Heuristics for Airline Schedule Recovery via the Virtual Hub Model

By

Jason A. Loy

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 7, 2004

Author_____
Department of Electrical Engineering and Computer Science
May 7, 2004

Certified by_____
John-Paul Clarke
Thesis Supervisor

Accepted by_____
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Heuristics for Airline Schedule Recovery via the Virtual Hub Model

By

Jason A. Loy

Submitted to the
Department of Electrical Engineering and Computer Science

May 7, 2004

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Airlines incur significant additional costs when bad weather at a hub-airport causes delays and cancellations throughout their entire network. One new recovery strategy called the virtual hub alleviates the effects of large delays at a hub-airport by setting up and diverting flight legs to nearby virtual-hub-airports. The effectiveness of the virtual hub strategy is tested through simulated days using the MIT Extensible Airport Network Simulator (MEANS). Increasingly complex heuristics are implemented to perform the virtual hub.

Results indicate that the virtual hub recovery strategy can reduce the number of passengers going through the hub that get abandoned by 38.0%, the number going through the hub with significant delays by 30.4% , and the average flight leg delay of the airline can be reduced by 49.5%. The heuristics are able to produce effective solutions in a matter of a few minutes.

Thesis Supervisor: John-Paul Clarke
Title: Associate Professor, MIT Department of Aeronautics and Astronautics

Acknowledgements

I would first and foremost like to thank my thesis advisor, Professor John-Paul Barrington Clarke. He has provided me both with an interesting thesis topic and the guidance to get it done.

I would like to thank Terran Melconian, the undisputed leader of my research group. He helped me to gain an understanding of MEANS, and explained the Airline Industry in a way even a programmer could understand.

I would like to thank Elizabeth Bly and Fabio Rabbani, who helped me to figure out all the nitty-gritty details behind airplanes, airports, and airlines.

I would like to thank Michelle Karow, whose work on virtual hubs paved the way for my research.

I would like to thank Jennifer, my family, and my friends, who have all been very patient and supportive of me during my stay at MIT.

I would finally like to acknowledge all of the classes and professors I have had this graduate year at MIT. But not for the previous years, because I had to pay for those.

Table of Contents

List of Figures

1 Introduction

For the past several decades, the airline industry has increasingly grown dependent on the hub-and-spoke system of airports because it allows an airline to reduce its costs by concentrating its resources and operations at a few airports. Furthermore, it allows airlines to supply more flights between destinations; more flights mean a larger market share and ultimately a large portion of the pool of customers. In part, the development of the hub-and-spoke system has been catalyzed by the development of effective optimization tools that produce schedules with little or no slack and corresponding high efficiency.

However, the hub-and-spoke system also has its shortcomings. One of the most significant shortcoming manifests when there is inclement weather at a hub airport. With so many flights connecting through the hub airport, a single storm at the wrong location can cause massive delays throughout the entire network. These delays result in delayed and cancelled flights. The resulting cost due to lost revenue, dissatisfied passengers, and additional operational and crew costs is estimated to be as much as $440 million a year for just one major US carrier, according to a January 21, 1997 New York Times article.

The virtual hub strategy was recently developed to try to combat this vulnerability of hub airports. The idea behind the strategy is to set up "virtual hubs" at nearby airports where flights can be diverted when there is inclement weather at the hub airport. These diverted flights would both reduce traffic (and thus delays) at the hub airport, and would also set up smaller hub-and-spoke systems at the virtual hubs. Virtual hub candidates would of course need to be able to handle the extra flights that have been diverted. In addition, they would need to be close enough to the hub airport that the additional flying time was not too great, but also far enough to not be affected by the same weather that is impacting the hub airport. In her Masters of Science thesis, Karow has shown that there

are good virtual hub candidates, and that they have strong potential to reduce delay over the network[1].

In this thesis, we develop and test intelligent airline agents that perform the virtual hub strategy within the MIT Extensible Airport Network Simulation (MEANS). In order for the simulator to run in a reasonable amount of time, the agents use a set of heuristics. These heuristics allow the agents to make quick, effective decisions over the large amount of flight and passenger information in the network. Although the heuristic score will always be an underestimate of virtual hub performance, it may achieve a close approximation in only a fraction of the time as a more thorough algorithm. Increasingly complex heuristics are developed and tested. In addition, a passenger reaccomodation model is developed to facilitate the agent.

