still be reallocated to the slot j in scenario t. Consequentially, $last(k, j, t)$ is the last slot of stage one.

Flight-Based Formulation

$$\min f(x,y) = \sum_{k \in Flights} \sum_{j \in Slots_2} \sum_{t \in Scenarios} \left(p_t \left(time(j,t) - arr(k) \right) y_{k,j,t} \right) \tag{2.28}$$

subject to

$$\sum_{\substack{i \in Slots_1 | \\ time(i) \ge arr(k)}} x_{k,i} = 1 \text{ for each flight } k \tag{2.29}$$

$$\sum_{\substack{k \in Flights | \\ arr(k) \le time(i)}} x_{k,i} + x_{s,i} = cap_1(i) \text{ for each arrival slot } i \tag{2.30}$$

$$\sum_{i \in Slots_1} x_{s,i} = \sum_{i \in Slots_1} cap_1(i) - |Flights| \tag{2.31}$$

$$\sum_{\substack{j \in Slots2| \\ time(j,t) \geq arr(k)}} y_{k,j,t} = 1 \text{ for each } k,\ t \tag{2.32}$$

$$\sum_{\substack{k \in Flights| \\ arr(k) \leq time(j,t)}} y_{k,j,t} + y_{s,j,t} = cap_2(j,t) \text{ for all } j,\ t \tag{2.33}$$

$$\sum_{j \in Slots_2} y_{s_t,j,t} = \sum_{j \in Slots_2} cap_2(j,t) - |Flights| \text{ for all } t \tag{2.34}$$

$$\sum_{\substack{i \in Slots_1| \\ arr(k) \leq time(i) \\ time(i) \leq last(k,j,t)}} x_{k,i} - \sum_{\substack{j' \in Slots_2| \\ time(j',t) \geq arr(k) \\ j' \leq j}} y_{k,j',t} \geq 0 \tag{2.35}$$

for each flight k, stage two slot j and scenario t

$x \in \{0,1\}$

$y \geq 0, y \in \mathbb{Z}$

Constraints sets (2.29), (2.30), and (2.31), the stage one constraints in this formulation are the same as in the Allocation-Based Formulation. The variables $y_{k,j,t}$ represent the reassignment of flight k to slot j in scenario t (These variables have no relationship to the variables $y_{k,i,t}$ in Chapter 2.3.1). Constraint set (2.32) requires that each flight be reallocated to a slot in every scenario. This should be compared to constraint set (2.4) in the Allocation-Based Formulation, which says that each arc from stage one must be reallocated to some slot in stage two. Constraint set (2.33) requires that each slot be used in each scenario (either by an actual flight or by the slack flight). This constraint set is similar to constraint set (2.5) in the Allocation-Based Formulation. Constraint set (2.34) requires that the slack flight must supply $\sum_{j \in Slots_2} cap_2(j,t)$ − |*Flights*| slots, where $\sum_{j \in Slots_2} cap_2(j,t)$ − |*Flights*| is the number of stage two slots that will go unused. This constraint set is similar to constraint set (2.6) in the Allocation-Based Formulation. Constraint (2.35) says that for each flight and final slot in each scenario, there needs to be enough initial allocations (supply) to support these final allocations (demand). These constraints, which link the variables in stage one with those in stage two, ensure that the solution returned by the LP-relaxation is feasible by guaranteeing that no subset of the slots is ever demanding more flights to be allocated to them than can possibly be allocated by the stage one variables.

Notice the difference between the stage two variables in the Allocation-Based Formulation and the Flight-Based Formulation. In the Allocation-Based Formulation, there were four subscripts corresponding to: the flight, the initial allocation, the final allocation, and the scenario. In this new formulation, there are now three subscripts, with the subscript representing the initial allocation now removed.

This next result compares the strength of the Flight-Based Formulation to that of the Allocation-Based Formulation. In order for such a comparison to be made though, a model needs to be constructed which consists of variables from both formulations. This can be constructed formulating an extension of the LP-relaxation of the Allocation-Based Formulation to the variables of the Flight-Based Formulation. The strength of the formulations can be done by showing that the Flight-Based Formulation is a projection of such a formulation.

Theorem 2.8: The LP-Relaxation of the Flight-Based Formulation is a projection of an extension of the LP-Relaxation of the Allocation-Based Formulation.

Proof:

To prove this, define the following set W_Y = {(x, w, y) : (x, w) is feasible to the LP-relaxation of the Allocation-Based Formulation and $y_{k,j,t} = \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k)}} w_{k,i,j,t}$ }. The projection of W_Y onto the subspace of (x, y) variables is Y = {(x, y) | there exists a w such that (x, w, y) ∈ W_Y }. Let Y' be the set of feasible solutions to the Flight-Based formulation. Our claim is that $Y = Y'$. This can be shown using basic set theory on the equality of sets.

Suppose first that (x, y) ∈ Y. Then $y_{k,j,t} = \sum_{i \in Slots_1 | time(i) \geq arr(k)} w_{k,i,j,t}$. Because constraint sets (2.29), (2.30), and (2.31) are repeats of constraint sets (2.1), (2.2), and (2.3) so they clearly hold.

$$
\begin{aligned}
&\sum_{j \in Slots_2 | time(j,t) \geq arr(k)} y_{k,j,t} = \sum_{j \in Slots_2 | time(j,t) \geq arr(k)} \sum_{i \in Slots_1 | time(i) \geq arr(k)} w_{k,i,j,t} \\
&= \sum_{i \in Slots_1 | time(i) \geq arr(k)} \sum_{j \in Slots_2 | time(j,t) \geq arr(k)} w_{k,i,j,t} \\
&= \sum_{i \in Slots_1 | time(i) \geq arr(k)} \left(1 - w_{k,i,p_t,t}\right) \\
&= \sum_{i \in Slots_1 | time(i) \geq arr(k)} x_{k,i} \\
&= 1
\end{aligned}
$$

so constraint (2.32) holds.

$$
\sum_{k \in Flights | time(j,t) \geq arr(k)} y_{k,j,t} + y_{s_t,j,t} = \sum_{k \in Flights | time(j,t) \geq arr(k)} \sum_{i \in Slots_1 | time(i) \geq arr(k)} w_{k,i,j,t} + w_{s_t,j,t} = 1
$$

so constraint (2.33) holds.

$$\sum_{j \in Slots_2} y_{s_t,j,t} = \sum_{j \in Slots_2} w_{s_t,j,t} = \sum_{j \in Slots_2} cap_2(j,t) - |Flights|$$

so constraint (2.34) holds.

$$\sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} x_{k,i} - \sum_{\substack{j' \in Slots_2 | \\ j' \leq j}} y_{k,j',t} = \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} x_{k,i} - \sum_{\substack{j' \in Slots_2 | \\ j' \leq j}} \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k)}} w_{k,i,j',t}$$

$$= \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} x_{k,i} - \sum_{\substack{j' \in Slots_2 | \\ j' \leq j}} \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j',t)}} w_{k,i,j',t}$$

$$\geq \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} x_{k,i} - \sum_{\substack{j' \in Slots_2 | \\ j' \leq j}} \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} w_{k,i,j,t}$$

$$= \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k) \\ time(i) \leq last(k,j,t)}} \left(x_{k,i} - \sum_{\substack{j' \in Slots_2 | \\ j' \leq j}} w_{k,i,j,t} \right)$$

$$\geq 0$$

so constraint (2.35) holds.

This proves that $Y \subseteq Y'$.

To see that $Y' \subseteq Y$, we need to show that for every $(x, y) \in Y'$, there exists a w such that $(x, w, y) \in W_Y$. To do this, suppose that (x, y) is feasible to the Flight-Based Formulation. For a given flight, k, and scenario, t, the amount $\sum_{i | time(i) \geq arr(k)} x_{k,i}$ can be regarded as the amount of supply to be distributed amongst the stage two slots. Similarly, the amount $\sum_{j | time(j,t) \geq arr(k)} y_{k,j,t}$ can be viewed as the amount of demand going to the stage two slots. Constraints (2.29) and (2.32) imply that $\sum_{i | time(i) \geq arr(k)} x_{k,i} = \sum_{j | time(j,t) \geq arr(k)} y_{k,j,t} = 1$, so supply is equal to demand. Then the attempt to formulate a solution (x, w) which is feasible to the Allocation-Based Formulation is equivalent to trying to find a feasible means of meeting the demands of the $y_{k,j,t}$ variables with the given supply of $x_{k,i}$ variables, with the additional constraint that an $x_{k,i}$ variable is not allowed to supply a $y_{k,j,t}$ variable that it cannot be reallocated to. This can be formulated as the following LP:

$$\min \sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{j \in Slots_2} \sum_{t \in Scenarios} 0 \cdot w_{k,i,j,t} \tag{2.36}$$

subject to

$$\sum_{i | time(i) \geq arr(k)} w_{k,i,j,t} = y_{k,j,t} \tag{2.37}$$

for each flight k, stage two slot j, and scenario t

$$\sum_{j \in Slots_2} w_{k,i,j,t} = x_{k,i} \tag{2.38}$$

for each flight k, stage one slot i, and scenario t

$$w_{k,i,j,t} = 0 \text{ if } j \in Feas(k,i,t) \tag{2.39}$$

This formulation is a transportation problem with a set of side constraints (more specifically certain variables restricted to be zero). Notice though, that the side constraints will be implicitly verified by every solution of the Flight-Based Formulation since constraint set (2.35) implies that the slot j in scenario t can only be supplied by those $x_{k,i}$ that can reach it. This is true if and only if $j \in Feas(k, i, t)$. This implies constraint set (2.39) so these constraints can be omitted from the above formulation.

Let k and t be given and suppose that $i_1,...,i_n$ are the slots receiving flow from k in stage one and suppose that $j_1,..., j_m$ are the slots receiving flow from k in stage two. Then the m by n transportation matrix for flight k in scenario t can be set up as stated above. Suppose that C is a subset of the columns (stage one slots) of this problem. Then a lower bound on the number of rows C can be matched to is given by $|C|$, since each stage one slot can be reallocated to itself in each scenario of stage-two. Hall's theorem provides this as a sufficient condition for the existence of a solution to the transportation problem. The solution (x, w, y) would then belong to W and we have that $Y' \subseteq Y$.

Q.E.D

2.3.2 Queue-Based Formulation

The Flight-Based Formulation makes use of pre-processing to reduce the number of subscripts necessary in stage two, while not losing any of the strength of the original formulation. A natural question then arises of whether this pre-processing can be used to obtain another formulation that is equally as strong, but still smaller in size.

To accomplish this, though, some problem modeling is required that uses specific properties of the application context. For instance, the cost of ground delaying a flight for one extra slot in the stage two scenario t is $p_t\left(time\left(j+1,t\right)-time\left(j,t\right)\right)$. This cost structure leads to a situation where, the objective function value does not depend on specific flight-to-slot assignments, but rather only on the set of slots being used. In light of this, it is possible to take advantage of a formulation where the subscript indicating flight is also omitted from the stage two variables. This leads to a formulation where there are only two subscripts on the stage two variables, indicating which slots are used in each scenario.

Instead of formulating stage two as an assignment problem in each scenario as the Allocation-Based Formulation and Flight-Based Formulation do, this formulation will set up a queue of the stage two slots, in each scenario t. Similar to the Allocation-Based Formulation and the Flight-Based Formulation, there must be a way of ensuring that no arc from stage one is reassigned to a slot in a scenario of stage two that it cannot reach. This is accomplished by the function $earliest(k,i,t)$. It determines at which slot the stage one arc (k, i) will enter the scenario t queue. A flight k that was assigned to slot i in stage one enters the queue at slot j if $earliest(k,i,t) = j$ and $x_{k,i} = 1$. Flights can then exit the queue which is done by increasing the value of $u_{j,t}$ by 1, or remain in the queue, which means being passed from slot j to slot j+1 in scenario t, which is done by increasing the value of $z_{j,t}$ by 1.

The function $earliest(k,i,t)$ can be computed in preprocessing by simply noting if the flight k that was initially assigned to slot i has already departed, if it is currently serving ground delay, or if its departure time has not yet arrived. Each of these conditions is a simple function of the prescheduled arrival time of the flight, the length of the flight, and the cancellation time $\tau(t)$.

- If the flight is currently airborne (i.e. $time(i) - len(k) \leq \tau(t)$) then the earliest airport arrival slot that (k, i) can be rescheduled to is the slot for which it had already departed, $time(i)$.
- If the flight is serving ground delay at the time of the GDP cancellation (i.e. $\tau(t) + len(k) \geq arr(k)$ and $time(i) - len(k) > \tau(t)$) then since the GDP has been cancelled, this flight may no longer need to serve ground delay subject to the reduced capacities at the airport. The earliest airport arrival slot that (k, i) can be rescheduled to is then the slot $\tau(t) + len(k)$, indicating that the flight can depart immediately.

- If the flight is currently grounded by not serving ground delay, then the prescheduled arrival time for this flight had not yet occurred (i.e. $\tau(t)$ + *len(k)* < *arr(k)*). These flights cannot depart until their prescheduled departure time occurs, in which case the earliest airport arrival slot they can be rescheduled to is *arr*(*k*).

The notion of an earliest reallocation for each initial allocation and each scenario does not necessarily mean that the allocation will be reallocated to this arrival slot, in the event that the allocation is used in stage one. The nominal capacity limitations must still be respected. In the event that there are more flights attempting to use an airport arrival slot than that slot's nominal capacity, some of the flights will wait on the ground for a later slot. This process continues until all flights have been reallocated to a slot.

Queue-Based Formulation

$$\min f(x,u,v,z) = \sum_{k\in Flights} \sum_{j\in Slots_2} \sum_{t\in Scenarios} \left(p_t \left(time(j,t) - arr(k) \right) u_{j,t} \right)$$

subject to

$$\sum_{\substack{i\in Slots_1 | \\ time(i)\geq arr(k)}} x_{k,i} = 1 \text{ for each flight } k \tag{2.40}$$

$$\sum_{\substack{k\in Flights | \\ arr(k)\leq time(i)}} x_{k,i} + x_{s,i} = cap_1(i) \text{ for each arrival slot } i \tag{2.41}$$

$$\sum_{i\in Slots_1} x_{s,i} = \sum_{i\in Slots_1} cap_1(i) - |\, Flights \,| \tag{2.42}$$

$$\sum_{(k,i) | earliest(k,i,t)=j} x_{k,i} + z_{j-1,t} - z_{j,t} - u_{j,t} = 0 \text{ for each slot } j \text{ and scenario } t \tag{2.43}$$

$$u_{j,t} + v_{j,t} = cap_2(t) \text{ for each slot } j \text{ and scenario } t \tag{2.44}$$

$$\sum_{j \in Slots_2} v_{j,t} = \sum_{j \in Slots_2} cap_2(j,t) - |Flights| \text{ for each scenario } t \tag{2.45}$$

$$x \in \{0,1\}$$
$$u, v \geq 0, u, v \in \mathbf{Z}$$

$z_{j,t}$ is the number of flights passed from slot j to slot j+1 in scenario t.

$u_{j,t}$ is the number of flights that are rescheduled to slot j in scenario t

$v_{j,t}$ is nonzero if no flight is rescheduled to slot j in scenario t

Constraint sets (2.40), (2.41), and (2.42) are equal to constraint sets (2.1), (2.2), and (2.3) of the Allocation-Based Formulation and constraint sets (2.29), (2.30), and (2.31) of the Flight-Based Formulation.

Constraint set (2.44) is similar to constraint set (2.5) of the Allocation-Based Formulation and constraint set (2.33) of the Flight-Based Formulation. It says that each slot must be used by a combination of real flights (i.e. $u_{j,t}$) and the slack flight (i.e. $v_{j,t}$). Constraint set (2.45) is similar to constraint set (2.6) of the Allocation-Based Formulation and constraint set (2.34) of the Flight-Based Formulation. It says that the number of unused slots must be equal to the total number of slots minus the total number of flights.

Again, notice the difference between the stage two variables in the Allocation-Based Formulation, the Flight-Based Formulation and the Queue-Based formulation. In the Allocation-Based Formulation, there were four subscripts indicating: the flight, the initial allocation, the final allocation, and the scenario. In the Flight-Based

Formulation, there were three subscripts indicating: the flight, the final allocation, and the scenario. In this new formulation, there are now two subscripts, with the subscripts representing the flight and initial allocation now removed.

Similar to the Flight-Based Formulation, the strength of this new formulation needs to be checked in comparison to the Allocation-Based Formulation. To do this, an extension of the LP-relaxation of the Allocation-Based Formulation is constructed where variables (u, v, z) representing the Queue-Based Formulation are defined based on the Allocation-Based Formulation.

Theorem 2.9: The LP-Relaxation of the Queue-Based Formulation is a projection of an extension of the LP-Relaxation of the Allocation-Based Formulation.

Proof:

To prove this, define the following set W_Q = {(x, w, u, v, z) : (x, w) is feasible to the LP-relaxation of the Allocation-Based Formulation, $u_{j,t} = \sum_{k \in Flights} \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k)}} w_{k,i,j,t}$ and $v_{j,t} = w_{s_t,j,t}$ for each stage two slot j and scenario t. Similarly, $z_{1,t} = \sum_{(k,i)|earliest(k,i,t)=time(1,t)} -u_{1,t}$ for all scenarios t and $z_{j,t} = \sum_{(k,i)|earliest(k,i,t)=j} z(j-1,t) - u_{j,t}$ for each stage two slot $j > 1$ and scenario t}. The projection of W onto the subspace of (x, u, v) variables is Q = {(x, u, v, z) | there exists a w such that (x, w, u, v, z) ∈ W_Q }. Let Q' be the set of feasible solutions to the Queue-Based Formulation. Our claim is that $Q = Q'$. This can be shown using basic set theory on the equality of sets.

Suppose first that (*x*, *u*, *v*, *z*) ∈ *Q*. Then $u_{j,t} = \sum_{k \in Flights} \sum_{i \in Slots_1 | time(i) \geq arr(k)} w_{k,i,j,t}$.

Because constraint sets (2.40), (2.41), and (2.42) are repeats of constraint sets (2.1), (2.2), and (2.3) so they clearly hold.

$$u_{j,t} + v_{j,t} = \sum_{k \in Flights} \sum_{\substack{i \in Slots_1 | \\ time(i) \geq arr(k)}} w_{k,i,j,t} + w_{s_t,j,t} = 1$$

so constraint (2.44) holds.

$$\sum_{j \in Slots_2} v_{j,t} = \sum_{j \in Slots_2} w_{s_t,j,t} = \sum_{j \in Slots_2} cap_2(j,t) - |Flights|$$

so constraint (2.45) holds.

$$\begin{aligned}
&\sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - z_{1,t} - u_{1,t} \\
&= \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - \left(\sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - u_{1,t} \right) - u_{1,t} \\
&= 0 \\
&\sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - z_{1,t} + z_{j-1,t} - u_{1,t} \\
&= \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - \left(\sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} + z_{j-1,t} - u_{1,t} \right) + z_{j-1,t} - u_{1,t} \\
&= 0
\end{aligned}$$

So constraint (2.43) holds.

$$z(j,t) = \begin{cases} \sum_{(k,i)|earliest(k,i,t)=j} -u_{j,t} & \text{if } j = 1 \\ \sum_{(k,i)|earliest(k,i,t)=j} +z(j-1,t) - u_{j,t} & \text{if } j \neq 1 \end{cases}$$

so there are two cases to check if z > 0.