In summary, this thesis aims to achieve the following goals:

- Test the viability of the virtual hub strategy using MEANS
- Develop heuristics to perform the virtual hub strategy
- Examine the trade off between optimization and calculation time for the heuristics

2 Virtual Hub Previous Work

The Virtual Hub strategy was developed by Michelle Karow in her Master of Science thesis supervised by John-Paul Clarke. In her work, Karow tested the virtual hub strategy for a given day and found that it reduced significant passenger delays by 94% and flight cancellations by 15%. [2]

Karow developed one virtual hub and tested it for a given airline on a given day. She identified one airport that was a good candidate virtual hub for a nearby hub, and then found a day where the hub had large delays because of a storm, and where the virtual hub had fair weather. She then developed and evaluated a virtual hub algorithm,

[1] Karow, Michelle J. "Virtual Hubs: An Airline Schedule Recovery Concept and Model." Masters of Science in Transportation Thesis. Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge MA. (2003)
[2] Karow, Michelle J. "Virtual Hubs: An Airline Schedule Recovery Concept and Model."

where the flights at the hub were partitioned into time windows, and the flights in each window to be diverted were determined using a network flow model that maximized passenger throughput. Disrupted passengers were re-accomodated after the diversions in each time window were determined. This algorithm was tested on paper using the flight schedule and passenger itineraries for the day in question.

The work presented in this thesis builds upon Karow's work. Specifically, we utilize MEANS to develop more complex algorithms to perform the virtual hub strategy. In addition, MEANS will provide a better testing platform for these algorithms under more real-world constraints.

3 MEANS

The MIT Extensible Airport Network Simulation (MEANS) simulates the movement of flights and passenger in the U.S. National Airspace system for any given day. Given initial flight schedules, passenger itineraries and weather information, MEANS will simulate the day and produce statistics for flight and passenger delay and cancellation.

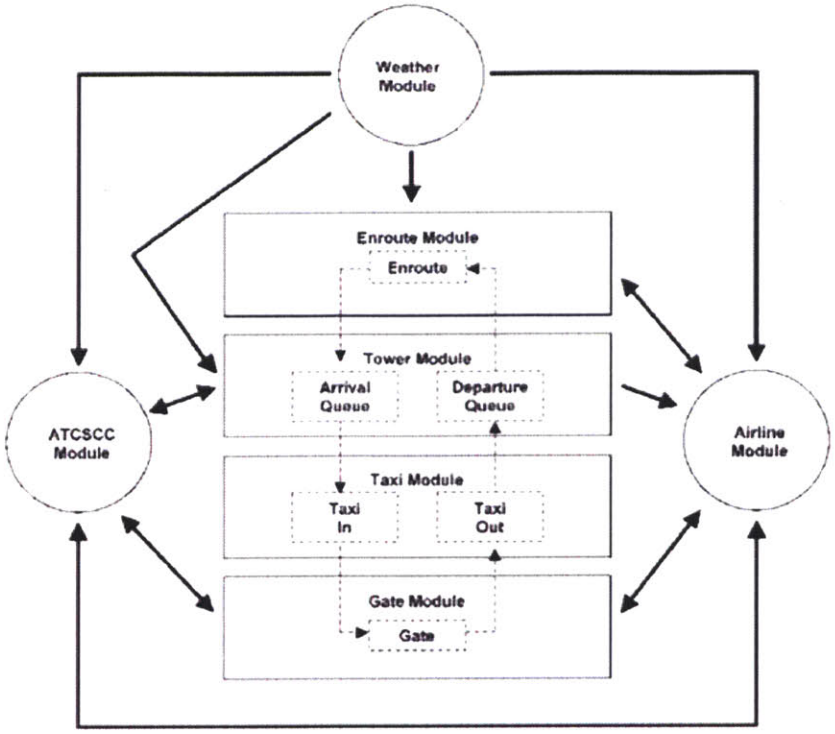


Figure 3.1: MEANS Module Design

MEANS has a modular design in which each portion of the flight process is modeled as a separate event. All flights are considered to be in one of the following phases: gate, taxi out, departure queue, enroute, arrival queue, and taxi in. Each of these phases is simulated by a different module (or sub-module). In addition, there are also modules to handle weather, traffic, flow-management, and airline decision making.. This modular design allows the different aspects of the simulation to be run at different levels of detail; for a given run, less important factors can be set to faster, more trivial implementations. In addition, this design allows for specialized and continuing improvement of the simulation.