Case 1: $j = 1$

Then

$$z(1,t) = \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - u_{1,t}$$

$$= \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - \sum_{k \in Flights} \sum_{i \in Slots_1} w_{k,i,1,t}$$

$$= \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - \sum_{\substack{k \in Flights \\ i \in Slots_1 \\ 1 \in Feas(k,i,t)}} w_{k,i,1,t}$$

$$= \sum_{(k,i)|earliest(k,i,t)=1} x_{k,i} - \sum_{\substack{k \in Flights \\ i \in Slots_1 \\ eariest(k,i,t)=1}} w_{k,i,1,t}$$

$$= \sum_{(k,i)|earliest(k,i,t)=1} \left(x_{k,i} - w_{k,i,1,t} \right) \geq 0$$

Case 2: $j > 1$

$$z_{j,t} = \sum_{(k,i)|earliest(k,i,t)=j} x_{k,i} + z_{j-1,t} - u_{j,t}$$

$$= \sum_{(k,i)|earliest(k,i,t)=j} x_{k,i} + z_{j-1,t} - u_{j,t}$$

$$= \sum_{\substack{j' \in Slots_2 \\ j' \leq j}} \sum_{(k,i)|earliest(k,i,t)=j'} x_{k,i} - \sum_{\substack{j' \in Slots_2 \\ j' \leq j}} \sum_{k \in Flights} \sum_{i \in Slots_1} w_{k,i,j',t}$$

$$= \sum_{(k,i)|earliest(k,i,t) \leq j} \left(x_{k,i} - w_{k,i,j,t} \right) \geq 0$$

This proves that $Q \subseteq Q'$.

To see that $Q' \subseteq Q$, we need to show that for every $(x, u, v, z) \in Q'$, there exists a w such that $(x, w, u, v, z) \in W_Q$. To do this, suppose that (x, u, v, z) is feasible to the Queue-Based Formulation. For a given scenario t, the amount $\sum_{k \in Flights} \sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} x_{k,i}$ can be regarded as the amount of supply to be distributed amongst the stage two slots. Similarly, the amount $\sum_{j \in Slots_2} u_{j,t}$ can be viewed as the amount of demand going to the stage two slots. Constraints (2.40) and (2.43) imply that

$\sum_{k \in Flights} \sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} x_{k,i} = \sum_{\substack{j \in Slots_2 \\ time(j,t) \geq arr(k)}} u_{j,t} = |Flights|$, so supply is equal to demand. Then the attempt for formulate a solution (x, w) which is feasible to the Allocation-Based Formulation is equivalent to trying to find a feasible means of meeting the demands of the $u_{j,t}$ variables with the given supply of $x_{k,i}$ variables, with the additional constraint that an $x_{k,i}$ variable is not allowed to supply a $u_{j,t}$ variable that it cannot be reallocated to. This can be formulated as the following LP:

$$\min \sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{j \in Slots_2} \sum_{t \in Scenarios} 0 \cdot w_{k,i,j,t} \tag{2.46}$$

subject to

$$\sum_{k \in Flights} \sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} w_{k,i,j,t} = u_{j,t} \tag{2.47}$$

stage two slot j, and scenario t

$$\sum_{j \in Slots_2} w_{k,i,j,t} = x_{k,i} \tag{2.48}$$

for each flight k, stage one slot i, and scenario t

$$w_{k,i,j,t} = 0 \text{ if } j \in Feas(k,i,t) \tag{2.49}$$

This formulation is a transportation problem with a set of side constraints (more specifically certain variables restricted to be zero). Notice though, that the side constraints will be implicitly verified by every solution of the Queue-Based Formulation since constraint set (2.43) implies that the slot j in scenario t can only be supplied by those $x_{k,i}$ that can reach it. This is true if and only if $j \in Feas(k, i, t)$. This implies constraint set (2.49) so these constraints can be omitted from the above formulation.

Let t be given and suppose that $(k_1, i_1), \ldots, (k_n, i_n)$ are the arcs receiving flow in stage one and suppose that $j_1, \ldots, j_m$ are the slots receiving flow in stage two. Then the m by n transportation matrix for scenario t can be set up as stated above. Suppose that C is a subset of the columns (stage one arcs receiving flow) of this problem. Then each element of C can be mapped to the first unused slot after *earliest*(k, i, t), a lower bound on the number of rows C can be matched to is given by $|C|$. Hall's theorem provides this as a sufficient condition for the existence of a solution to the transportation problem. The solution (x, w, u, v, z) would then belong to W_Q and we have that $Q' \subseteq Q$.

Q.E.D.

In both Theorem 2.8 and Theorem 2.9, the sets W_Y and W_Q were presented with added variables to the Allocation-Based Formulation representing those in the Flight-Based Formulation and Queue-Based Formulation respectively. Because these variables do not change the feasibility of any solutions of the Allocation-Based Formulation, Theorem 2.7 will hold on these formulations as well. In particular, any solution given by the RBD Algorithm will be optimal in both W_Y and W_Q. Because The Flight-Based Formulation and Queue-Based Formulation were shown to projections of W_Y and W_Q respectively, this means that there exists a corresponding RBD-Inspired optimal solution in each of these formulations as well.

2.4 Polyhedral Results

Knowing now that the LP-relaxation of this IP solves the IP, questions arise as to how strong the LP-relaxation actually is. A large class of IPs that have this property are those IPs with totally unimodular (TU) constraint matrices. A matrix is TU if every square sub-matrix of A has determinant +1, -1, or 0. A natural follow-up question is whether or not the constraint matrix for this problem is TU. The next claim addresses that question.

Theorem 2.10: For any instance of the Allocation-Based Formulation with at least two flights k_1, and k_2 and two stage one slots i_1, and i_2, where both k_1 and k_2 can be assigned to both i_1 and i_2, the constraint matrix for this problem is not TU and, further, the polyhedron for this formulation can have non-integer extreme points.

Example 2.3

Consider the following example, with all variables between 0 and 1.

$$\min f(x,w) = -x_{k_1,i_1} - x_{k_1,i_2} - x_{k_2,i_1} - x_{k_2,i_2} - w_{k_1,i_1,p_t,t} - w_{k_1,i_2,p_t,t} - w_{k_2,i_1,p_t,t}$$
$$- w_{k_2,i_2,p_t,t} - w_{k_1,i_1,j_1,t} - w_{k_1,i_2,j_2,t} - w_{k_2,i_1,j_2,t} - w_{k_2,i_2,j_1,t} - w_{s_t,j_3,t}$$

subject to

$$x_{k_1,i_1} + x_{k_1,i_2} = 1$$
$$x_{k_2,i_1} + x_{k_2,i_2} = 1$$
$$x_{k_1,i_1} + x_{k_2,i_1} = 1$$
$$x_{k_1,i_2} + x_{k_2,i_2} = 1$$
$$w_{k_1,i_1,p_t,t} + w_{k_1,i_2,p_t,t} + w_{k_2,i_1,p_t,t} + w_{k_2,i_2,p_t,t} = 2$$
$$w_{k_1,i_1,p_t,t} + w_{k_1,i_1,j_1,t} + w_{k_1,i_1,j_2,t} + w_{k_1,i_1,j_3,t} = 1$$
$$w_{k_1,i_2,p_t,t} + w_{k_1,i_2,j_2,t} + w_{k_1,i_2,j_3,t} = 1$$
$$w_{k_2,i_1,p_t,t} + w_{k_2,i_1,j_1,t} + w_{k_2,i_1,j_2,t} + w_{k_2,i_1,j_3,t} = 1$$
$$w_{k_2,i_2,p_t,t} + w_{k_2,i_2,j_1,t} + w_{k_2,i_2,j_2,t} + w_{k_2,i_2,j_3,t} = 1$$
$$w_{k_1,i_1,j_1,t} + w_{k_2,i_1,j_1,t} + w_{k_2,i_2,j_1,t} + w_{s,j_1,t} = 1$$
$$w_{k_1,i_1,j_2,t} + w_{k_1,i_2,j_2,t} + w_{k_2,i_1,j_2,t} + w_{k_2,i_2,j_2,t} + w_{s,j_2,t} = 1$$
$$w_{k_1,i_1,j_3,t} + w_{k_1,i_2,j_3,t} + w_{k_2,i_1,j_3,t} + w_{k_2,i_2,j_3,t} + w_{s,j_3,t} = 1$$
$$w_{s,j_1,t} + w_{s,j_2,t} + w_{s,j_3,t} = 1$$
$$x_{k_1,i_1} + w_{k_1,i_1,p_t,t} = 1$$
$$x_{k_1,i_2} + w_{k_1,i_2,p_t,t} = 1$$
$$x_{k_2,i_1} + w_{k_2,i_1,p_t,t} = 1$$
$$x_{k_2,i_2} + w_{k_2,i_2,p_t,t} = 1$$

Stage one:

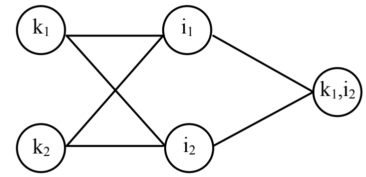

Figure 2.6: Stage one of Example 2.3

Stage two:

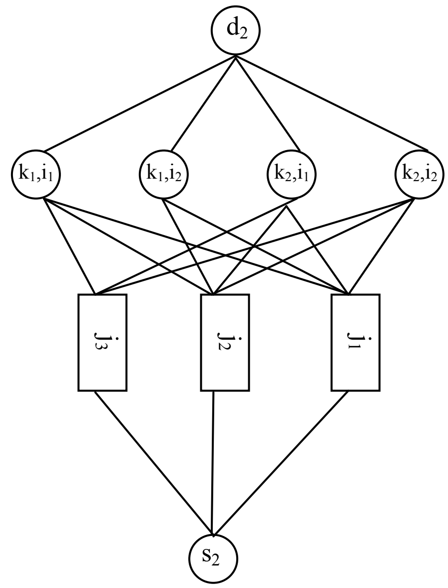

Figure 2.7: Stage two of Example 2.3

An optimal solution to this LP is

$$x_{k_1,i_1} = 0.5 \quad x_{k_1,i_2} = 0.5 \quad x_{k_2,i_1} = 0.5 \quad x_{k_2,i_2} = 0.5$$
$$w_{k_1,i_1,d_t,t} = 0.5 \quad w_{k_1,i_2,d_t,t} = 0.5 \quad w_{k_2,i_1,d_t,t} = 0.5 \quad w_{k_2,i_2,d_t,t} = 0.5$$
$$w_{k_1,i_1,j_1,t} = 0.5 \quad w_{k_1,i_2,j_2,t} = 0.5 \quad w_{k_2,i_1,j_2,t} = 0.5 \quad w_{k_2,i_2,j_1,t} = 0.5$$
$$w_{s_t,j_3,t} = 1.0$$

with all other variables equal to zero. A sufficient condition for a matrix A to be TU is if the LP, $\max\{cx : Ax \le b, x \in \square\}$, has an integer optimal solution for all integer vectors b. Because this example returns a non-integer solution with an integer vector b, it shows that the constraint matrix for this problem is not TU.

Q.E.D.

Not only does this example show that the constraint matrix is not TU, it answers a second line of questioning, showing that the structure of the corresponding polyhedron is not integer. This line of reasoning continues with the question of if there exists an inequality which cuts off the above non-integer extreme points. The next claim answers that question.

Theorem 2.11: The following inequality is valid for all integer solutions feasible to the Allocation-Based Formulation, but are not satisfied by the non-integer extreme points of Theorem 10

$$w_{k_1, i_1, p_t, t} + w_{k_2, i_2, p_t, t} + w_{k_1, i_1, j_1, t} + w_{k_2, i_2, j_1, t} + w_{k_1, i_2, p_t, t} + w_{k_2, i_1, p_t, t} + w_{k_1, i_2, j_2, t} + w_{k_2, i_1, j_2, t} \le 3$$

Proof:

Since our non-integer solution consists of :

$$w_{k_1, i_1, p_t, t} = 0.5$$
$$w_{k_1, i_2, p_t, t} = 0.5$$
$$w_{k_2, i_1, p_t, t} = 0.5$$
$$w_{k_2, i_2, p_t, t} = 0.5$$
$$w_{k_1, i_1, j_1, t} = 0.5$$
$$w_{k_1, i_2, j_2, t} = 0.5$$
$$w_{k_2, i_1, j_2, t} = 0.5$$
$$w_{k_2, i_2, j_1, t} = 0.5$$

The left hand side (LHS) of these constraints gives:

$$w_{k_1,\,i_1,\,p_t,\,t} + w_{k_2,\,i_2,\,p_t,\,t} + w_{k_1,\,i_1,j_1,\,t} + w_{k_2,\,i_2,j_1,\,t} + w_{k_1,\,i_2,\,p_t,\,t} + w_{k_2,\,i_1,\,p_t,\,t} + w_{k_1,\,i_2,j_2,\,t} + w_{k_2,\,i_1,j_2,\,t} = 4$$

and the right hand side (RHS) of these constraints is 3, so the non-integer solution does not satisfy this constraint.

To show that this constraint is valid for the integer solutions, consider the LHS for integer solutions.

$$w_{k_1,\,i_1,\,p_t,\,t} = 1 - x_{k_1,\,i_1}$$
$$w_{k_2,\,i_2,\,p_t,\,t} = 1 - x_{k_2,\,i_2}$$
$$w_{k_1,\,i_2,\,p_t,\,t} = 1 - x_{k_1,\,i_2}$$
$$w_{k_2,\,i_1,\,p_t,\,t} = 1 - x_{k_2,\,i_1}$$

This constraint can be simplified to:

$$1 - x_{k_1,\,i_1} + 1 - x_{k_2,\,i_2} + 1 - x_{k_1,\,i_2} + 1 - x_{k_2,\,i_1} + w_{k_1,\,i_1,j_1,\,t} + w_{k_2,\,i_2,j_1,\,t} + w_{k_1,\,i_2,j_2,\,t} + w_{k_2,\,i_1,j_2,\,t} \le 3,$$

which further simplifies to

$$4 - x_{k_1,\,i_1} - x_{k_2,\,i_2} - x_{k_1,\,i_2} - x_{k_2,\,i_1} + w_{k_1,\,i_1,j_1,\,t} + w_{k_2,\,i_2,j_1,\,t} + w_{k_1,\,i_2,j_2,\,t} + w_{k_2,\,i_1,j_2,\,t} \le 3,$$

which can be rewritten as

$$w_{k_1,\,i_1,j_1,\,t} + w_{k_2,\,i_2,j_1,\,t} + w_{k_1,\,i_2,j_2,\,t} + w_{k_2,\,i_1,j_2,\,t} \le -1 + x_{k_1,\,i_1} + x_{k_2,\,i_2} + x_{k_1,\,i_2} + x_{k_2,\,i_1}.$$

But $x_{k_1,i_1} + x_{k_1,i_2} \le 1$ and $x_{k_2,i_1} + x_{k_2,i_2} \le 1$, so $w_{k_1,\,i_1,j_1,\,t} + w_{k_2,\,i_2,j_1,\,t} + w_{k_1,\,i_2,j_2,\,t} + w_{k_2,\,i_1,j_2,\,t} \le 1$

$w_{k_1,\,i_1,j_1,\,t}$ and $w_{k_2,\,i_2,j_1,\,t}$ cannot both be 1 because (j_1, t) only has a demand of 1 in the example given.

$w_{k_1, i_2, j_2, t}$ and $w_{k_2, i_1, j_2, t}$ cannot both be 1 because (j_2, t) only has a demand of 1 in the example given.

If $w_{k_1, i_1, j_1, t}$ and $w_{k_1, i_2, j_2, t}$ are both 1, then this implies that $w_{k_1, i_1, p_t, t}$ and $w_{k_1, i_2, p_t, t}$ are both 0, which implies that x_{k_1,i_1} and x_{k_1,i_2} are both 1. But $x_{k_1,i_1} + x_{k_1,i_2} \leq 1$, so this cannot happen.

If $w_{k_1,i_1,j_1,t}$ and $w_{k_2,i_1,j_2,t}$ are both 1, then this implies that $w_{k_1,i_1,p_t,t}$ and $w_{k_2,i_1,p_t,t}$ are both 0, which implies that x_{k_1, i_1} and x_{k_2, i_1} are both 1. But $x_{k_1,i_1} + x_{k_2,i_1} \leq 1$, so this cannot happen.

If $w_{k_2,i_2,j_1,t}$ and $w_{k_1,i_2,j_2,t}$ are both 1, then this implies that $w_{k_2,i_2,p_t,t}$ and $w_{k_1,i_2,p_t,t}$ are both 0, which implies that x_{k_2,i_2} and x_{k_1,i_2} are both 1. But $x_{k_1,i_2} + x_{k_2,i_2} \leq 1$, so this cannot happen.

If $w_{k_2,i_2,j_1,t}$ and $w_{k_2,i_1,j_2,t}$ are both 1, then this implies that $w_{k_2,i_2,p_t,t}$ and $w_{k_2,i_1,p_t,t}$ are both 0, which implies that x_{k_2,i_2} and x_{k_2, i_1} are both 1. But $x_{k_2,i_1} + x_{k_2,i_2} \leq 1$, so this cannot happen.

This covers all 6 cases, and no integer solution fails this constraint, hence it is a valid inequality.

Q.E.D.

Non-integer extreme points similar to the one listed in Theorem 10 arise in all three formulations listed here. These extreme points occur when more than one stage one allocation is attempting to be rescheduled to the same stage two slot in some scenario. Theorem 11 provides a means to eliminate such extreme points in the Allocation-Based formulation. The existence of similar cuts for the Flight-Based and Queue-Based formulations is a topic of further research.

2.5 Computational Results

The preceding results show that an integer optimal solution always exists for the total expected delay objective function. Thus, the LP-relaxation will solve the IP and fast times will result. In order to evaluate the strength of the above formulations more generally the objective function was slightly modified. CDM human-in-loop experiments and the subsequent widespread use of RBS, led to the general acceptance of RBS s an allocation standard. Research has shown that RBS has fundamental properties required of a fair allocation method (Vossen et al., 2003) (Vossen and Ball, 2006). Instead of minimizing the total expected delay, the "pure" RBS solution was seen as the "ideal" solution and a term was added to the objective function which seeks to minimize the total deviation from RBS. This is done by first computing a term *cost*(k, i) = *time*(i) – *arr*(k). The following term is the added to each of the objective functions:

$$\min \sum_{k \in Flights} \sum_{i \in Slots_1} \mathrm{cost}(k,i) \cdot x(k,i)$$

It is of interest to test the computational performance of IP solvers with this new objective function; of particular note is whether or not the LP-relaxation will return an integer solution. It is interesting that in all our examples, this was the case.

A test data set was created such that the expected GDP duration was 4 hours, with probabilities of the GDP ending after 60, 120, 180, and 240 minutes. The probabilities associated with these end times were 0.1, 0.2, 0.3, and 0.4 respectively.

The following graphs compare the performance of the three different formulations. The experiments were run on a PC with Two quad-core Xeon processors, 12GB RAM, and XpressMP 2008A.

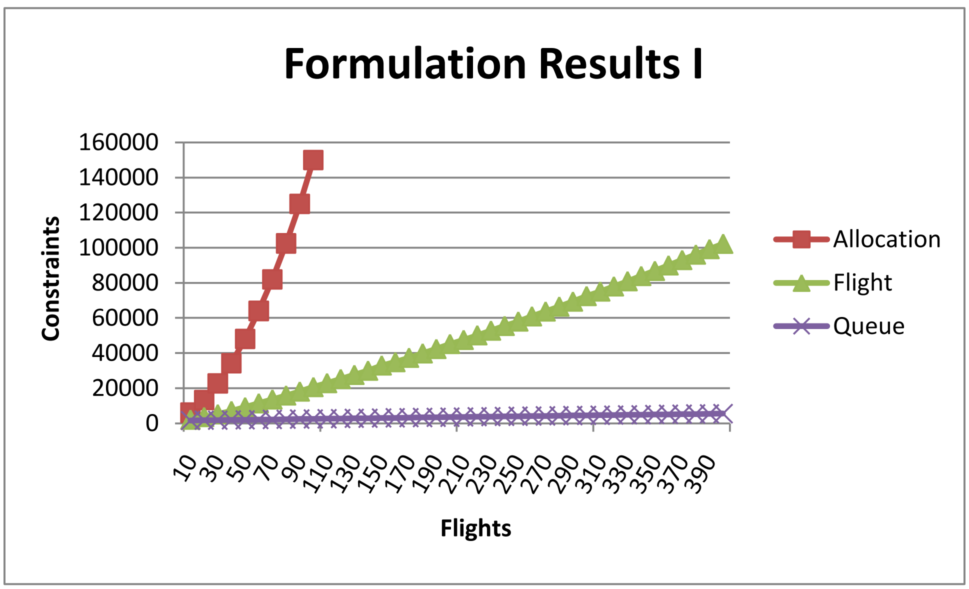

Figure 2.8: Constraint Comparison of the Formulations Presented

Figure 2.8 shows that the number of constraints in the Allocation-Based formulation grows at a much more rapid pace than either of the other two formulations. Notice also that, as the number of flights increases above 100, there is a significant difference between the Queue-Based and Flight-Based Formulations.

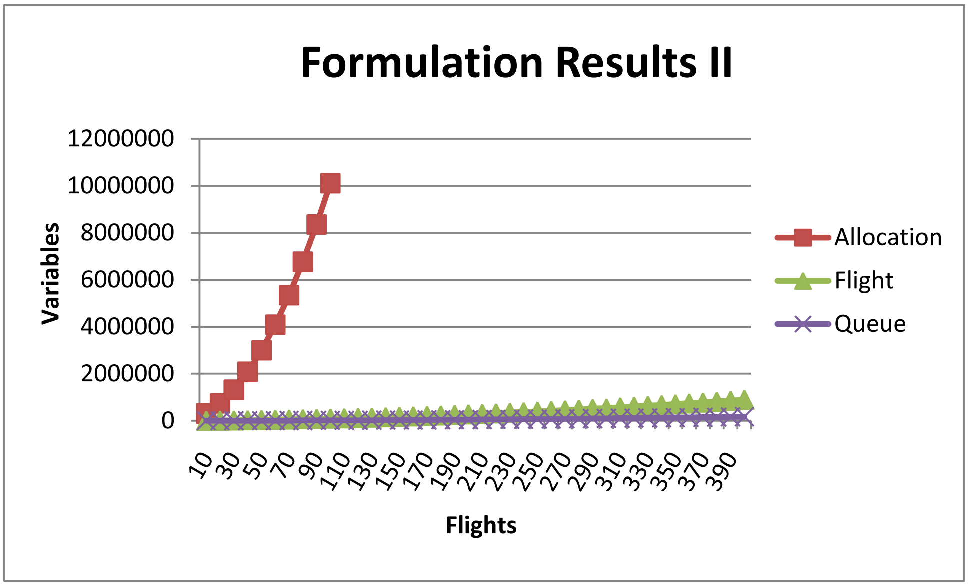

Figure 2.9: Variable Comparison of the Formulations Presented

Figure 2.9 shows the growth in the number of variables as a function of the number of flights. Once again, the Allocation-Based Formulation grows at the fastest rate. The Flight-Based and Queue-Based formulations had a variable count that was both much smaller, but difference eventually became apparent. Again, the Queue-Based formulation remained relatively small throughout.

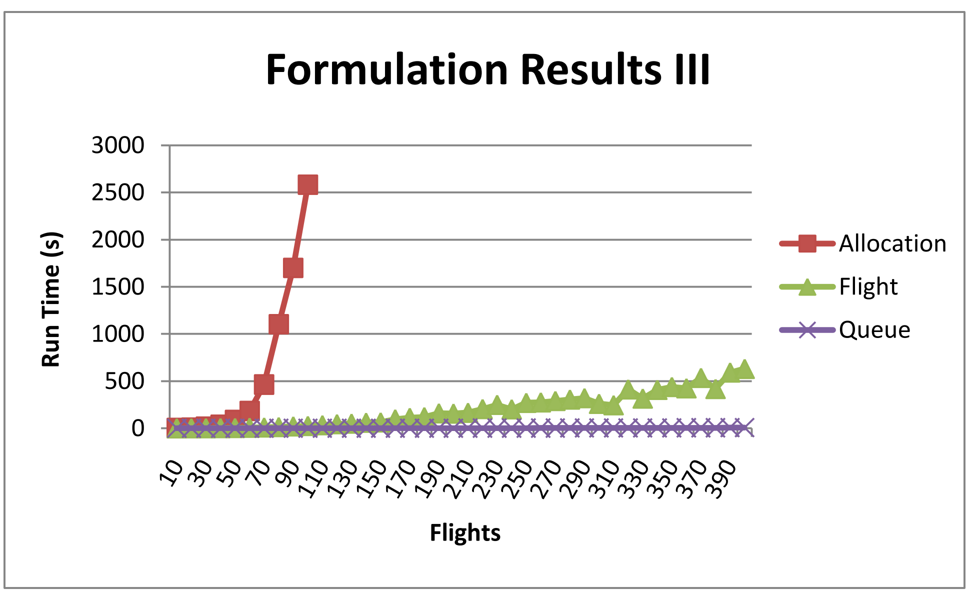

Figure 2.10: Run Time Comparison of the Formulations Presented

Figure 2.10 shows growth in computation time as a function of the number of flights. Here, the Queue-Based formulation provides the most efficient run times, often not exceeding 5 seconds. This makes the Queue-Based formulation an ideal target of investigation for further study.

Chapter 3 Models that Trade-off Equity and Efficiency

As discussed in Chapter 1, there are many challenges to effective GDP planning. One primary challenge is that of equity. Before the current standard of Collaborative Decision Making was adopted, participants felt that GDPs were implemented in an inequitable manner. Airlines were reluctant to provide information to the FAA because they (correctly) felt providing such information could give a much greater benefit to their competition than to the airline providing the information. This lack of equity and incentives led to inefficient solution procedures and often resulted in more system delay. CDM was initiated to resolve these issues by instituting methods that were based on agreed upon standards and allocation procedures that provided incentives for participation with accurate timely information (Ball et al., 2007), (Vossen et al., 2003), (Vossen and Ball, 2006), (Chang et al., 2001).

One of the major CDM components is the ration-by-schedule (RBS) principle, which decoupled the information provided by the airlines on a day of operations and the resources they received. The RBS principle provides that slots be allocated on a first-scheduled-first-served basis, so in a GDP, flights are kept in the order that they were originally scheduled. Some flights, though, are exempt from RBS. One set, flights that have already taken off, obviously cannot be given ground delay and must be exempt. The other set, though, has a more subtle justification.

Because of the stochastic nature of weather, an air traffic manager is reluctant to delay a flight several hours in advance of a storm that may or may not materialize.

An overly pessimistic forecast could result in some longer flights being given what, in hindsight, is unnecessary delay. To offset this, a distance radius is set from the troubled airport, and ground delays are only assigned to flights that originate inside that radius. The remaining flights are exempt from this GDP. This second form of RBS will be referred to as distance based ration by schedule (DB-RBS) (Ball and Lulli, 2004).

Ball et al. (Ball, Hoffman and Mukherjee, 2010) developed a formal stochastic model of GDP's to gain a fundamental understanding of how giving preferential treatment to long-haul flights improves expected GDP performance. They proposed the ration-by-distance algorithm, which allocates flights to arrival time slots using a priority scheme based on flight list ordered by decreasing flight length. This algorithm is structurally the same as RBS, but with an alternative priority scheme. They showed that RBD, under a fairly general model of GDP dynamics, minimizes the expected delay (Ball, Hoffman and Mukherjee, 2010). It is easy to see, however, that RBD can generate an inequitable distribution of flight delays. To address this problem, they proposed a heuristic algorithm, E-RBD, which seeks to balance efficiency and equity. In this chapter, some IPs are formulated, which represent the stochastic dynamic ground holding problem (SDGDP). It will be shown that this IP can more precisely balance efficiency and equity to a larger scale than either RBS, RBD or E-RBD.

3.1 Related Work

The GDP is a well-studied problem in aviation research. Odoni first proposed an IP model for the Ground Holding Problem (Odoni, 1987). Bertsimas and Stock Patterson formulated a model to address issues concerned with congestion in the National Airspace System (NAS) (Bertsimas and Stock Patterson, 1998). This model minimizes the total ground delay and airborne delay, while ensuring that the departure capacities, arrival capacities, sector capacities and time connectivity constraints are not violated. Although, the model is for the general ATFM problem, it can easily be adapted to represent the Single Airport Ground Holding Problem (SAGHP) and Multiple Airport Ground Holding Problem (MAGHP). This model allows for adjustments to the timing of flights. A second model by the same authors also allowed for route alternatives (Bertsimas and Stock Patterson, 2000). These models are deterministic and do not account for the ways that the weather uncertainty can play into the planning of a GDP.

Ball and Lulli (Ball and Lulli, 2004) showed how the decision of which flights to include in a GDP affects the performance of that GDP. This is directly related to the time that the command center commits to implementing a GDP. Earlier file times imply that more flights can be assigned ground delay and thus included in the GDP. These file times, though, are also based on less accurate weather forecasts and make it more likely that some flights (particularly long flights) will be assigned unnecessary delay. On the other hand, later file times include fewer flights in the GDP, which make the amount of ground delay per flight greater. As a result they concluded that the problem of determining the included set for a GDP should be done based both on the average delay assigned to flights and the expected cost of ground delay that is unnecessarily assigned.

They defined the distance based criteria of exemption, where a circle is drawn around the airport experiencing the GDP. Fights that depart from airports inside this circle are included in the GDP, while flights that depart from outside the circle are exempted (Ball and Lulli, 2004). There are many possible sets of included flights, based on how large the radius is set from the airport experiencing the GDP. When we refer to DB-RBS, we will be referring to RBS with various exemption radii, as defined by Ball and Lulli.

Vossen et al. (Vossen et al., 2003) provided a justification for why the "pure" RBS allocation is deemed as the most equitable. They showed that it is better to measure equity relative to RBS rather than to compare simple statistics like the average delay of a carrier. They also found that the exemption radius of DB-RBS can have a bias towards airlines operating long haul flights. A similar result would hold for RBD. Thus, in order to present equitable solutions, it is of interest to minimize this deviation from 'pure' RBS. There are several different ways of minimizing the deviation from RBS though: maximum deviation, total deviation, five worse, ten worse, etc. The IPs generated in this chapter will take different metrics of equity into account in attempt to balance equity and efficiency.

There are four main papers that studied IP approaches to the stochastic GDPs. These are given by Richetta and Odoni (Richetta and Odoni, 1994), Ball et al.(Ball et al., 2003), Kotynek and Richetta (Kotnyek and Richetta, 2006) and Mukherjee and Hansen (Mukherjee and Hansen, 2007). The first three present static-stochastic models, in the sense that decisions are made at the start of a GDP, but cannot be adjusted throughout the duration of a GDP as new information becomes available.

The last presents a dynamic-stochastic model which prepares to model the uncertainty of new information becoming available and making an initial decision that's able to take advantage of that new information.

The first IP approach to stochastic GDPs was given by (Richetta and Odoni, 1994). This paper seeks to minimize the ground delay and expected airborne delay given to flights. This was minimized over a finite set of scenarios of airport capacity profiles.

Ball et al. next studied a stochastic case of GDPs (Ball et al., 2003). In this problem, they were concerned with Airport Arrival Rates (AARs), the number of flights the airport can receive in a given time period, in an environment where the weather is uncertain. The model takes into account an AAR distribution, and produces a planned AAR (PAAR) vector, which is the number of flights that the airport should schedule to arrive in each time period, given the stochastic nature of the weather and the probabilities of different AARs.

Kotnyek and Richetta then showed that a model first proposed by Richetta and Odoni (Richetta and Odoni, 1994) could also be used to produce the PAAR vector (Kotnyek and Richetta, 2006). Comparisons were then made between the Ball et al. model and the Richetta-Odoni model. The Richetta-Odoni model is larger in size than the Ball et al. model, but because its cost function for ground delay is more general, it allows for more specific adjustments of the relationship between the costs of airborne holding and ground holding. However, the model may still not be compatible with CDM processes (Vossen and Ball, 2006).

All these models operate under the condition of weather uncertainty. Due to the excessive costs of airborne holding when compared to that ground holding, the papers try to avoid the situation where an airport has more flights seeking to land than it has landing slots available in a given time period. It is also possible to have a larger number of available landing slots than flights seeking landing. Such a situation can arise when an airport expects bad weather and flights are given more ground delay than necessary to offset this weather. In these situations the airport would like to be able to re-schedule flights to utilize this unexpected capacity. Because the papers by Richetta and Odoni, Ball et al. and Kotnyek and Richetta consider only the static case of stochastic ground delay programs, their models do not allow us to adjust the delays dynamically as the weather changes.

Mukherjee and Hansen presented a dynamic stochastic IP formulation for the stochastic GDP, which took as part of its input the possible changes the weather can take throughout the duration of the GDP (Mukherjee and Hansen, 2007). This formulation employed a scenario tree to capture all the possible changes in weather outcomes. This scenario tree can grow large in size, which can lead to computational challenges. In the work presented here we take advantage of certain problem structures to invoke scenario trees of smaller size.

In (Ball, Hoffman and Mukherjee, 2010), Ball et al. consider the problem of maximizing the throughput into the airport. Here, the RBD algorithm is first proposed. The authors prove that the RBD algorithm minimizes total expected delay if the GDP cancels earlier than anticipated. In their proof, the authors were able to compare the total expected delay of the RBD allocation with that of other allocations and are able to show optimality. This algorithm, though, by definition, gives preference to airlines operating long flights. By providing a formal basis for prioritization based on distance this paper justified the distance based exemptions of (Ball and Lulli, 2004).

3.2 Formulations

We now review and summarize the Queue-Based Formulation from Chapter 2. It depends on two sources of input, flight based input and airport based input. For each flight $k \in Flights$, the following input is provided:

1. *arr*(*k*) – the published arrival time of the flight *k*
2. *len*(*k*) – the length of the flight *k*
3. *rbs*(*k*) – where the 'pure' RBS algorithm would allocate the flight *k* in a GDP.

Likewise, the following input is provided for the airport experiencing the GDP:

4. For each landing slot, *i*, the reduced capacity of this landing slot, $cap_1(i)$.
5. A list of possible GDP cancellation times, called scenarios, indexed by t = 1…T. Each scenario, *t*, has its own corresponding time $\tau(t)$.
6. The nominal capacity of the airport landing slot *j*, in scenario *t*, $cap_2(j,t)$.
7. A discrete probability distribution *p* over the set of cancellation times t = 1…T.
8. Each slot has an associated time. If the slot is a stage one slot, *i*, this is represented by *time*(*i*), and if it is a stage two slot *j* in the scenario *t*, then the associated time is represented by *time*(*j*, *t*).

Taking this input into account, the Queue-Based formulation presented in Chapter 2 can be stated as follows:

$$\min \sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{t \in Scenarios} \left(p_t \left(time(j,t) - arr(k) \right) \right) \tag{3.1}$$

subject to

$$\sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} x_{k,i} = 1 \text{ for each flight } k \tag{3.2}$$

$$\sum_{\substack{k \in Flights \\ | arr(k) \leq time(i)}} x_{k,i} \leq cap_1(i) \text{ for each arrival slot } i. \tag{3.3}$$

$$\sum_{(k,i) | earliest(k,i,t) = j} x_{k,i} + z_{j-1,t} - z_{j,t} - u_{j,t} = 0 \tag{3.4}$$

for each arrival slot j and each scenario t

$$u_{j,t} \leq cap_2(j) \tag{3.5}$$

for each arrival slot j and each scenario$t$

$$x_{k,i} \in \{0,1\}$$

$$u_{j,t}, z_{j,t} \geq 0, u_{j,t}, z_{j,t} \in \square$$

The function $earliest(k,i,t)$ was defined in Chapter 2.4.2. This is briefly described below.

$$earliest(k,i,t) = \begin{cases} time(i) & \text{if } time(i) - len(k) \leq \tau(t) \\ \tau(t) + len(k) & \begin{array}{c} \text{if } \tau(t) + len(k) \geq arr(k) \\ \text{and} \\ time(i) - len(k) > \tau(t) \end{array} \\ arr(k) & \text{if } \tau(t) + len(k) < arr(k) \end{cases}$$

Because of possible inequity in a RBD solution Ball et al. proposed E-RBD. E-RBD finds an "RBD-like" solution with a maximum allowed deviation from RBS.

The efficiency of the solution generated by the E-RBD solution was left as an open question. To answer this question, we pose the problem of minimizing the total expected delay of all flights in a GDP subject to a set of possible cancellation times, each with its own probability. We will refer to this problem as the Two-Stage Stochastic Dynamic Ground Delay Program with Maximum Deviation (TSDG -MD). It should be noted though that an E-RBD solutions does not depend on any probability distribution, while TSDG-MD does.