MEANS has an airline module that can act as an intelligent airline agent during a simulation run. The airline module has the ability to look at the network information and modify the flight schedules and passenger itineraries, allowing the airline to make adjustments as the simulated day progresses. The airline module can be implemented for

any airline behavior, and can be used to test the results of such behavior on flight and passenger delays.

Two important modules that the airline agent interacts with to set up the virtual hub are the traffic flow management module and the tower module.

3.1 Ground Delay Program

A Ground Delay Program (GDP) is initiated at an airport when the airport is expected not to be able to handle all of its flight traffic. This generally occurs when there is inclement weather, and planes must be spaced further apart. A GDP at the hub airport is used as the first indicator for enacting the virtual hub recovery strategy, since it indicates that the hub is over capacity.

A GDP utilizes a set of slots to manage the overcapacity at the airport. Each plane is assigned to a slot of time in which it may arrive. The slots are given to flight legs in the same order as the original schedule (with a few exceptions such as for international flights). The slots effectively "stretch" out the schedule to allow more time between arrivals. An airline owns the slots assigned to its flight legs, and may use the slots however it chooses. However, if the airline is unable to use a slot, it is given to the airline with the following slot, and the initial airline is given priority to claim the newly opening slot. This is repeated until the airline is able to find a slot that it can use

3.2 Tower Module

Pareto frontiers are used in MEANS to model the arrival and departure rates at airports. Real rates are set through a combination of runway configuration, weather, and tower controller behavior. Pareto frontiers are generated through data analysis of airport arrival and departure rates, and serve as an adequate prediction of the maximum rates for an airport. Each airport has two Preto frontiers- one for good weather where aircraft must be flown according to VFR, and one for bad where aircraft must be flown according to IFR.

4 Airline Module

The Airline Module acts as an intelligent airline agent during MEANS runs. The Airline Module receives notifications of important events as they occur within MEANS and takes the appropriate actions to deal with them. The Airline Module is instantiated for each airline, so that each airline may be represented by its own agent and implementation.

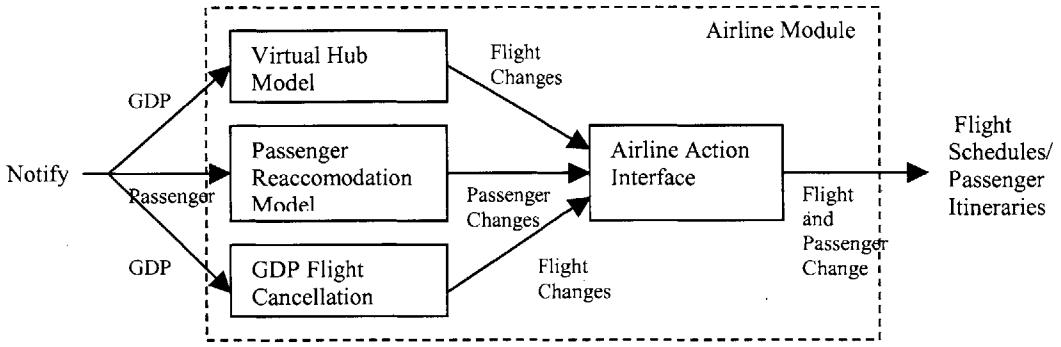


Figure 4.1: Airline Module

In this research, the Airline Module performed three tasks: utilizing the Virtual Hub strategy, passenger reaccomodation, and flight leg cancellation. The type of notification sent to the Airline Module determines the task that is performed. When a GDP is initiated at an airport, each airline is notified of their slots and the accompanying delays. The Airline Module then solves the Virtual Hub Model and the Flight Cancellation Model in response to these delays. When a passenger is unable to make it to his final destination, the airline is notified. The airline then performs the passenger reaccomodation model to try to determine the best way to get the passengers to their final destination. The Virtual Hub Model, Flight Cancellation Model, and Passenger Reaccomodation Model enact changes to the actual flight schedules and passenger itineraries through the Airline Action Interface.

4.1 Airline Action Interface

Design

The Airline Action Interface provides an interface by which other modules in the Airline Module may make changes in flight schedules and passenger itineraries. It contains a set of functions that represent "atomic" changes:

Flight
- Adding a new flight leg to the schedule
- Removing a flight leg from the schedule

Passenger
- Abandoning a passenger
- Removing a passenger from a set of flight legs
- Adding a passenger to a set of flight legs

These "atomic" changes in the Airline Action Interface ensure that the Virtual Hub Model only makes changes that are consistent with real world constraints. They prevent the other modules from modifying flight or passenger information that would disrupt the continuity of time or geography in the simulator. For example, they prevent the airline from changing a passenger's location or a flight leg's actual arrival time. For the functionality that the Airline Action Interface does permit, it makes sure that an airline owns object it is attempting to change, and makes sure the change is appropriate. For example, in removing a passenger from a flight leg, the Airline Actions Interface makes sure that the airline owns the passenger and that the passenger is actually on the flight leg.

Implementation

The Airline Action Interface was implemented as a namespace of functions. The Virtual Hub Model, Flight Cancellation Model, and Passenger Reaccomodation modules

each include the namespace and use its functions to perform flight and passenger changes.

4.2 Flight Cancellation Model

A Ground Delay Program at an airport often results in major delays for flights arriving there. Flight legs arriving at the GDP airport are delayed. This causes delays in some if not all of the future flight legs flown by those delayed aircraft. In order to reduce major delays, airlines often cancel flight legs. Since the majority of major delays occur during GDPs at airports, the Airline Module uses GDP calls to target flight legs to cancel due to delay. In this thesis, we incorporated existing code in which any flight leg that is expected to be delayed more than two hours is cancelled.

5 Passenger Reaccomodation Model

The Passenger Reaccomodation Model takes care of passengers who are unable to reach their final destination on their planned itinerary. This situation happens if a passenger misses his connecting flight leg or if one of his flight legs is cancelled. The Passenger Reaccomodation Model finds the best set of flight legs to get the passenger to his final destination and moves the passenger to the new set of legs.

5.1 Design

The Passenger Reaccomodation Model consists of four parts: a preprocessor, a flight indexer, an itinerary search algorithm, and a conflict resolution algorithm.

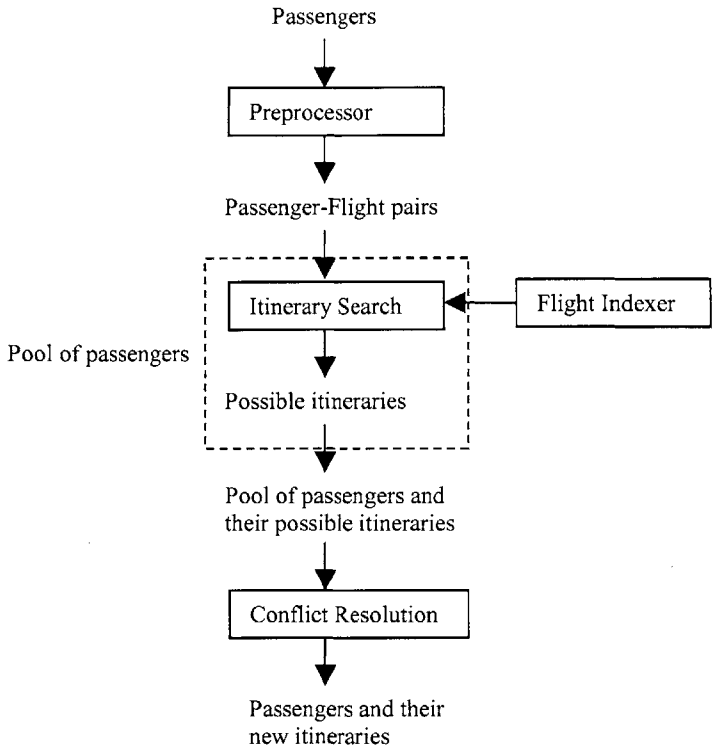


Figure 5.1: Passenger Reaccomodation Model

The Passenger Reaccomodation Model receives a list of the passengers and the problem flight leg from the Airline Module. This list is preprocessed and then sent to the Itinerary Search Algorithm. The Itinerary Search Algorithm searches for possible new itineraries that a passenger can use to reach his final destination, and returns a set of possible itineraries for a passenger. The Flight Indexer speeds up the accessing of possible flight legs during the Itinerary Search Algorithm. The Conflict Resolution Algorithm takes a group of passengers and their possible new itineraries, and selects itineraries that optimize reaccomodation for the entire group.