To formulate TSDG-MD as an IP, a change only needs to be made to constraints (3.2) and (3.3), where the new constraints are

$$\sum_{\substack{i \in Slots_1 \\ time(i) = arr(k) \\ time(i) \leq RBS(k) + \delta}} x_{k,i} = 1 \text{ for each flight } k \tag{3.6}$$

$$\sum_{\substack{k \in Flights \\ arr(k) \leq time(i) \leq RBS(k) + \delta}} x_{k,i} \leq cap_1(i) \text{ for each arrival slot } i \tag{3.7}$$

These constraints merely place an upper bound on the set of slots to which a flight can be initially assigned. If the objective function remains minimizing the total expected delay, then this new IP will provide an optimal solution to the TSDG-MD problem.

The TSDG-MD problem, though, is not the only way to find more equitable solutions that also seek to minimize total expected delay in a GDP. A second consideration would be a weighted objective between total expected delay and *total deviation from RBS*. We will call this problem Two-Stage Stochastic Dynamic Ground Delay Program with Weighted Objective (TSDG-WO). This again can be modeled by the IP described by (3.2) – (3.5), with the objective function now as:

$$\min \ \gamma \sum_{i \in Slots} \sum_{k \in Flights} cost(k,i) x_{k,i} + \\ (1-\gamma) \sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{t \in \{1..T\}} \Big(p(t) \big(time(j,t) - arr(k) \big) \Big) \tag{3.8}$$

Here, *cost*(*k, i*) is the deviation of the stage one assignment of flight *k* from its RBS slot, i.e. $cost(k, i) = 0$ if $time(i) \le \text{RBS}(k)$ and $\text{cost}(k, i) = time(i) - \text{RBS}(k)$ otherwise. The parameter γ with $0 \le \gamma \le 1$, can be varied to adjust the weight given to the two objective function components.

Because both the TSDG-MD and TSDG-WO problems can be modeled by modifications of the IP proposed in Chapter 2, these modifications can be combined into an IP that seeks to minimize the weighted objective of total expected delay and total deviation from the RBS allocation, while placing a maximum limitation on the deviation of any flight's initial allocation from its pure RBS allocation. We will call this problem Two-Stage Stochastic Dynamic Ground Delay Program with Max Deviation and Weighted Objective (TSDG-MW). This new IP has the objective function of (3.8) and constraints (3.4) – (3.7) together with nonnegative integer variable restrictions.

3.3 Heuristics

Chapter 2 showed that the IP presented in (3.1) – (3.5) with the objective function of minimizing total expected delay has an optimal solution where the stage one solution is generated by the RBD algorithm. This result no longer holds if the TSDG-MD or TSDG-WO formulations are used. Although the IPs generated here have provided fast run times, in a CDM setting, people more readily associate fairness with a basic allocation principle and an allocation method implementing that principle. IP solutions are seen as more of a black box, and the understanding behind how the solutions are reached is not always understood. Thus it is important to formulate heuristics which take advantage of this problem's particular structure.

3.3.1 GreedySlot and GreedyDist

One heuristic was originally proposed by (Ball, Hoffman and Mukherjee, 2010). Here the slots are ordered by their arrival times and the heuristic repeatedly assigns the earliest remaining slot to the longest flight that can be allocated to that slot, so long as this assignment does not force any flights to violate the maximum deviation constraints. Accordingly, this heuristic will be denoted GreedySlot. It can be formalized as follows:

GreedySlot Algorithm

1. Initially each flight is temporarily assigned to its RBS slot, temp(k).
2. If any flight *k* has only one possible assignment *i*, permanently assign flight *k* to slot *i* and goto 2.
3. If there are flights that have not been permanently assigned, let i_E be the earliest slot with capacity at least 1.
4. If there are no flights that can be assigned to i_E, set the capacity of i_E to 0 and goto 3.
5. Let poss(i_E) be the set of flights that can be assigned to i_E.
6. Let k_0 be the longest flight in poss(i_E).
7. If every slot *i* between i_E and temp(k_0) that has capacity at least 1 has a flight *k* temporarily assigned to it that can be assigned to a later slot, temporarily assign each of these flights *k* to a later slot and permanently assign flight k_0 to slot i_E and goto 2.
8. Remove k_0 from poss(i_E).
9. Goto 6.
10. Return the permanent assignment of the flights.

Figure 3.1: Pseudo code for the GreedySlot Algorithm

Similarly, instead of rationing based on the order of the slots, a heuristic can be formulated by first sorting the flights by their lengths, and repeatedly assigning the longest flight to the earliest unused slot that does not cause any other flight to violate its maximum deviation constraint. More formally, it can be stated as:

GreedyDist Algorithm

1. Initially each flight is temporarily assigned to its RBS slot, temp(k).
2. If any flight *k* has only one possible assignment *i*, permanently assign flight *k* to slot *i* and goto 2.
3. If there are flights that have not been permanently assigned, let k_L be the longest remaining flight.
4. Let want(k_L) be the set of slots that k_L can be assigned to.
5. Let i_0 be the longest flight in want(k_L).
6. If every slot *i* between i_0 and temp(k_L) that has capacity at least 1 has a flight *k* temporarily assigned to it that can be assigned to a later slot, temporarily assign each of these flights k to a later slot and permanently assign flight k_L to slot i_0 and goto 2.
7. Remove i_0 from want(k_L).
8. Goto 6.
9. Return the permanent assignment of the flights.

Figure 3.2: Pseudo code for the GreedyDist Algorithm

Example 3.1

To understand the difference between these two heuristics, consider the following example of six flights flying into an airport whose capacity is reduced from 60 flights per hour to 12 flights per hour, or one flight per 5 minutes. Suppose that the maximum deviation allowed is 5 minutes (one arrival slot). The input is as follows:

Flight, *k*	*arr*(*k*)	*rbs*(*k*)	*len*(*k*)	*dep*(*k*)=*arr*(*k*) – *len*(*k*)
1	5:01	5:05	60	4:01
2	5:02	5:10	65	3:57
3	5:03	5:15	75	3:48
4	5:04	5:20	90	3:34
5	5:05	5:25	120	3:05
6	5:06	5:30	150	2:36

Table 3.1: Input Data for the GreedySlot and GreedyDist algorithms

The GreedySlot and GreedyDist algorithms would allocate the flights to slots as illustrated in Figure 3.3:

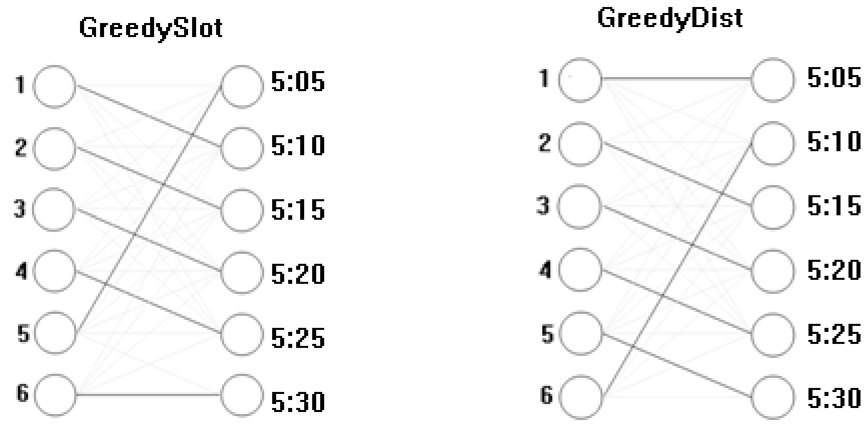

Figure 3.3: Execution of the GreedySlot and GreedyDist algorithms on Example 3.1

GreedySlot starts with the earliest slot (5:05) and finds a flight to assign to it. The longest available flight for that slot is Flight 5, so this flight is assigned to the 5:05 time slot. Flights 1, 2, 3, and 4 are iteratively each fixed to slots 5:10, 5:15, 5:20 and 5:25 because they have reached their maximum deviation. When 5:30 is reached Flight 6 is the only feasible flight and it is assigned to 5:30. Similar reasoning explains the GreedyDist solution.

Although both these heuristics provide feasible solutions to the TSDG-MD problem, no result currently exists giving conditions under which either is optimal. In fact, a simple example can show that neither of these solutions provides a general optimal solution to the TSDG-MD problem.

Example 3.2

Consider the Flight-Based input from Example 3.1, with the addition of 7 possible GDP cancellation times and the assumption that at the GDP cancellation, the nominal capacity of 60 flights per hour, or a flight per minute is restored. Consider also the two probability distributions that are as follows:

	3:05	3:20	3:35	3:50	4:05	4:20	4:35
p_1	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{64}$
p_2	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{32}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$

Table 3.2: GDP cancellation times and associated probabilities

With this additional input, the GreedySlot and GreedyDist heuristics can each be evaluated for the GDP under the two different probability distributions. They performed as follows:

	p_1	p_2
GreedySlot	30.2188	76.625
GreedyDist	19.875	77.2344

Table 3.3: Performance of the GreedySlot and GreedyDist algorithms on Example 3.2

Notice that in Example 3.2, under p_1, GreedyDist has less total expected delay than GreedySlot, thus showing that GreedySlot does not always provide an optimal solution to the TSDG-MD problem. Notice also that under p_2, GreedySlot has less total expected delay than GreedyDist, thus showing that GreedyDist does not always provide an optimal solution to the TSDG-MD problem. It should also be noted that in Example 3.2, the optimal solution varies with the probability distribution. Under p_1, GreedyDist returned the optimal solution, while under p_2, GreedySlot returned the optimal solution.

The fact that these solutions do not always provide optimal solutions to the TSDG-MD problem does not make them unable to find close to optimal solutions. Experiments will show how close to optimal these solutions generated by the GreedySlot and GreedyDist heuristics actually are.

3.3.2 The Infinite Capacity Solution

In Chapter 3.2, three different means for producing solutions that seek to balance both equity and efficiency were provided: TSDG-MD, TSDG-WO, and TSDG-MW. The heuristics mentioned so far were constructed to only provide solutions for one of those methods. Because all three of these methods provide different approaches to this equity/efficiency trade-off, it is of interest to find ways to generate near optimal solutions to each of these.

All three of these methods were formulated as IPs, and there is a host of existing literature on IPs. One IP that has been well studied is the transportation problem. In a transportation problem, there are m plants, each with supply s_i (for i = 1 to m) and n warehouses each with demand d_j (for j = *1* to n). The assumption is that $\sum_i s_i \geq \sum_j d_j$ because if not, then no feasible solution exists to this problem (demand exceeds supply). The cost for transporting a unit from plant i to warehouse j is c_{ij}. The goal is to minimize total transportation costs. This problem can be formulated as an IP as follows, where x_{ij} represents the amount of units shipped from plant i to warehouse j:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{3.9}$$

subject to

$$\sum_{j=1}^{n} x_{ij} \leq s_i \text{ for } i \in \{1,\ldots,\mathrm{m}\} \tag{3.10}$$

$$\sum_{i=1}^{m} x_{ij} \geq d_j \text{ for } j \in \{1,\ldots,n\} \tag{3.11}$$

$$x_{ij} \geq 0, x_{ij} \in \mathbb{Z}$$

The transportation problem is known to have a TU constraint matrix. A primary consequence of this is that as long as the right hand sides are integer, then the optimal solution to the LP-relaxation will be integer.

This problem is of practical importance here because stage one of each of the three formulations mentioned in Chapter 3.2 is a transportation problem. This implies that, since the reduced capacity of every airport arrival slot is given in nonnegative integers, any formulation that consists of only these constraints will be guaranteed to give back an integer optimal solution to the LP-relaxation, assuming that a feasible solution to the problem exists.

The challenge becomes how to make optimal decisions about delaying flights in the event of the weather becoming clear earlier than anticipated, without the constraints dictating that the capacity constraints are not violated when the weather clears. One approach to this problem is to assume that the airport will go from a reduced capacity setting to an infinite capacity setting, rather than going to a nominal capacity setting. Such an assumption implies that at each possible GDP cancellation time, every flight will be rerouted to the earliest slot that it can reach. Because the assumption is that the slot has infinite capacity, there will be no need to restrict the number of flights that can be reallocated to this slot at this cancellation time. The earliest slot that a flight can reach at a possible GDP cancellation time is precisely what the function $earliest(k,i,t)$, as defined in Chapter 2, measures. Consequently, the following formulation will be an infinite capacity solution for the weighted objective metric.

$$\min \ \gamma \sum_{i \in Slots} \sum_{k \in Flights} cost(k,i) x_{k,i} + \\ (1-\gamma) \sum_{k \in Flights} \sum_{i \in Slots_1} \sum_{t \in \{1..T\}} \left(p_t \left(earliest(k,i,t) - arr(k) \right) x(k,i) \right) \tag{3.12}$$

subject to

$$\sum_{i \geq a(k)} x_{k,i} = 1 \text{ for each flight } k \tag{3.13}$$

$$\sum_{k|a(k) \leq i} x_{k,i} \leq cap_1(i) \text{ for each arrival slot } i. \tag{3.14}$$

The different versions of this IP will produce a solution feasible to stage one of the IPs. Because each of these is a stochastic IP with complete recourse, there is always a feasible stage two solution for each feasible stage one solution. These stage two solutions can be produced from the sub-problems that exist in each scenario of stage two of the IP. Because the sub-problems have TU constraint matrices and each scenario is independent of one another once the stage one solution has been determined, determining the efficiency of a stage one solution leaves little work to be completed. It should be noted that RBD, RBS, GreedySlot and GreedyDist all produce only stage one solutions as well, leaving stage two to be later determined.

3.4 Experimental Results

The formulations presented in this chapter were tested using data based on GDPs run on three different dates at San Francisco International Airport (SFO), La Guardia Airport (LGA), Newark International Airport (EWR), and Chicago O'Hare

International Airport (ORD). The experiment was set up with GDPs expected to run for a duration of six hours and seven possible weather clearance times. The tests were run with three different probability distributions: a uniform distribution, a distribution where the probabilities of weather clearance were decreasing, $p = \left[\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{64}\right]$, and a distribution where the probabilities of weather clearance were increasing, $p = \left[\frac{1}{64}, \frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}\right]$.

The following table represents the nominal and reduced capacity rates used on the data sets. These are general representatives of the respective capacities.

	Nominal	Reduced
SFO	60	30
LGA	40	30
ORD	100	72
EWR	50	33

Table 3.4: Nominal and Reduced capacities at select US airports

For TSDG-WO, the experiment was conducted on each day and distribution with γ ranging from 0 to 1, incrementing by 0.05. To obtain the TSDG-MD results, we used an objective function that measured only efficiency and restricted the allowed assignments of a flight to only δ slots after its RBS allocated slot, for each given $\delta \in \{0,1,...,30\}$. The TSDG-MW formulation was tested with γ ranging from 0 to 1, incrementing by 0.05, as was done with the TSDG-WO formulation. The maximum deviations for the TSDG-MW were $\delta \in \{0,1,...,30\}$, as was done with the TSDG-MD formulation. All three of the tested formulations give the RBS and RBD solutions at extreme parameter values. For instance, if the coefficient for equity is 1 in the weighted objective IP, the focus is only on equity. The result will then be the most equitable solution, RBS. On the other hand, if the coefficient for efficiency to 1 in the weighted objective IP, the focus is only on efficiency. The result will then be the solution that has the least total expected delay, which is an RBD solution.

The TSDG-MD formulation has similar properties. When the maximum deviation, δ, is set equal to 0, flights are not allowed to deviate from their RBS allocation, which is the optimal solution in that case. If δ is set to an arbitrarily large constant then all stage one assignments are allowed, in which case, the TSDG-MD formulation will output the RBD solution.

One key difference between the TSDG-WO IP and the TSDG-MD problem, though, is the fact that the TSDG-WO IP allows one to choose a γ small enough that it remains close to either the RBD or RBS solution, while still taking into account both equity and efficiency. For instance, instead of allowing the weighted objective to focus completely on efficiency, similar extreme values can be reached in the weighted objective IP when γ is 0.999. This helps find a "more equitable" solution with the same total expected delay as the RBD solution. Because the TSDG-MD problem does not promote or discourage this, similar considerations were made for an equity term in the objective function of the TSDG-MD problem. The result is a set of solutions that are optimal to the problem under consideration, but which also consider total deviation as well.

For the purposes of understanding the effectiveness of these metrics, they were all compared against the way GDPs are currently implemented, DB-RBS, with exemption radii set between 80 minutes and 400 minutes, incrementing by 20 minutes.

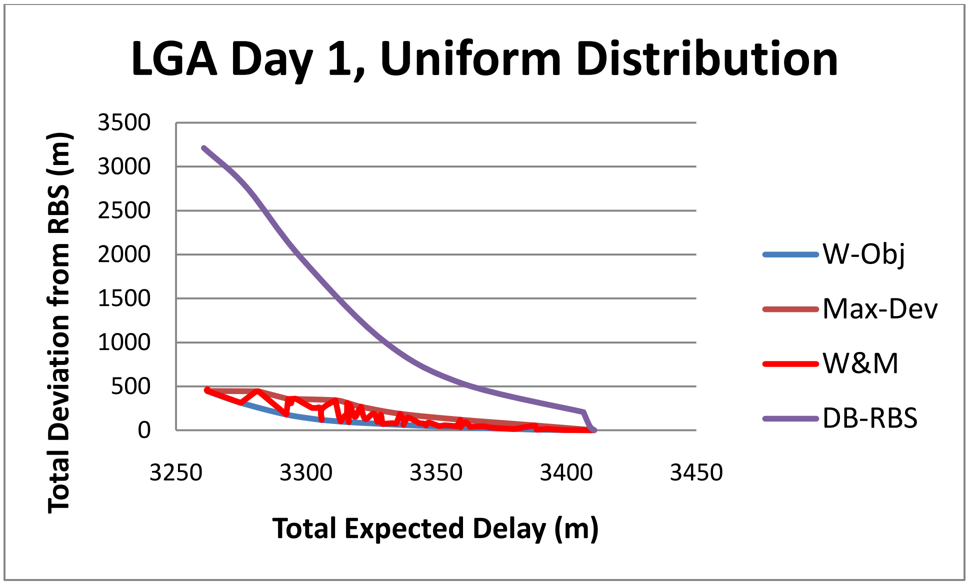

Figure 3.4: A comparison of the different metrics at EWR

There are several things that stand out about Figure 3.4. The most notable thing is the amount that the DB-RBS solution deviates from the IP generated solutions as it approaches its more efficient points. The three formulations introduced in 3.2 all had solutions that dominated the DB-RBS solution (i.e. the solutions had lower costs under both objective functions). In the 45 sample airports, days, and probabilities, this was a repeated occurrence, with the DB-RBS repeatedly producing solutions that were inefficient when the metric of equity is total deviation from RBS.

Naturally, a second question to ask is how these formulations compare to DB-RBS under a different metric of equity. Figure 3.5 measures total expected delay versus the metric of maximum deviation from RBS (instead of total deviation from RBS).