5.2 Preprocessor

The list of passengers and their problem (i.e. cancelled or missed) flight legs are preprocessed before a search is performed on them. Rather then reaccomodate a passenger starting at the problem flight leg, he is reaccomodated at the earliest flight

legpossible. This allows more flexibility when searching for schedules. It is assumed that if a passenger must be reaccomodated, he does not care the amount of legs in his itinerary that must be reaccomodated, especially if it can reduce his delay. For each passenger, the Preprocessor determines the next flight leg that the passenger will fly on. These passenger-flight leg pairs are then passed to the Itinerary Search Algorithm.

5.3 Flight Indexer

The Flight Indexer speeds up the retrieval of flight legs during the Itinerary Search Algorithm. The Flight Indexer contains only the flights that the airline owns. In addition, it filters out any flight legs whose status prevents them from taking on new passengers, for example if the flight leg is already in the air. It then separates the flight legs according to the departure city, and orders each group of flight legs by the expected departure time. A group of possible flight legs is retrieved for a given departure city and beginning time. Because each group is sorted according to time, the beginning time can be found by continuously dividing the group in half, and selecting the half that contains the beginning time. Thus, a group may be retrieved in logarithmic time.

The Flight Indexer is updated as the airline makes changes to flights. These changes may involve adding, canceling, or delaying a flight leg. The Flight Index performs the necessary insertions, removals, and resorting. Flight legs that no longer have eligible status to reaccomodate passenger are not filtered out, however, since the passage of time removes most of them anyways.

5.4 Itinerary Search Algorithm

The Itinerary Search Algorithm uses a modified A* search to look for a given number of the best passenger itineraries. It uses a time cost to guarantee that it finds the quickest itineraries, and uses a time prediction heuristic to bias the search towards the goal airport. The algorithmic flow is shown in figure 5.2.

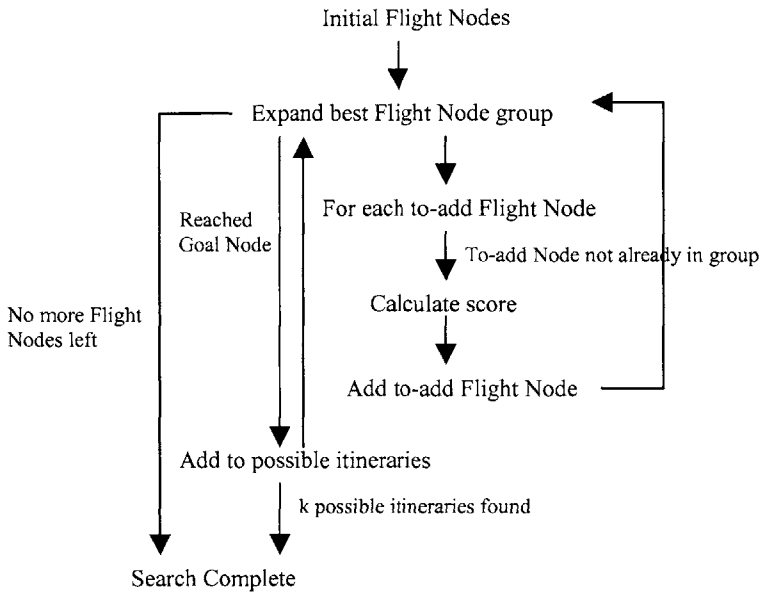


Figure 5.2: Itinerary Search Algorithm

As shown in the figure, flight legs serve as the search nodes. A flight leg node is expanded by finding all flight legs that match location and time constraints. It finds all flight legs that depart from the airport that the flight leg node arrives in. It then filters for the flight legs that the passenger would have time to make the connection from the current flight leg node. The search is successful when it expands a flight leg that arrives at the passenger's final destination. When the search has found a given amount of possible itineraries or runs out of nodes to expand, the search is complete.

An itinerary is scored by how soon the passenger reaches his final destination. The cost of a group of flight leg nodes is the sum of the arrival time of the last flight leg in the group and an underestimate (as required by an A* search) prediction of the remaining time required to reach the destination. The remaining time is calculated to be the minimum required connection time for the passenger to move between flight legs plus the minimum flying time between the arrival airport of the last flight leg in the group and the final goal airport. The normal A* search algorithm uses a list of expanded nodes to prevent backtracking during the search. However, the normal A* search only attempts to find the best itinerary, while the Itinerary Search Algorithm attempts to find a given

number of the best itineraries. Thus, instead of using a list of expanded nodes, when the Itinerary Search Algorithm expands a node and finds nodes to add, it makes sure that the node to add is not already in the group of nodes in order to prevent backtracking.

5.5 Conflict Resolution Algorithm

The Conflict Resolution Algorithm matches passengers to new itineraries in a way that optimizes the number of passengers that can be reaccomodated and thereby reach their final destination. It takes in the passengers to be reaccomodated, and each passenger's set of possible itineraries. It then tries to assign each passenger to a particular itinerary that follows the constraint of aircraft capacity in order to minimize the number of passengers who can not be reaccomodated.

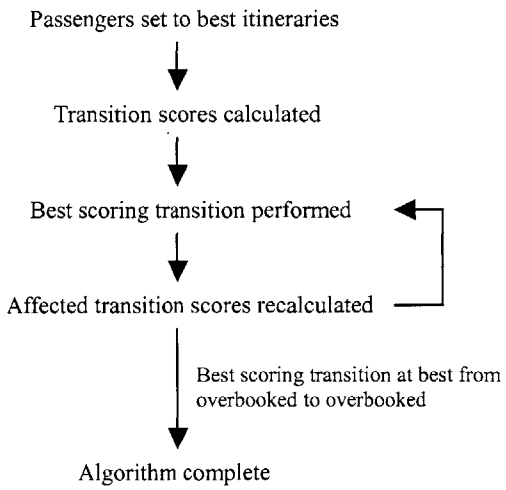


Figure 5.3: Conflict Resolution Algorithm

The Conflict Resolution Algorithm starts with the set of best possible itineraries for each passenger that is calculated through the Itinerary Search Algorithm. It initially assigns each passenger to his best possible itinerary. It then attempts to reassign passengers to different itineraries in order to maximize the number of passengers who can reach their destination, following the constraint of aircraft capacities.

For each passenger, the algorithm calculates a score for moving from the current itinerary to the other itineraries, and figures out the best scoring transition. The scoring

metric takes into account the degree to which the flight legs in the current itinerary and the target itinerary are overbooked. An itinerary with at least one overbooked flight leg is considered in conflict. It determines the type of transition from the current to the target itinerary: conflict-nonconflict, conflict-conflict, nonconflict-nonconflict, and nonconflict-conflict. In addition, it calculates the change in the number of overbooked passengers on each leg of the itineraries and the change in delay for the passenger. The transition type is the most important measure, then the change in number of overbooked passengers, then the change in passenger delay.

Primary	Secondary	Tertiary
Transition type	Change in number overbooked on legs	Change in passenger delay

Figure 5.4: Conflict Resolution Scoring

The passenger with the highest transition score is transitioned to his best itinerary, and the affected transition scores are recalculated. For each passenger, the Conflict Resolution Algorithm keeps track of the best transition and the flight legs that affect each itinerary. It also keeps track of the flight legs that affect each passenger in a table.

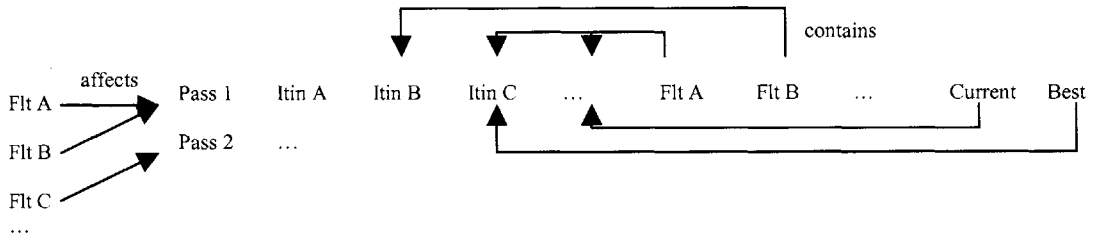


Figure 5.5: Conflict Resolution Table

This allows the Conflict Resolution Algorithm to quickly update just those passengers and itineraries affected by the transition made.

The algorithm is complete when the best transition is no longer from a conflict itinerary to a nonconflict itinerary.

5.6 Additional Greedy A* Search Design

The Passenger Reaccomodation Model may also be simplified to be used as an A* search. Instead of finding several possible flight leg itineraries in the Itinerary Search, it can also be set to find just one. Then instead of performing the Conflict Resolution to optimize over the entire group, the passenger could just be greedily reaccomodated one at a time.

5.7 Additional Trivial Design

The Passenger Reaccomodation Model also incorporates an existing trivial reaccomodation algorithm. The trivial algorithm simply attempts to replace the one offending flight leg in the passenger itinerary with another flight leg. If that flight leg can be replaced, the passenger is moved to the new flight leg. Otherwise, the passenger is abandoned.