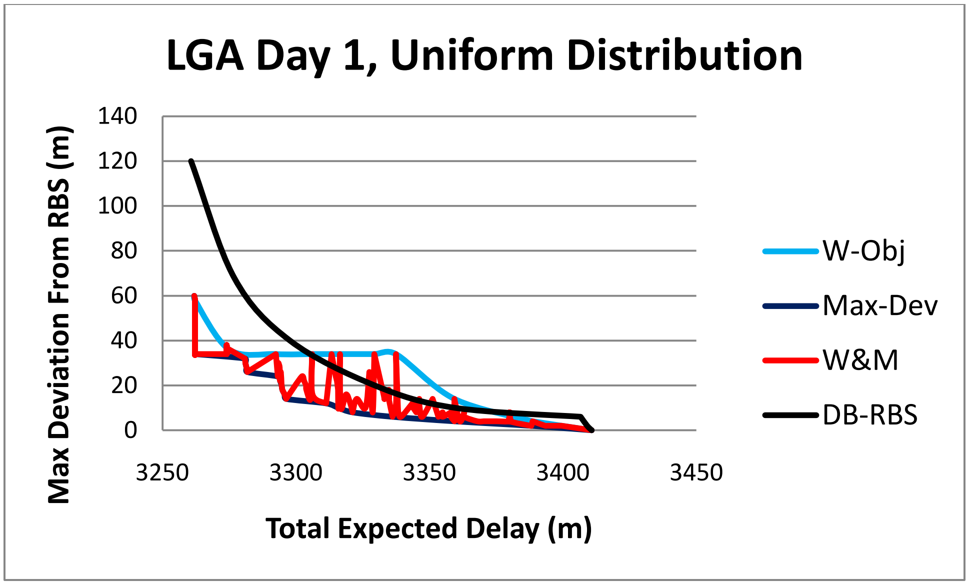

Figure 3.5: A comparison with a max deviation metric for equity

Here, notice that DB-RBS received its more efficient solutions by forcing some flights to experience staggeringly long amounts of delay. When DB-RBS achieves comparable results in terms of maximum deviation, the efficiency is greatly reduced. Also, TSDG-WO IP, which is the method that minimized both the total deviation from RBS and total expected delay in Figure 3.4 does not perform as well when the metric for equity is changed to maximum deviation from RBS.

An interesting point of both Figure 3.4 and Figure 3.5 is the number of solutions generated by TSDG-MW formulations. At its extremes, these solutions are equivalent to either the TSDG-MD solutions or the TSDG-WO solutions. However, there are a wide range of points that exist between the two solutions; thus offering the notion that this TSDG-MW IP may offer the ability to do a very fine grained trade-off analysis.

Because the three formulations tested are all IPs, another important question is the run time of these IPs. Figure 3.6 addresses that question. The experiments were run on a PC with Two quad-core Xeon processors, 12GB RAM, and XpressMP 2008A.

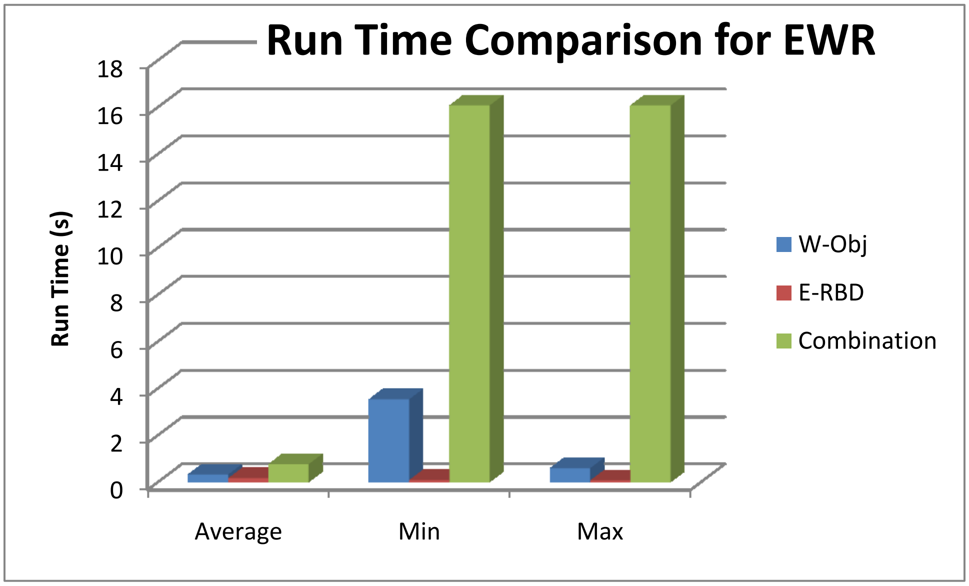

Figure 3.6: Run Time Comparison of the formulations

Figure 3.6 shows that these IPs can be used to solve real time problems relatively quickly. The results were similar for the other days, probabilities, and airports considered. Many of these solutions were generated by the LP-relaxations of the IPs.

A second set of tests performed was how often the heuristics offered were close to the optimal solution. To test the different heuristics, GreedySlot and GreedyDist were programmed in C++, while the Infinite Capacity Solution was programmed in Xpress. What was of interest was how often these methods were close to the optimal solutions and how often they give the actual optimal. Below are the results for the infinite capacity heuristic for the three days at SFO.

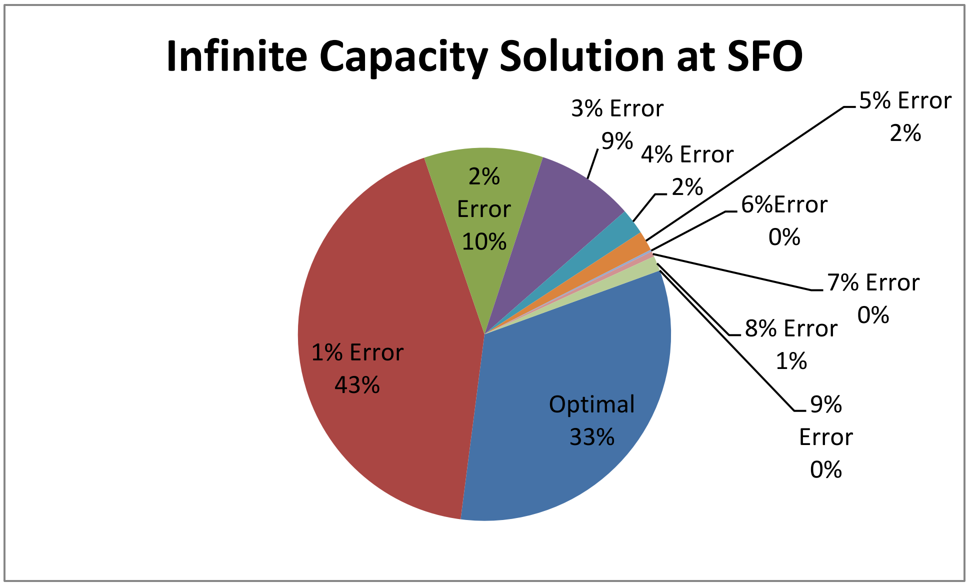

Figure 3.7: Performance of the Infinite Capacity Solution at SFO

Figure 3.7 shows that the Infinite Capacity Solution performed reasonably well in estimating the optimal solutions. Notice that it returned the optimal solution 33% of the time over the three days, three different probability distributions and 21 different equity to efficiency ratios at SFO. It was within 1% of optimal on 76% of these cases.

Next we considered the performance of the GreedySlot and GreedyDist heuristics. These heuristics could only be performed on the input given to the E-RBD problems, and thus could not take into account considerations for total deviation from RBS. Both the GreedySlot and GreedyDist heuristics achieve this level of performance without the need for a probability distribution.

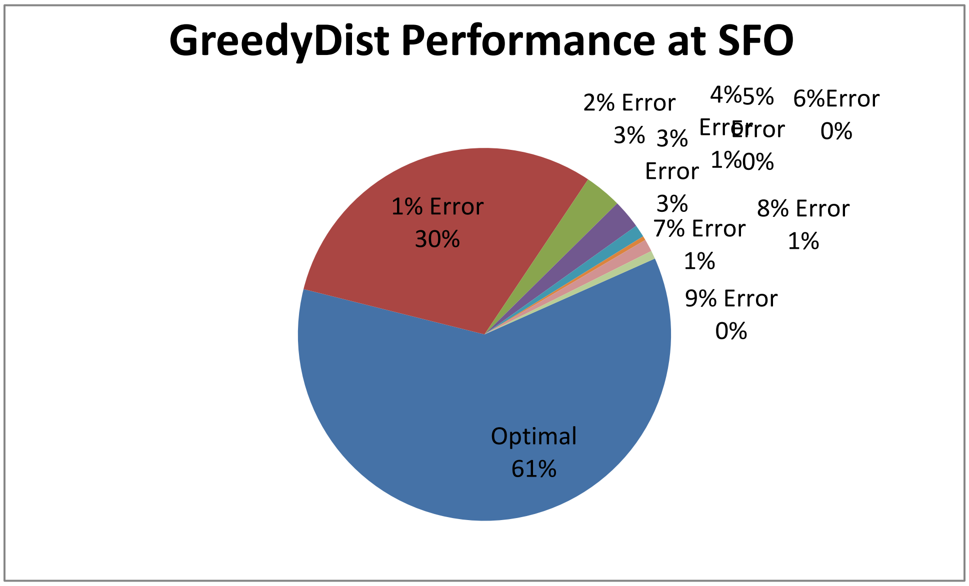

Figure 3.8: GreedyDist Performance at SFO

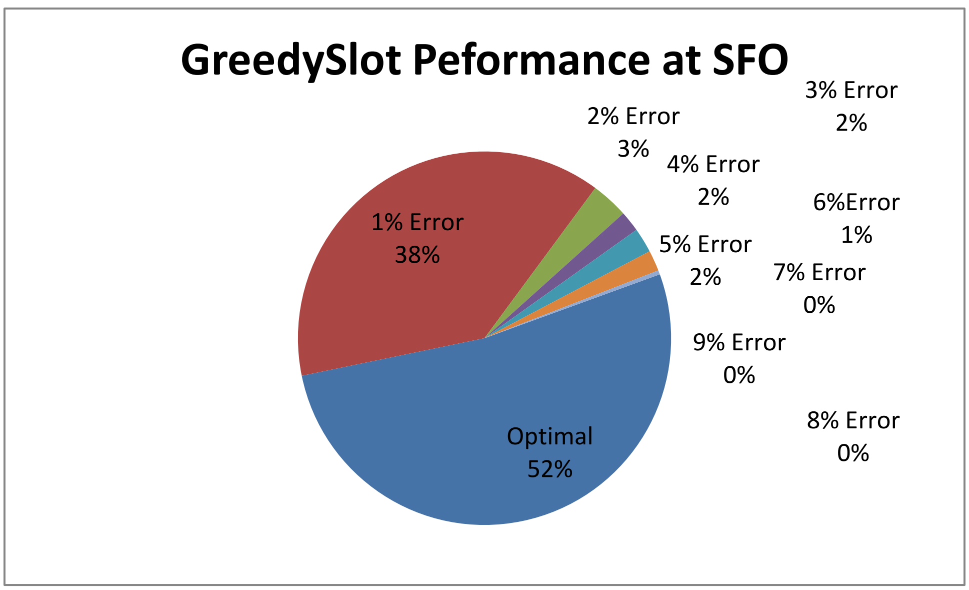

Figure 3.9: GreedySlot Performance at SFO

It should be noted though that Figure 3.8 and Figure 3.9 are a bit misleading because the GreedySlot and GreedyDist heuristics only work to generate solutions to the TSDG-MD problem, whereas the Infinite Capacity Solution can be used to for all the problems mentioned in this chapter. Below is a corresponding graph on the Infinite Capacity Solution on the TSDG-MD problems at SFO.

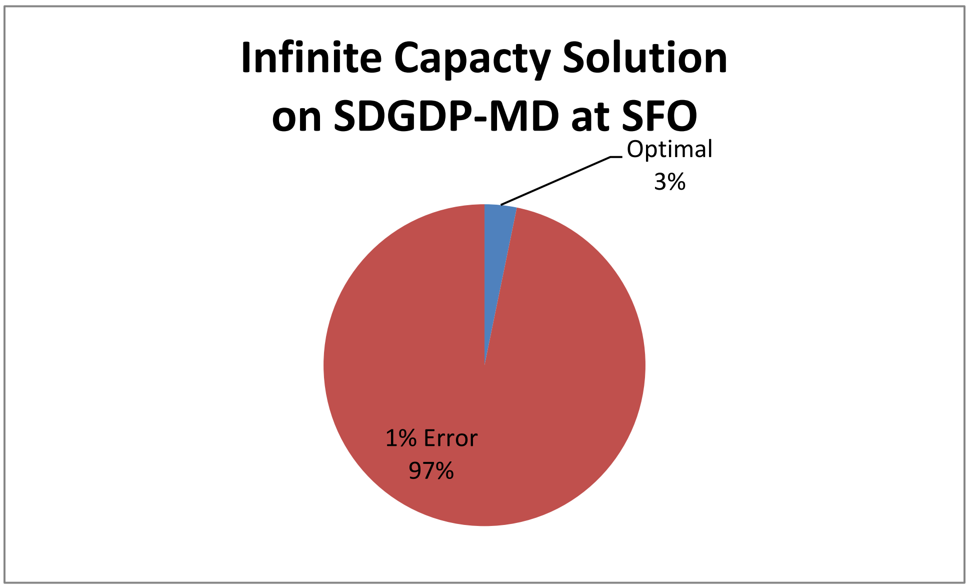

Figure 3.10: Infinite Capacity Solution on SDGDP-MD at SFO

This shows that the Infinite Capacity Solution does a good job in producing feasible solutions for the TSDG-MD problem. While these solutions are not optimal as often as the GreedySlot or GreedyDist heuristics, they seem to be closer to optimal on a more consistent basis, although with GreedySlot and GreedyDist providing solutions that are within 1% of optimal in 90 and 91 percent respectively of the cases run, it shows that all three heuristics do an admirable job of providing near-optimal solutions. It should be noted though that the Infinite Capacity Solution requires a probability distribution, whereas the GreedySlot and GreedyDist heuristics do not.

Chapter 4 En Route ATFM with Weather Uncertainty.

As a general rule, in the U.S. constraints that restrict air traffic flows arising from airport runway capacities are much more limiting than those associated with the en route airspace. However, the presence of bad weather will cause portions of the airspace to decrease in capacity and even temporarily close. With many airports and airspace routes already operating at or near capacity limits, this reduction in capacity can and often does lead to significant delays. Thus the FAA and the research community have been devoting more and more attention to problems associated with congested airspace.

In 2006, the FAA introduced the concept of Airspace Flow Programs (AFPs) (FAA, 2006). These initiatives are similar to GDPs, except that they are used to restrict flow through a volume of airspace, a Flow Constrained Area (FCA), instead of restricting flow into an airport. In general, flights may be given ground delay, reroute options, or airborne delay. Similar to the GDP though, it will be important to be able to make these decisions with knowledge that the weather may clear earlier than anticipated, thus allowing the AFP to be cancelled early. Our work uses as a starting point the model described by Ganji et al. who formulated an AFP planning problem as an integer program (Ganji et al., 2009). We refer to this integer program as the Ganji model.

Airspace capacity depends on a complex set of issues, including the workload that can be handled by the air traffic controllers and the structure of traffic flows through airspace (Mitchell, Polishchuk and Krozel, 2006). For instance, many flights do not seek to use the same portions of that sector at the same time. If a sector is long and narrow and has two flights attempting to fly through it at the same time, one flying through the top of the sector and one flying through the bottom of the sector. While currently sector capacity in a AFP is modeled similar to airport capacity in a GDP, the topic of how to define new sector capacity models is an active area of research.

Nilim et al. (Nilim et al., 2001) considered the problem of routing a single aircraft around multiple storms with the goal of minimizing time and fuel costs. There is a probability associated with each storm and the decision to fly toward the storm or around it is dependent on the probability of the prediction that the storm will be present. This takes into account the concept that the probabilities will be updated throughout the duration of the flight, and the decision of the optimal route to take is updated accordingly. The optimal routing strategy is determined using a Markov Decision Process and a dynamic programming algorithm. These authors later considered extending the model to handle multiple aircraft and multiple storm characteristics (Nilim, El Gahoui and Duong, 2003), (Nilim and El Gahoui, 2004). The complexity of these extended models, however, limits their ability to be used in practice. Our work builds directly on the Ganji model, but also uses concepts from these papers.

In this chapter we assume that the capacities under variant weather conditions of the FCA are estimated by some method and provided to our model as input. The chapter will then shows that the model for GDPs with weather uncertainty presented in Chapter 2 can be extended to a formulation that models AFP with weather uncertainty. Our analysis leads to a strengthening of the Ganji model, producing much faster solution times.

4.1 Problem Description

AFPs were introduced with three possible actions being taken on individual flights – ground delay, rerouting, or airborne delay. In practice though, the later option is only recommended in extreme conditions. The models considered here have thus focused more on ground delay and rerouting, although they can be extended to include other options.

Generally, an AFP is put into effect in a congested area of airspace that has its capacity reduced for some period of time due to severe weather. How long this weather will last is generally not known. Decisions must thus be made at the start of this AFP, which take into account the possible changes in weather and how these changes impact the AFP. Although the notion of a 'primary route', 'secondary route' and 'hybrid routes' are not currently part of the AFP conceptual framework, they represent a basic framework our problem and are consistent with previous work (Ganji et al., 2009), (Nilim, El Gahoui and Duong, 2003), (Nilim et al., 2001), (Nilim and El Gahoui, 2004).

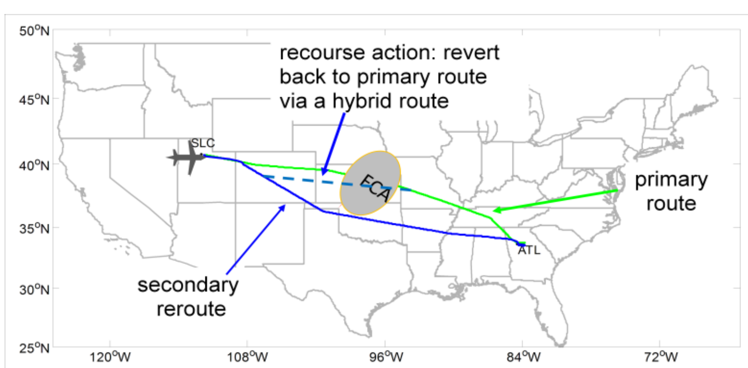

Figure 4.1: The Impact of an FCA on a Single Flight

Figure 4.1 gives an example of the options available to a single flight whose scheduled route goes through an FCA. Initially, the flight has two options: wait on the ground for some period of time for the ability to fly through the FCA, or depart immediately on a (longer) secondary route around the FCA. Each of these decisions has a possible recourse action available to it. If the flight is initially scheduled to its primary route with some ground delay, then in the event that the weather clears earlier than anticipated the possibility exists to reduce the amount of ground delay given to this flight. If the flight has already departed on its secondary route, then there is a possibility of giving it a shorter route through the FCA via a hybrid route. This option is indicated by the dashed blue line in Figure 4.1. Although Figure 4.1 only shows the options for a single flight, similar possibilities exist for every flight whose primary route is interrupted by the FCA.