5.8 Implementation

The Itinerary Search Algorithm was implemented as its own class, which is instantiated in the Airline Module. The Conflict Resolution Algorithm was implemented as its own class, which is instantiated in the Itinerary Search Algorithm

6 Virtual Hub Model

The Virtual Hub Model (VHM) is part of the Airline Module within MEANS, and performs the virtual hub strategy.

The VHM consists of two main parts: the Virtual Hub Decision Model and the Virtual Hub Core.

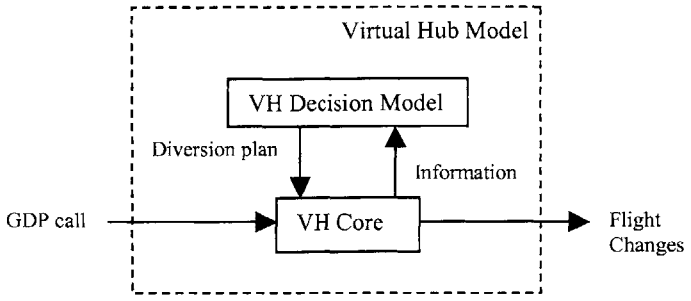


Figure 6.1: Virtual Hub Model

The Virtual Hub Decision Model determines the flight legs to be diverted to the virtual hub. In this thesis, we present different implementations of the Virtual Hub Decision Model, each of increasing complexity. Each Virtual Hub Decision Model interacts with the Virtual Hub Core, which handles all of the core functionality of the airline agent. The Virtual Hub Core both feeds information into the Virtual Hub Decision Model and executes the decisions made by that model.

6.1 Implementation

The Virtual Hub Decision Model and Core were combined into a virtual hub airline class. Two different frameworks were designed as two classes. The virtual hub trivial airline class inherits from the base airline class. The virtual hub greedy airline class inherits from the virtual hub trivial airline class.

6.2 Virtual Hub Core

The Virtual Hub Core is the "body" of the Virtual Hub Model. It performs many of the basic information gathering tasks, and well as executing the decision plan of the Virtual Hub Decision Model.

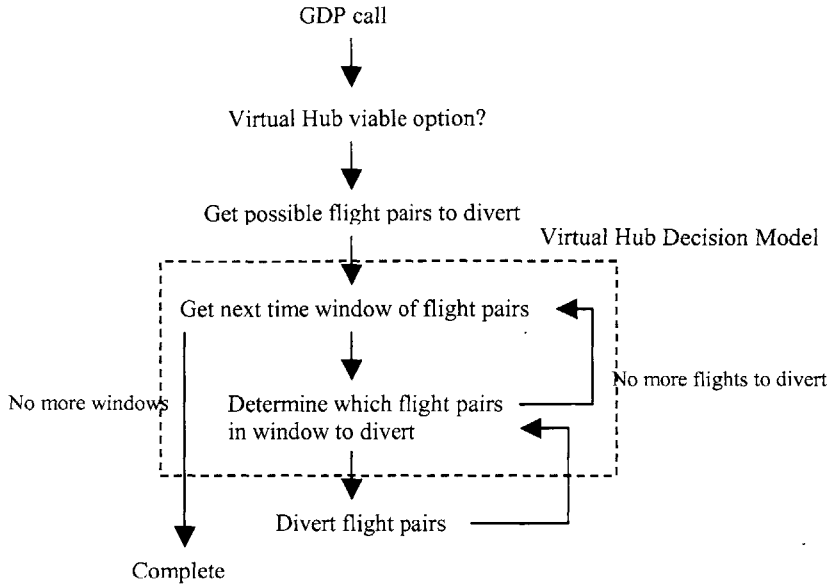


Figure 6.2: Virtual Hub Core and Virtual Hub Decision Model

Upon receiving a GDP call, the Virtual Hub Core determines whether or not creating a Virtual Hub is a viable option. It checks if the affected airport is a hub airport, and if there is a predetermined virtual hub for that airport. If so, it checks if the virtual hub is expected to have excess capacity during the time of the GDP. If the virtual hub will be able to handle enough additional arrivals to make the use of the virtual hub strategy worthwhile, then the virtual hub strategy is executed.