An initial solution that is too aggressive would send more flights to the flow constrained area than the capacity can handle, in expectation that the weather will clear before these flights reach the area. This can result in the possibility of large amount of airborne delay or rerouting if the weather does not clear early. At the other extreme is rerouting more flights than necessary around the flow constrained area, in expectation that the weather will stay severe. If the weather does clear earlier than anticipated though, these flights will have already started on a path around the FCA and will either be forced to stay on this longer route, or to take a hybrid route through the FCA that is still longer than the flight's preferred initial route.

In between these two extremes lies an initial allocation that sends some flights on their preferred (primary) routes, other flights on their secondary routes and seeks to minimize the total expected delay of all flights involved. In order to compute the total expected delay, a probability distribution of how the weather is expected to behave over the FCA is necessary. There are many changes the weather can make over this time period, and attempting to model all such changes would cause the problem to grow to a size that would make the LP (and thus the IP as well) too time consuming to solve. Instead, as in Chapter 2, this formulation will focus on a simplified forecast of the weather where the assumption is that the weather will clear at some time in the future. What is uncertain then is when the weather will clear, allowing capacity of the FCA to go back to its normal capacity.

The problem inputs are:

- A set of flights, *Flights*, and data about these flights.
- For each flight $k \in$ *Flights,* the following is provided:
    - the distance from the FCA, *Enr*(*k*),
    - the scheduled departure, *Dep*(*k*),
    - the length of time required to travel along its secondary route, c_k^s,
    - the latest acceptable FCA slot, *last*(*k*).
    - a set of possible hybrid routes which would arrive at the FCA at some arrival slot *j*. The information provided for each flight *k*, slot *j* is:
        - the time at which the flight must deviate its secondary route in order to travel this hybrid route $t_{k,j}^d$,

- the savings offered by the flight k travelling a hybrid route which travels through the FCA at slot j, $sv_{k,j}^{h}$.

- The maximum duration of the AFP
- The reduced capacity of the FCA, $cap_1(i)$ for each initial slot i.
- There are T possible AFP endings (cancellations). Each cancellation $t = 1, \ldots, T$, has an associated time, $\tau(t)$ and a probability p_t. The AFP end time $\tau(t)$ will be referred to as scenario t.
- The nominal capacity of the airport, $cap_2(j,t)$ for each slot j in scenario t. (We assume that for each slot j in each scenario t, $cap_2(j, t) \geq cap_1(j)$.)

Slot i	$time(i)$	$cap_1(i)$	$cap_2(j, t)$
1	6:00	1	1
2	6:01	0	0
3	6:02	1	1
4	6:03	0	0
5	6:04	1	1
			AFP Cancelled at 6:05
6	6:05	0	**2**
7	6:06	1	**2**
8	6:07	0	**2**
9	6:08	1	**2**

Table 4.1: An Example of Stage One and Stage Two Capacities

The input to this problem is similar in nature to the problem described in Chapter 2. One major difference is the addition of secondary and hybrid routes for each flight, each with their associated costs. Table 4.1 illustrates sample cap_1 and

cap_2 vectors. Slots 6, 7, 8, and 9 each revert to their nominal capacity (of 2) starting at the storm clearance time $\tau(t)$. This extra capacity can be used by reducing the ground delay of certain flights (as is done in Chapter 2), or by rerouting a flight from its secondary route to a hybrid route through the FCA.

4.2 The Ganji Model

This chapter builds on the work of Ganji et al. who formulated a two stage stochastic IP of this problem. In their model, stage one assigns flights either to their primary route with some amount of ground delay, or on a secondary route with no ground delay. In order to accomplish this, the binary variables $x_{k,i}^{p}$ are introduced, where *k* is a flight and *i* is an FCA arrival slot, which is assumed to be the primary route of each flight. The variable $x_{k,i}^{p}$ is one if the flight *k* is initially assigned to its primary route which would go through the FCA at time *i*. The model assumes constant flight speed so a delay in the arrival time at the FCA also indicates a delay in the departure of the flight.

There is also a second class of variables in stage one of this model, x_{k}^{s}. These are the variables representing a flight being initially assigned to its secondary route. Because of the assumption that the secondary routes travel around the weather impacted area, i.e. they do not require an FCA slot, flights travelling along their secondary route would depart immediately and experience no ground delay. Obviously, a flight cannot depart on both its primary and secondary route. Also, the capacity of FCA arrival slots must be enforced. Stage one of his formulation can thus be shown to be:

$$\sum_{i \in Slots} x_{k,i}^{p} + x_{k}^{s} = 1 \text{ for each flight } k \tag{4.1}$$

$$\sum_{k \in Flights} x_{k,i}^{p} \leq cap_1(i) \text{ for each FCA arrival slot } i \tag{4.2}$$

$$x_{k,i}^{p}, x_{k}^{s} \in \{0,1\}$$

Constraint set (4.1) says that each flight is assigned to exactly one route, either a primary route with some ground delay, or a secondary route. Constraint set (4.2) says that the number of flights that are initially assigned to an FCA slot i cannot exceed the stage one capacity of that slot.

In stage two of this formulation there is a state corresponding to each possible realization of the random variable that gives the time at which severe weather over the impacted area may clear. Given that the weather is clear in each scenario, the capacity of the FCA is increased to a nominal capacity. This is represented in each scenario t by a capacity on each FCA arrival slot j, defined by $cap_2(j,t)$. In order to take advantage of this increase in capacity, one would like to adjust the ground delay and secondary routes given to flights. In each scenario though, we first need to account for the changes flights have taken over time. Some flights will have already departed on their primary routes and will only be able to continue on their primary route. The departure time of some flights will not have come yet, and these flights will be able to depart on a primary route when their departure time comes. Some flights will be serving ground delay. These flights will also be able to depart immediately. Finally, some flights will have already departed on their secondary route. These flights will thus be unable to depart on any primary route. However, there will be a set of hybrid routes for each flight in each scenario that these flights may or may not be eligible for. Each of these options is presented in the constraints of stage two of this formulation.

There are three types of stage two variables present in this formulation. The first type $y^p_{k,j,t}$ is a binary variable which is one if the flight *k* is reassigned to its primary slot *j* through the FCA in scenario *t*. The second type, $y^h_{k,j,t}$ is a binary variable which is one if the flight *k* is reassigned to a hybrid route which enters the FCA at slot *j*. The third type, $y^s_{k,t}$ is a binary variable which is one if the flight *k* remains on its secondary route in scenario *t*.

The stage two constraints of this model can be presented in three sets – those dealing with only the stage two primary routes, those dealing with the stage two primary and hybrid routes, and those dealing with the stage two primary, secondary and hybrid routes. First consider those constraints dealing only with the stage two primary routes:

$$\begin{aligned} &y^p_{k,j,t} = x^p_{k,j} \text{ for all flights } k, \text{ scenarios } t, \\ &\text{and } j \in \left\{1 \ldots \max\left\{\tau(t), Dep_k\right\} + Enr_k\right\} \end{aligned} \tag{4.3}$$

Dep_k and Enr_k represent the published departure and en route times for the flight *k*. These constraints restrict flight from be reassigned to a slot that is earlier its scheduled FCA arrival time, $Dep_k + Enr_k$. It also says that a flight cannot be reassigned to a slot in scenario *t* that is before the time it would take for it to depart at time $\tau(t)$ and travel to the FCA, unless it has already departed.

$$\sum_{j \in Slots_2} y^{p}_{k,j,t} + x^{s}_{k} \leq 1 \text{ for all scenarios } t, \tag{4.4}$$
$$\text{and all flights } k \text{ such that } Dep_k < \tau(t)$$

If a flight has already departed on its secondary route, then constraint set (4.4) says that this flight cannot be rescheduled to any primary route through the FCA.

$$y^{p}_{k,j,t} \leq \sum_{i \geq j} x^{p}_{k,i} + x^{s}_{k} \quad \text{for all flights } k\text{, scenarios } t \text{ and } j \in Slots_2 \tag{4.5}$$

This constraint set prevents a flight from being penalized when the weather clears by receiving a later primary slot than it was initially assigned in stage one. The secondary routes for a flight also appear in this constraint set because it is possible for a flight to be reassigned to a primary route if its secondary route has not yet departed.

Next, consider the constraints that deal with the stage two primary routes and hybrid routes.

$$y^{h}_{k,j,t} = 0 \quad \text{for all flights } k\text{, scenarios } t \text{ and } j \text{ s.t. } t^{d}_{k,j} \leq \tau(t) \tag{4.6}$$

The constants $t^{d}_{k,j}$ represent the time at which the flight k must depart its secondary route in order to arrive at the FCA slot j through its hybrid route. Because this model does not allow for pre-emptive hedging, this constraint set says that a flight cannot depart its secondary route for a hybrid route if the weather has not yet cleared.

$$y^{h}_{k,j,t} = 0 \quad \text{for all flights } k\text{, scenarios } t\text{, and } j > last(k) \tag{4.7}$$

This constraint set prevents a flight from being assigned to a FCA slot through its hybrid route that is later than it is willing to accept, *last*(*k*).

$y^h_{k,Dep_k+Enr_k,t} = 0$ for all k in flights and t in scenarios (4.8)

This constraint prevents flights from being initially assigned to their secondary route, but immediately rerouting for a hybrid route, essentially departing on the primary route.

$\sum_{k \in Flights} y^p_{k,j,t} + \sum_{k \in Flights} y^h_{k,j,t} \leq cap_2(j)$ for all k in flights, t in scenarios (4.9)

This enforces the capacity constraints for each FCA arrival slot in each scenario of stage two.

Finally, consider the constraints that deal with the stage two primary, hybrid, and secondary routes.

$y^h_{k,j,t} + y^s_{k,t} \leq x^s_k$ for all k in flights, t in scenarios, and j in $Slots_2$ (4.10)

This says that no flight can be reassigned to a secondary or hybrid route in any scenario unless it was initially assigned to its secondary route.

$\sum_{j \in Slots_2} y^p_{k,j,t} + \sum_{j \in Slots_2} y^h_{k,j,t} + y^s_k = 1$ for all k in flights, t in scenarios (4.11)

This says that every flight is reassigned to exactly one of its primary, secondary, or hybrid route options in each scenario of stage two.

The objective function for this model assigns a stage one cost $c^p_{k,i}$ to the variable $x^p_{k,i}$, which corresponds to the amount of ground delay the flight k receives by being assigned to slot i. There is also a stage one cost c^s_k for the variable x^s_k which corresponds to the amount of delay a flight receives by travelling on its secondary route instead of its non-delayed primary route. The stage one cost of this model is the sum of these costs over all flights and stage one slots.

The stage two costs for this model represent the expected savings from the reassignments. This is done by introducing two new variables, z_t^3 and z_t^4, which are respectively defined to be the amount of delay that is saved by flights that are reassigned to primary routes and hybrid routes in the scenario t. These variables are each multiplied p_t, the probability of the scenario t occurring. This objective function is modeled as follows:

$$\min Z = z_1 + z_2 - \sum_{t \in Scenarios} p_t \left(z_t^3 + z_t^4 \right)$$

where

$$z_1 = \sum_{k \in Flights} \sum_{i \in Slots_1} c_{k,i}^p x_{k,i}^p$$

$$z_2 = \sum_{k \in Flights} c_k^s x_k^s$$

$$z_t^3 = z_1 - \sum_{k \in Flights} \sum_{j \in Slots_2} c_{k,j}^p y_{k,j,t}^p + \sum_{k \in Flights} \sum_{j \in Slots_2} c_k^s s_{k,j,t}^p$$

$$z_t^4 = \sum_{k \in Flights} \sum_{j \in Slots_2} sv_{k,j}^h y_{k,j,t}^h$$

4.3 Strengthening the Ganji Model

There are many ways to represent a discrete set of integers as the integer solutions to a system of linear inequalities. These different formulations are not always equivalent in strength. Although each of these formulations will accurately represent the discrete set in question, some may also have the property that its extreme points are the members of X. This implies that if we wish to solve an IP and X is our set of feasible solutions, then such a formulation will be solved by its LP-relaxation. The strength of an IP formulation refers to "how close" the polyhedron for the constraint matrix is to the convex hull of integer solutions. Strong formulations play an important role in integer programming, as these formulations generally are able to return optimal solutions quickly.

The Ganji model has been shown to solve the problem of stochastic weather affecting en route traffic. However, generating these solutions can take large amounts of time. We now develop a strengthening of the Ganji model and show that this leads to much faster solution times.

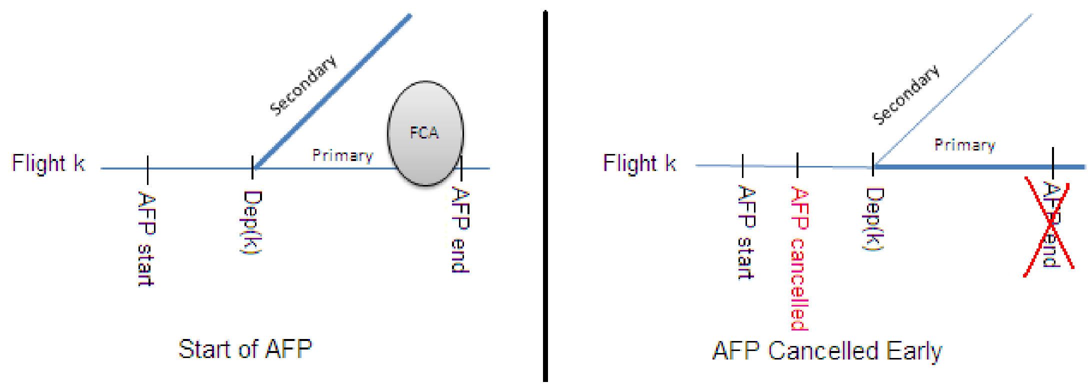

Figure 4.2: The AFP is cancelled before the flight k can depart its secondary route.

One situation the Ganji model must be able to represent is given by Figure 4.2. Here, a flight k is initially assigned to its secondary route, but the cancellation time of scenario t is before flight k's departure time. This allows the possibility of reassigning flight k to its primary route. Of course, this could only be done if there is sufficient capacity available. In order to handle the costs associated such a situation, the variables $s_{k,j,t}^{p}$ were introduced to the Ganji model. These variables are one if the flight k was originally assigned to its secondary route and is reassigned to its slot j on its primary route in scenario t. The authors ensure that this happens by defining the variables as follows, $s_{k,j,t}^{p} = \min\left\{x_k^s, y_{k,j,t}^{p}\right\}$. The associated linear constraints that were implemented in the model are:

$$\sum_{j \in Slots_2} s_{k,j,t}^{p} \le x_k^s \tag{4.12}$$

$$s_{k,j,t}^{p} \le y_{k,j,t}^{p} \tag{4.13}$$

These variables were supposed to only have an impact on the objective function, but they can directly affect the solution (to the LP-relaxation) by introducing a number of non-integer extreme points to the formulation. This happens because x_k^s and $y_{k,j,t}^{p}$ both appear in the constraint $\sum_{j \in Slots_2} y_{k,j,t}^{p} + x_k^s \le 1$ and so the following is now a sub-matrix of this formulation:

$\begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix}$. This matrix has a determinant 2. These submatrices introduce many non-integer extreme points to the LP-relaxation of the Ganji model. This means that the $s_{k,j,t}^{p}$ variables, which were supposed to only affect the objective function, in fact weaken the formulation. Further, the LP-relaxation often returns optimal solution with the variables corresponding to such a submatrix having values of 0.5. Thus, eliminating these submatrices from the formulation could lead to more efficient run times.

Suppose we define a new set of variables to serve the same purpose: the binary variable $\hat{s}_{k,t}$ is one if the flight k is assigned to its secondary route in stage one and reassigned to its primary route in scenario t of stage two. Consider the following equation:

$$\hat{s}_{k,t} = 0.5\left(x_k^s - \sum_{i \in Slots_1} x_{k,i}^p + \sum_{j \in Slots_2} y_{k,j,t}^p - \sum_{j \in Slots_2} y_{k,j,t}^h - y_k^s \right) \tag{4.14}$$

Proposition 4.1: For a flight k and a scenario t, the variable $\hat{s}_{k,t} = \sum_{j \in Slots_2} s_{k,j,t}^p$ for all integer solutions to the Ganji formulation.

Proof:

Because of the constraint $x_k^s + \sum_{i \in Slots_1} x_{k,i}^p = 1$, all integer solutions will have for each flight k, either $x_k^s = 1$ or $\sum_{i \in Slots_1} x_{k,i}^p = 1$. Likewise because of the constraint $\sum_{j \in Slots_2} y_{k,j,t}^p + \sum_{j \in Slots_2} y_{k,j,t}^h + y_{k,t}^s = 1$, all integer solutions will have that either $\sum_{j \in Slots_2} y_{k,j,t}^p = 1$ or $\sum_{j \in Slots_2} y_{k,j,t}^h + y_{k,t}^s = 1$. There are then 4 cases to consider:

Case 1: $x_k^s = 1$ and $\sum_{j \in Slots_2} y_{k,j,t}^p = 1$.

In this case, (4.18) evaluates to $\hat{s}_{k,t} = 1$. Since this is the exact case where the flight k starts on its secondary route, $x_k^s = 1$, and is rerouted to a primary route, $\sum_{j \in Slots_2} y_{k,j,t}^p = 1$, any integer feasible solution to the Ganji formulation in would also have $s_{k,j,t}^p = \min\left\{x_k^s, y_{k,j,t}^p\right\} = 1$ since $\sum_{j \in Slots_2} y_{k,j,t}^p = 1$ implies that $y_{k,j,t}^p = 1$ for some $j \in Slots_2$ in the integer solution.

Case 2: $x_k^s = 1$ and $\sum_{j \in Slots_2} y_{k,j,t}^h + y_{k,t}^s = 1$.

In this case, the new constraint evaluates to $\hat{s}_{k,t} = 0$. This is a case where the flight k starts on its secondary route and is either rerouted to a hybrid route or stays on its secondary route. This means that any integer feasible solution to the Ganji formulation in would have $\sum_{j \in Slots_2} y_{k,j,t}^p = 0$. This implies that $s_{k,j,t}^p = \min\left\{x_k^s, y_{k,j,t}^p\right\} = 0$ since $\sum_{j \in Slots_2} y_{k,j,t}^p = 0$ implies that $y_{k,j,t}^p = 0$ for all $j \in Slots_2$ in the integer solution.

Case 3: $\sum_{i \in Slots_1} x_{k,i}^p = 1$ and $\sum_{j \in Slots_2} y_{k,j,t}^h + y_{k,t}^s = 1$.

Since this is a case where the flight starts its primary route and is rerouted to a hybrid route or secondary route (something outlawed by the constraint $y_{k,j,t}^h + y_{k,t}^s \le x_k^s$), this is not a feasible situation for the Ganji formulation.

Case 4: $\sum_{i \in Slots_1} x_{k,i}^p = 1$ and $\sum_{j \in Slots_2} y_{k,j,t}^p = 1$.

In this case, the new constraint evaluates to $\hat{s}_{k,t} = 0$. Since this is the case where the flight k starts on a primary route and is rerouted to another primary route, any integer feasible solution to the Ganji formulation would have $s_{k,j,t}^p = \min\left\{x_k^s, y_{k,j,t}^p\right\} = 0$ since $x_k^s = 0$.

Because in all cases, the variables were equal, it follows that the claim is true in general.

Q.E.D.

The noticeable difference between the $\hat{s}_{k,t}$ variables defined here and the $s_{k,j,t}^p$ variables defined in the Ganji formulation is that the new variables have one less subscript. This does not present a problem because the only places this variable occurs in the formulation are in the objective function and the two constraints presented earlier. The objective function coefficient for these variables is $-c_k^s$, the secondary cost of the flight k, which does not depend on the FCA reroute slot j.

Further, using these new variables, the submatrix of determinant 2 induced by constraints (4.12) and (4.13) can be removed from the formulation and replaced with an equality constraint. Also, because the only occurrence of the $s_{k,j,t}^p$ is in the objective function for the Ganji model, this substitution of the $s_{k,j,t}^p$ with the $\hat{s}_{k,t}$ variables effectively eliminates the need for either of them in the IP. That is, we initially include $\sum_{t \in Scenarios} \sum_{k \in Flights} -c_k^s \hat{s}_{k,t}^p$ in the objective function, but use equation (4.14) to eliminate $\hat{s}_{k,t}^p$ by adjusting the coefficients of other variables. This builds a stronger formulation with fewer variables and no change in the objective function value for the integer feasible solutions.

This new formulation, though, still has submatrices of determinant 2 which weaken the LP-relaxation of this IP. The following constraint set represents one such submatrix:

$$y_{k,j_1,t}^p \le \sum_{i \ge j} x_{k,i}^p + x_k^s$$

$$y_{k,j_2,t}^p \le \sum_{i \ge j} x_{k,i}^p + x_k^s$$

$$\sum_{j \in Slots_2} y_{k,j,t}^p + \sum_{j \in Slots_2} y_{k,j,t}^h + y_k^s = 1$$

These constraints contain the following submatrix:

$$\begin{pmatrix} x_{k,i}^p & y_{k,j_1,t}^p & y_{k,j_2,t}^p \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix}$$

This matrix has a determinant of 2, and allows for non-integer extreme points to the Ganji formulation. There are also a large number of variables present in this formulation due to the assignment problem set up in stage two. What follows is a new stochastic IP formulation for this problem that eliminates some of these submatrices and is more compact.

4.4 New Formulation

Before describing a new formulation for the stochastic AFP problem, it is also important to briefly describe the formulation presented in Chapter 2 for the stochastic dynamic GDP (SDGDP). The AFP problem can be seen as a similar, but more complex version of this problem. In SDGDP, one is preparing to assign flights to landing slots at an airport that is about to experience a GDP due to severe weather. Similar to the AFP, one is uncertain when this severe weather will clear, and thus seeks an initial assignment that minimizes the total expected delay over the given possible end times.

Chapter 2 models this problem as a two stage stochastic integer program. Stage one gives flights their initial assignments, while stage two models the possible cancellation times as scenarios and reallocates flights to newly available airport landing slots in each scenario. In stage one, $x_{k,i}$ is the binary variable which is one if the flight k is initially assigned to the arrival slot i. Then the following constraints model stage one:

$$\sum_{\substack{i \in Slots_1 \\ time(i) \geq arr(k)}} x_{k,i} = 1 \quad \text{for each flight } k \tag{4.15}$$

$$\sum_{\substack{k \in Flights \\ time(i) \geq arr(k)}} x_{k,i} \leq cap_1(i) \quad \text{for each arrival slot } i \tag{4.16}$$

$$x_{k,i} \in \{0,1\}$$

Constraints (4.15) say that each flight, k, is assigned to exactly one slot, while constraints (4.16) say that the number of flights assigned to a slot, i, cannot exceed its stage one capacity, $cap_1(i)$.

Next, consider stage two of this formulation. The Queue-Based Formulation presented in Chapter 2 sets up each scenario of stage two as a queue amongst the airport arrival slots. Each arc of stage one has its own entry time into the queue in each scenario, defined by the function, $earliest(k,i,t)$, which determines the earliest slot that the arc (k,i) can be reallocated to in scenario t of stage two. This function can be determined for each arc and each scenario in preprocessing. The airport arrival slots in stage two will have an equal or higher capacity than in stage one, but there will indeed be a capacity on each slot. Because any number of arcs can have the same earliest reallocation for some scenario, the purpose of the queue is to enforce the stage two capacity constraints at each arrival slot by allowing for flights to be reallocated to a later slot at an associated cost. This allows for the construction of a feasible stage two solution, (u,z), where $u_{j,t}$ is an integer variable whose value represents the number of flights that are reallocated to the FCA slot j in scenario t, and $z_{j,t}$ is an integer variable representing the number of flights that are sent from slot j to slot $j+1$ in scenario t. The stage two constraints are as follows:

$$\sum_{(k,i):earliest(k,i,t)=time(j,t)} x_{k,i} + z_{j-1,t} - z_{j,t} - u_{j,t} = 0 \quad \text{for all } j, t \tag{4.17}$$

$$u_{j,t} \leq cap_2(j,t) \quad \text{for all } j, t \tag{4.18}$$

Constraint set (4.17) sets up a queue in each scenario of stage two. Each of these constraints is essentially a flow conservation constraint, where the nodes are stage two slots. The flow into the stage two networks are the stage one variables. They enter at the stage two queue in each scenario at the arrival slot defined by the function $earliest(k,i,t)$. The $z_{j,t}$ variables represent the flow between the slots and the $u_{j,t}$ variables represent the flow out of the nodes. There will need to be three different variants of this constraint set depending on if j is the first, last or another slot in the scenario t. Constraint set (4.18) limits the number of flights that can be reallocated to arrival slot j in scenario t, not to exceed the stage two capacity, $cap_2(j, t)$.

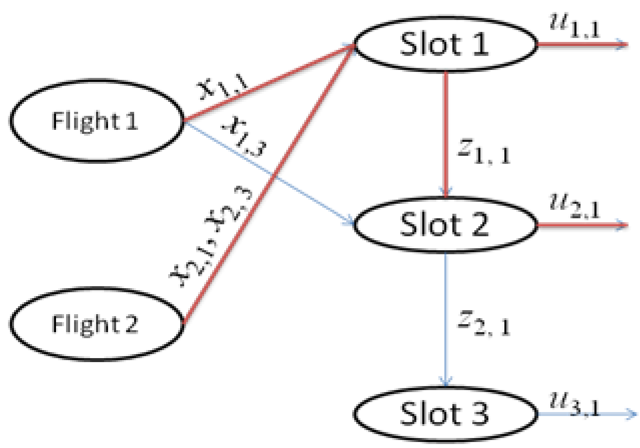

Figure 4.3: Stage Two Queue Example on Two Flights

Figure 4.2 gives an example of how constraint set (4.17) would operate on two flights with the same earliest, and three slots with a stage two capacity of 1 in this scenario. If the stage one solution is $x_{1,1}$ and $x_{2,3}$ both set equal to one, then because $earliest(1, 1, 1) = earliest(2, 3, 1) = 1$, both Flight 1 and Flight 2 enter the queue at slot 1. Slot 1, however, only has capacity 1, so both flights cannot be assigned to this single slot. Instead one flight is assigned to slot 1 and one is passed to the next slot in the queue, slot 2. Since slot 2 has a capacity 1, a flight can be assigned here and the queue is now empty, leaving slot 3 unused.

The cost metric which the stochastic GDP seeks to minimize is total expected delay. This can be measured by $\sum_{k \in Flights} \sum_{j \in Slots_2} \sum_{t \in Scenarios} \left(time(j,t) - arr(k) \right) u_{j,t}$, where *time*(*j*, *t*) is the time that the slot *j* begins in scenario *t*.

Here, a new formulation will be presented that reduces the size of the IP while also eliminating some of the odd cycles present in the Ganji model. This formulation will take the same input as the Ganji model, make the same assumptions and will combine the model presented in Chapter 2 with the Ganji formulation.

Similar to the Ganji model, we define the variables $x_{k,i}^p$ to be binary variables which is one if flight *k* is initially assigned to the arrival slot *i* on its primary route, and x_k^s is the binary variable which is one if the flight *k* is initially assigned to its secondary route. Then the following two constraint sets model the stage one restrictions. These constraints are very similar to the model proposed by Odoni (Odoni, 1987) as well as the stage one constraints in the Ganji model and in Chapter 2.

$$\sum_{\substack{i \in Slots_1 \\ time(i) \geq Dep(k) + En(k)}} x_{k,i}^p + x_k^s = 1 \quad \text{for each flight } k \tag{4.19}$$

$$\sum_{\substack{k \in Flights \\ Dep(k) + Enr(k) \leq time(i)}} x_{k,i}^p \leq cap_1(i) \quad \text{for each arrival slot } i \tag{4.20}$$

$x_{k,i}^{p}, x_{k}^{s} \in \{0,1\}$

Each flight has a scheduled time at which it is due in the impacted FCA, *Dep(k)+Enr(k)*, and the first constraint set ensures that each flight is either assigned to its secondary route or some arrival slot after its scheduled arrival time. Each slot i in the FCA has an initial capacity, $cap_1(i)$, the number of flights the FCA can handle during the reduced capacity during time interval i. The second constraint set represents the limit on slot capacity.

Stage two of the model can be viewed as a combination of the stage two of the Ganji model and the Queue Model presented in Chapter 2. There is a scenario, t, for each possible weather clearance time. Similar to the Ganji model, flights could have taken off on a primary route or a secondary route. Because the flights that are attempting to travel through the FCA are being handled in a manner similar to a GDP, constraints similar to those in stage two of the queue model will be simulated to handle these flights.

The following constraint set sets up a queue in each scenario of stage two amongst the primary FCA slots available in that scenario. The function $earliest(k,i,t)$ maps the allocation (k, i) from stage one to the earliest FCA arrival slot that it can be reallocated to in scenario t. If $earliest(k,i,t) = time(j,t)$, then the variable $x_{k,i}$ can enter the scenario t queue at slot j, depending on whether its value is nonzero or not. The variable $z_{j,t}$ is the amount that is passed from slot $j-1$ to slot j in scenario t. The following constraint immediately follows:

$$\sum_{(k,i,t)|earliest(k,i,t)=time(j,t)} x_{k,i} + \sum_{k|Dep(k)+Enr(k)=time(j,t)} s_{k,t}^{p} + z_{j-1,t} - z_{j,t} - u_{j,t} = 0 \tag{4.21}$$

for each arrival slot j and each scenario t

This says that the flights such that $earliest(k,i,t)$ = $time(j,t)$ will enter the queue at slot j. flights that did not leave the queue at slot $j-1$ are sent to slot j through the variable $z_{j-1,t}$. Those flights that depart the queue at slot j do so via the variable $u_{j,t}$. Similar to Chapter 2, there will need to be three different versions of this constraint, depending on whether the slot j is the first slot (in which case, there is not $z_{j-1,t}$ variable), the last slot (in which case, there is no $z_{j,t}$ variable), or an in-between slot.

The difference between this constraint and the similar version in the queue model presented in Chapter 2 is the addition of the $s_{k,t}^{p}$ variables. Allocations that begin on their secondary route and are rerouted to their primary route are also allowed to enter the queue at their originally expected arrival time. This immediately leads to a question of how we will ensure not to have similar odd cycles as the Ganji model. This can be done with the following constraint set:

$$\sum_{j \in Slots_2} y_{k,j,t}^{h} = 0 \quad \text{for all } t \text{ with } \tau(t) \le Dep(k) \tag{4.22}$$

$$s_{k,t}^{p} = 0 \quad \text{for all t with } \tau(t) > Dep(k) \tag{4.23}$$

This will have the desired effect because in scenarios before the departure time of the flight $\tau(t) \le Dep(k)$ the flight has not yet departed on its secondary route which means that it cannot depart the secondary route for a hybrid route. Thus, if $x_k^s = 1$, then $s_{k,t}^{p} + y_{k,t}^{s} = 1$ in such a scenario. Likewise, if $x_k^s = 0$, then $s_{k,t}^{p} = 0$ in this scenario. In scenarios that are after the departure time of the flight $\tau(t) > Dep(k)$ the flight will have already departed on its secondary route and will thus be unable to depart on its primary route. Hence $s_{k,t}^{p} = 0$. The question of whether or not a flight has yet departed on its secondary route does not depend on any information other than the input data to the problem, namely the scheduled flight departure times and the possible weather clearance times.

Based on the arrival time of each flight, *Dep*(*k*)+*Enr*(*k*), the primary route length of each flight, *Enr*(*k*), and the set of AFP cancellation times the function $earliest(k,i,t)$, for each stage one allocation (*k, i*) and each scenario *t*, can be determined as a pre-processing step. The definition of this function will be similar to its definitions in Chapters 2 and 3.

Some of the arcs in stage one will not be eligible for the queue because they will have already departed on their secondary route. In each scenario, *t*, these flights will have the option of rerouting through the FCA on a hybrid route or continuing on their secondary route. To model the options available to these flights in stage two, the variables $y_{k,j,t}^{h}$ and $y_{k,t}^{s}$ are introduced, indicating the hybrid and secondary options, respectively for the flight *k* in scenario *t*.

These variables will be handled in a manner similar to the Ganji model. It must first be ensured that no flight departs on an ineligible hybrid route. This can be accomplished by only defining these variables for the scenarios *t* which are equal to or after the hybrid diversion time for the flight *k* and FCA arrival slot *j*.

$$y^{h}_{k,j,t} = 0 \quad \text{for all flights } k, \text{ scenarios t} \\ \text{and } j \text{ s.t. } t^{d}_{k,j} \le \tau(t) \tag{4.24}$$

Next, because these are recourse actions a flight which has already departed on its secondary route can take once the weather clears, it needs to be ensured that the only time one of these is used is when the flight was originally assigned to its secondary route:

$$\sum_{j \in Slots_2} y^{h}_{k,j,t} + y^{s}_{k,t} + s^{p}_{k,t} = x^{s}_{k} \qquad \text{for all flights } k \text{ and scenarios } t \tag{4.25}$$

This has the desired effect because the only recourse options available when a flight is initially assigned to its secondary route are the hybrid routes, the primary route or staying on the secondary route. The situation where the flight reroutes from its secondary route to its primary route in scenario *t* is handled by the variable $s^{p}_{k,t}$.

The capacities for the stage two FCA slots need to also be respected. There are two types of routes that can be allocated trough the FCA in a scenario of stage two: primary routes and hybrid routes. To ensure that no FCA arrival slot's capacity is violated the following constraint is enforced:

$$u_{j,t} + \sum_{k \in Flights} y^{h}_{k,j,t} \le cap_2(j) \text{ for each FCA} \\ \text{arrival slot } j \text{ and stage two scenario } t \tag{4.26}$$

It is sometimes better to allow the hybrid routes to use up an FCA slot that was previously reserved for a primary route. The purpose for this is that delaying a primary route adds a constant amount of time to the objective function, *time*(j+1,t) – *time*(j), whereas delaying the hybrid route means taking a later hybrid route or staying on the secondary route in that scenario. The constraint in the Ganji formulation, $y^p_{k,j,t} \le \sum_{i \ge j} x^p_{k,i} + x^s_k$ ensures that no flight assigned to a primary route is given an FCA slot in stage two that is later than the slot that it received in stage one. This inherently enforces some flights on secondary routes to take later hybrid routes or to remain on their secondary routes, which has an adverse effect on the objective function. The similar constraint set for this new model would be as follows:

$$\sum_{j=earliest(k,i,t)}^{time(i)} u_{j,t} \ge \sum_{\substack{k' \in Flights| \\ earliest(k',i,t)=earliest(k,i,t)}} x^p_{k',i} \qquad (4.27)$$

for each flight k, initial slot i, and scenario t

Constraint set (4.27) ensures that the number of flights that exit the queue between the queue entry and exit points (*earliest*(k, i, t) and i, respectively) for a given flight is at least the number of flights that have the same queue entry and exit points. This guarantees that each flight leaves the queue by the slot it was initially assigned, i. Secondary routes that are switched to their primary routes could also be assigned to these slots, but because these flights are entered into the queue, they receive no priority over originally scheduled primary routes. Thus, in post-processing, when the decision of which flight leaves the queue is made, the no-preemption rule can be applied and the flight with originally scheduled to its primary route can be chosen to leave the queue.

As stated above, (4.27) will create many additional constraints. However many constraints will be redundant and can eliminated.

Finally, we have $x^p_{k,i}$, x^s_k, $y^h_{k,j,t}$, $s^p_{k,t}$ and $y^s_{k,t}$ $\in \{0, 1\}$. $u_{j,t}$ and $z_{j,t} \geq 0$ and are integer. We call this the Queue based En Route (Q-EN).

4.5 Formulation Comparison

There are a number of different ways this new formulation can be compared to the Ganji model, but some key areas where they differ will be presented here. The first difference is in the size of the formulations. Below are two graphs showing how each formulation grows when given the same sample problems as input. To better contrast the differences, a logarithmic scale was used instead of a linear scale:

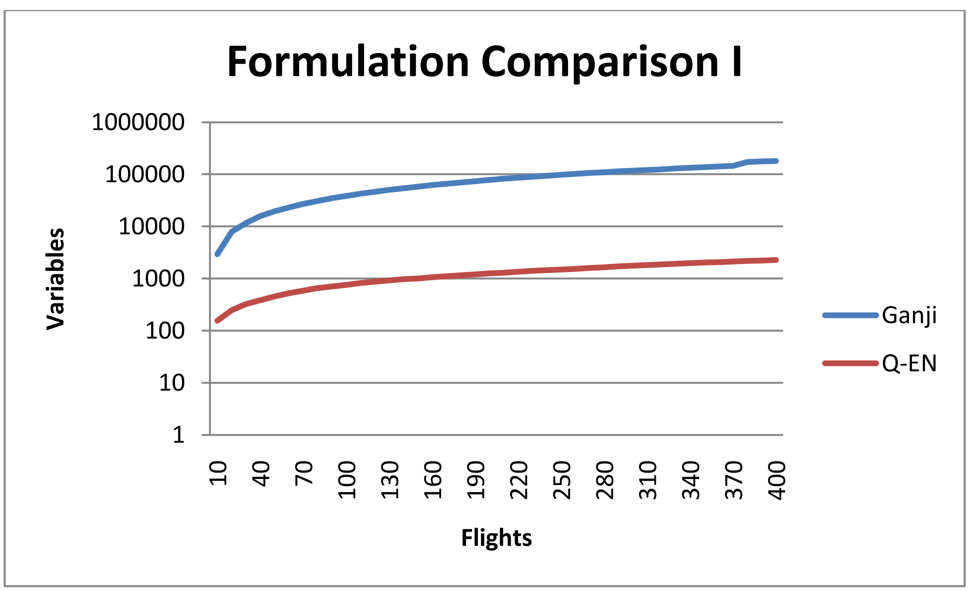

Figure 4.4: Variable Comparison of the Formulations of Q-EN and Ganji

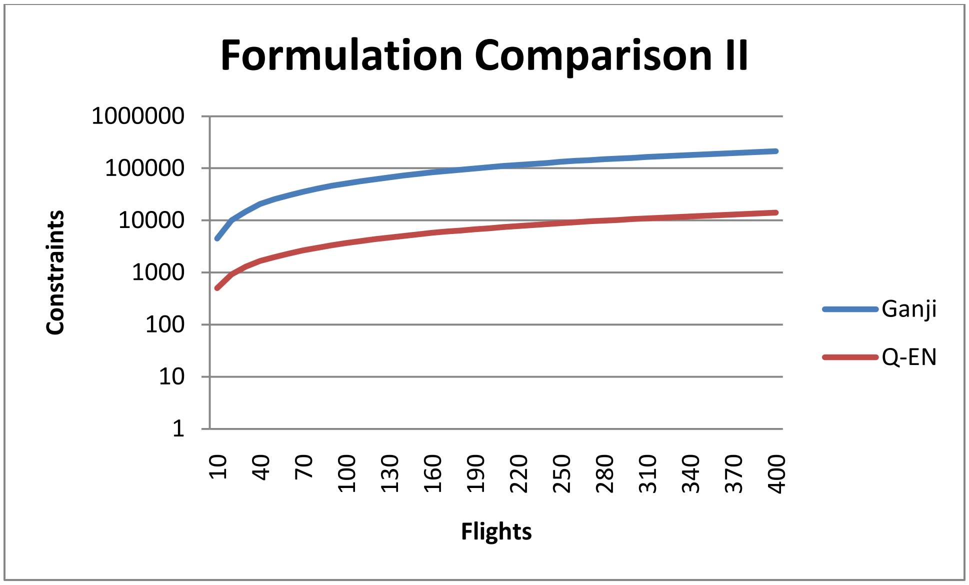

Figure 4.5: Constraint Comparison of the Formulations of Q-EN and Ganji

Next the different formulations were compared in execution time. The experiments were run on a PC with Two quad-core Xeon processors, 12GB RAM, and XpressMP 2008A. The following two graphs analyze the differences here:

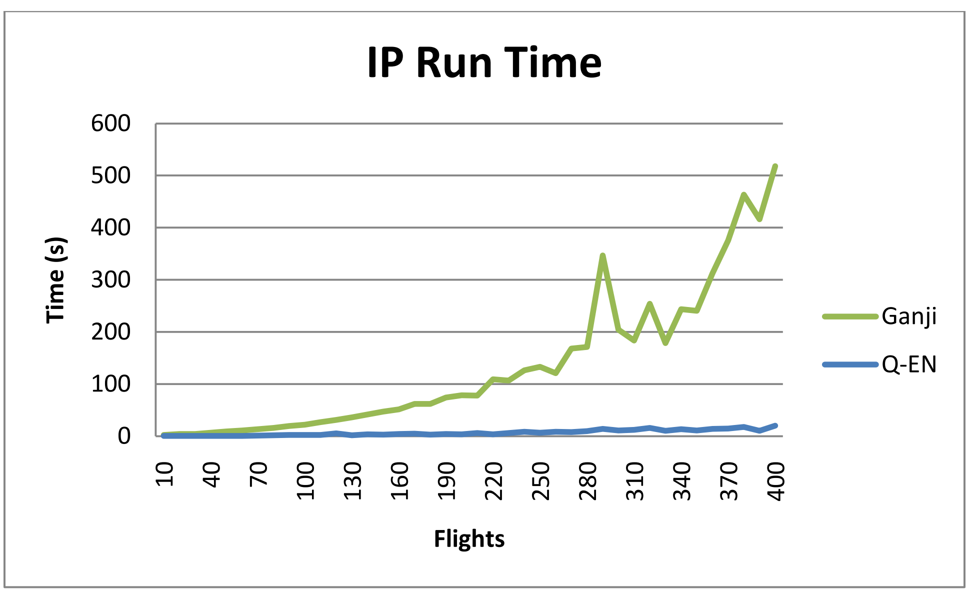

Figure 4.6: IP Run Time Comparison of the Q-EN and Ganji Formulations

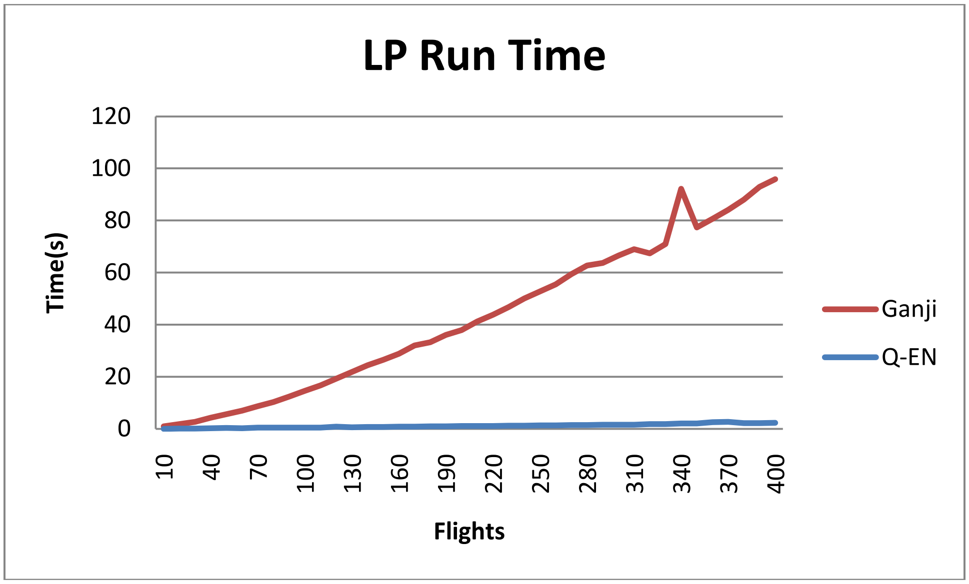

Figure 4.7: LP Run Time Comparison of the Q-EN and Ganji Formulations

Because the LP-relaxation of a minimization IP is minimizing over a larger set of values, the objective function value of the LP-relaxation provides a lower bound on the objective function value of the IP. How close this LP-relaxation is to the IP solution is a good indicator of the strength of the formulation. The following two graphs provide insight into this.

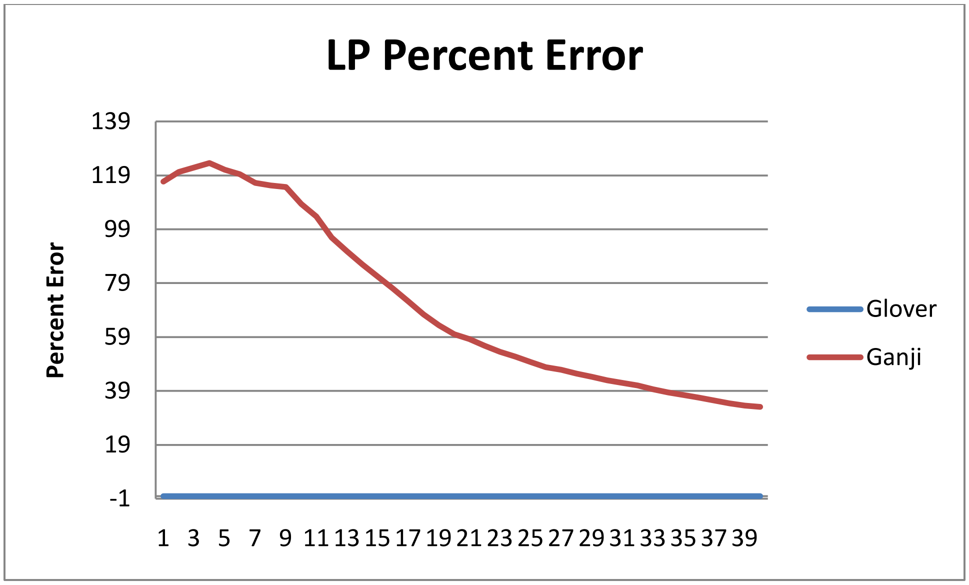

Figure 4.8: The Percentage of error in the LP-relaxations

From the above tables we can see that the new formulation smaller, faster, and returns an LP-relaxation, which is closer to the integer optimal solution.

When the results are compared with the Ganji model with the change in the $s_{k,t}^{p}$ variables, the run time for the Ganji model is greatly improved, as is the difference between the IP and LP solutions. What remains large in the improved Ganji model would be the number of variables and constraints which would make the model much more difficult to run on larger instances of airspace congestion.

Also, as noted by the instances where the LP-relaxation is not equal to the IP solution, there are still fractional extreme points in this new model. Many of these will come from instances where two nonadjacent arcs in stage one have the same earliest reallocation in stage two. It remains to be seen if this new model can be strengthened to guarantee that the LP-relaxation of the IP will always give an integer optimal solution.

Chapter 5 Conclusions and Future Work

Chapter 2 described three models to solve the stochastic dynamic GDP and shows that the LP-relaxation of these models solve the IP. This extends the work of Ball et al. (Ball, Hoffman and Mukherjee, 2010), who showed that the RBD Algorithm minimizes the expected delay of a ground delay program when the cancellation times are uncertain, and provides a basis for comparison to other problems that look at planning around uncertainty. The proof that the LP-relaxations of these formulations solve the IP utilizes Monge matrices. Although the Monge property was not used in the proof, a property which is closely related, lower-Monge, did apply.

The polyhedron for the stochastic dynamic GDP models were shown to be non-integer in general, and a class of valid inequalities were provided to improve the strength of the formulation. A question remains of how strong these cuts are. Also what, if any, are some of the other non-integer extreme points? What patterns in objective functions exist so as to make these non-integer extreme points optimal? These questions are important, not only for more understanding of the two-stage stochastic dynamic ground delay problem, but also as we seek to gain understanding of some of the formulations presented in Chapter 3.

Although several assumptions were made on the formulation of the stochastic GDP that apply to the ways GDPs are implemented in practice, a question does arise of how many similar problems could be formulated in a related manner. There is a large class of resource allocation problems which attempt to assign a resource whose capacity has been temporarily reduced with an uncertain time of capacity increase.

Because this general problem class matches some of the basic assumptions of the stochastic dynamic GDP, it is natural to seek to understand how many problems in this larger class of problems could be formulated by similar stochastic integer programs.

Chapter 3 extends the Queue-Based Formulation from Chapter 2 to a setting where equity/efficiency trade-offs are modeled. The formulations also give solutions that are comparable in both equity and efficiency to other rationing principles in the literature such as RBD and RBS. While all the formulations presented had their benefits, the formulation which included a constraint limiting maximum deviation from RBS, with a weighted objective means of minimizing total deviation from RBS and total expected delay was able to provide solutions that looked good under a number of different equity metrics while remaining efficient. Because these new formulations had ether a different objective function, or a limitation on the set slots to which a flight can be assigned, the lower-Monge results no longer holds. As a result, it is an open question of whether there exists an extension of the result in Chapter 2 to a larger class of problems which includes those presented in Chapter 3.

Finally, one of the formulations presented in Chapter 2 was used to strengthen the model presented by Ganji et al. (Ganji et al., 2009), which seeks to maximize throughput through a volume of the airspace system where a capacity-demand imbalance is expected usually due to adverse weather. The new formulation was both stronger and more compact than the Ganji model. These properties led to much improved computation time and the solution of larger problem instances.

Glossary

Airport Arrival Rates (AARs)

Airspace Flow Program (AFP)

Air Traffic Flow Management (ATFM)

Collaborative Decision Making (CDM)

Expected Time of Arrival (ETA)

Federal Aviation Administration's (FAA)

Flow Constrained Area (FCA)

Ground Delay Programs (GDPs)

Instrument Flight Rules (IFR)

Integer Programming (IP)

Multiple Airport Ground Holding Problem (MAGHP)

National Airspace System (NAS)

Planned AAR (PAAR)

Ration-By-Schedule (RBS)

Ration-by-Distance (RBD)

Single Airport Ground Holding Problem (SAGHP)

Stochastic Dynamic Ground Delay Problem (SDGDP)

Totally Unimodular (TU)

Traffic Flow Management Rerouting Problem (TFMRP)

Visual Flight Rules (VFR)

Bibliography

Ball, M.O., C.Barnhart, G.Nemhauser and A.Odoni (2007) 'Air Transportation: Irregular Operations and Control', in Barnhart, C. and G.Laporte (ed.) *Handbook of Operations Research and Management Science: Transportation*, Elsevier, Amsterdam.

Ball, M.O., R.Hoffman and A.Mukherjee (2010) 'Ration-By-Distance with Equity Guarantees: A New Approach to Ground Delay Program Planning and Control', *Transportation Science*, 44, 1-14.

Ball, M.O., R.Hoffman, A.Odoni and R.Rifkin (2003) 'A Stochastic Integer Program with Dual Network Structure and its Application to the Ground-Holding Problem', *Operations Research*, 51, 167-171.

Ball, M.O. and G.Lulli (2004) 'Ground Delay Programs: Optimizing over the Included Set Based on Distance', *Air Traffic Control Quarterly*, 12, 1-25.

Bein, W., P.Brucker, J.Park and P.Pathak (1995) 'A Monge Property for the d-dimensional Transportation Problem', *Discrete Applied Mathematics*, 58, 97-109.

Bertsimas, D. and S.Stock Patterson (1998) 'The Air Traffic Flow Management Problem with Enroute Capacities', *Operations Research*, 46, 406-422.

Bertsimas, D. and S.Stock Patterson (2000) 'The Traffic Flow Management Rerouting Problem in Air Traffic Control - A Dynamic Network Flow Approach', *Transportation Science*, 34, 239-255.

Bertsimas, D. and J.Tsitsiklis (1997) *Introduction to Linear Optimization*, Nashua, NH: Athena Scientific.

Birge, J. and F.Louveaux (1997) *Introduction to Stochastic Programming*, New York, NY: Springer.

Burkard, R.E., B.Klinz and R.Rudiger (1996) 'Perspectives of Monge Properties in Optimization', *Discrete Applied Mathematics*, 70, 95-161.

Chang, K., K.Howard, R.Oiesen, L.Shisler, M.Tantino and M.Wambsganss (2001) 'Enhancements to the FAA Ground-Delay Program Under Collaborative Decision Making', *Interfaces*, 31, 57-76.

FAA (2006) *AFP Concept*, [Online], Available: http://www.fly.faa.gov/What_s_New/AFP_Concept.pdf&pli=1.

FAA (2006) *CDM Web Site*, [Online], Available: http://cdm.fly.faa.gov/index.html.

Ganji, M., D.Lovell, M.O.Ball and A.Nguyen (2009) 'Resource Allocation in Flow-Constrained Areas with Stochastic Termination Times', *Transportation Research Board*, 90-99.

Garey, M.R. and D.S.Johnson (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York: W.H.Freeman and Company.

Halmos, P.R. and H.E.Vaughan (1950) '"The marriage problem"', *American Journal of Mathematics*, 72, 214-215.

Hoffman, A.J. (1963) 'On Simple Linear Programming Problems', in Klee, V.L. *Convexity, Proceedings of Symposia in Pure Mathematics, Vol. 7*, Providence, RI: American Mathematical Society.

Innis, T. and M.O.Ball (2004) 'Estimating One Parameter Airport Arrival Capacity Distributions for Air Traffic Flow Management', *Air Traffic Control Quarterly*, 12, 223-252.
Kotnyek, B. and O.Richetta (2006) 'Equitable Models for the Stochastic Ground-Holding Problem Under Collaborative Decision Making', *Transportation Science*, 40, 133-146.
Mitchell, J.S.B., V.Polishchuk and J.Krozel (2006) 'Airspace Throughput Analysis Considering Stochastic Weather', AIAA Guidance, Navigation, and Control Conference, Keystone, CO.
Mukherjee, A. and M.Hansen (2007) 'A Dynamic Stochastic Model for the Single Airport Ground Holding Problem', *Transportation Science*, 41, 444-456.
Nilim, A. and L.El Gahoui (2004) 'Algorithms for Air Traffic Flow Management under Stochastic Environments"', In Proceedings of the 2004 American Control Conference, Boston. Massachusetts, 3429-3434.
Nilim, A., L.El Gahoui and V.Duong (2003) 'Multi-Aircraft Routing and Traffic Flow Management under Uncertainty', In Proceedings of 5th USA/Europe Air Traffic Meeting R&D Seminar, Budapest, Hungary.
Nilim, A., L.El Gahoui, V.Duong and M.Hansen (2001) 'Trajectory-Based Air Traffic Management (TB-ATM) under Weather Uncertainty', In Proceedings of the 4th USA/Europe Air Traffic Meeting R&D Seminar, Santa Fe, New Mexico.
Odoni, A.R. (1987) 'The Flow Management Problem in Air Traffic Control', in Odoni, A.R., L.Bianco and G.Szego (ed.) *Flow Control of Congested Networks*, Berlin: Springer.
Richetta, O. and A.Odoni (1994) 'Dynamic Solution to the Ground-Holding Problem in Air Traffic Control', *Transportation Research. Part A: Policy and Practice*, 28, 167-185.
Vazirani, V. (2001) *Approximation Algorithms*, New York, NY: Springer-Verlag.
Vossen, T. and M.O.Ball (2006) 'Optimization and Mediated Bartering Models for Ground Delay Programs', *Naval Research Logistics*, 53, 75-90.
Vossen, T., M.O.Ball, R.Hoffman and M.Wambsgans (2003) 'A General Approach to Equity in Traffic Flow Management and its Application to Mitigating Exemption Bias in Ground Delay Programs', *Air Traffic Control Quarterly*, 11, 277-292.
Vossen, T., R.Hoffman and A.Mukherjee (2011) 'Air Traffic Flow Management', in Barnhart, C. and B.Smith (ed.) *Quantitative Problem Solving Methods in the Airline Industry: A Modeling Methodology Handbook* , Norwell, MA: Springer.
Vranas, P., D.Bertsimas and A.Odoni (1994) 'Dynamic Ground Holding Policies for a Network of Airports', *Transportation Science*, 28, 275-291.
Wilber, R. (1988) 'The Concave least weight subsequence problem revisited', *Journal of Algorithms*, 9, 418-425.
Wolsey, L. (1998) *Integer Programming*, New York, NY: Wiley-Interscience Publication.
Young, H.P. (1994) *Equity in Theory and Practice*, Princeton, NJ: Princeton University Press.