Once it has been decided to use the virtual hub, the Virtual Hub Core retrieves all of the candidate flight legs to divert from the hub to the virtual hub. It filters for flight legs that will arrive within the GDP window. These candidates are put into pairs of flight legs, consisting of the flight legs arriving to and departing from the hub for a given aircraft.

These candidate flight leg pairs are then sent to the Virtual Hub Decision Model. The Virtual Hub Decision Model performs two functions. First, it defines time windows in which to process the flight legs. Second, for each window, it determines which flight legs should be diverted.

The flight legs that should be diverted are then passed back to the Virtual Hub Core, which performs the diversion of each flight leg pair. Rather than modifying the

arrival and departure fields of the flight legs, the flight legs are cancelled and new flight legs with the new arrival and departure fields are added to the schedule. Passengers who had been on both legs of the flight leg pair to/from the hub are placed on the new flight leg pair. All other passengers are handled through the Passenger Reaccomodation Model.

Control then loops back to the Virtual Hub Decision Model, which determines if any more flight legs in the current window should be diverted. This looping allows small amounts of flight legs to be diverted, and then the effects take place before deciding on any more flight legs to divert. When there are no more flight legs to divert in the current window, it increments to the next window. When there are no more time windows left, the algorithm is complete.

6.3 Partial Passenger Reaccomodation

Normally, the passenger reaccomodation module either finds a new itinerary for a passenger or abandons him. During the Virtual Hub Model, however, many flight legs are being changed. Thus, although a passenger may not be reaccomodated at the moment, a future flight leg pair change may result in the passenger being able to be reaccomodated. So if a passenger can not be reaccomodated, he is removed from his itinerary and added to a list of previous unreaccomodated passengers. Each time a flight leg pair is changed, the passenger reaccomodation module goes through the list of unreaccomodated passengers to see if any can now be reaccomodated. Once the Virtual Hub Model is complete, the passengers on the list of unreaccomodated passengers are abandoned.

6.4 Virtual Hub Decision Model

The Virtual Hub Decision Model is the "brains" of the Virtual Hub Model. It determines which flight legs should be diverted to the virtual hub. It receives information from the Virtual Hub Core, and then tells the Virtual Hub Core which flight legs to divert. The Virtual Hub Decision Model performs searches and calculations through the

flight schedules and passenger itineraries in order to minimize flight and passenger delays and cancellations.

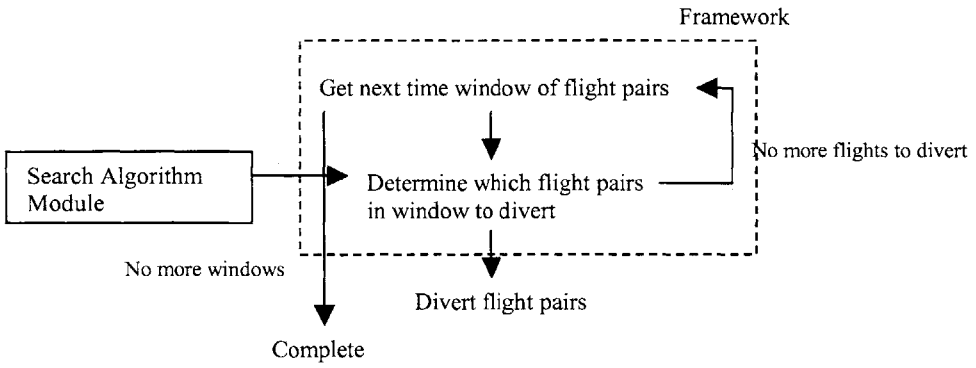


Figure 6.3: Virtual Hub Decision Model

The Virtual Hub Decision Model consists of a framework that determines the window and group diversion size. The framework interacts with the Search Algorithm Module, which provides the heuristics to score each flight leg diversion.

6.4.1 Framework

The Virtual Hub Decision Model framework is set up to allow different algorithms to be used to determine which flight legs to divert. It supports partitioning flight legs into time windows, where each window is processed one at a time. These windows allow each smaller group to be processed more quickly than one large group. In addition, the windows allow flight legs to be diverted in different distribution patterns in the GDP. Each possible flight leg pair changed is calculated for flight leg pairs in the window. The Virtual Hub Decision Model supports choosing a group of flight legs to divert all at the same time, or in diverting a few flight legs at a time and updating the simulator in between. Diverting one flight leg at a time allows MEANS to be updated between each diversion, allowing for more accurate calculations for the next iteration. Diverting a group of flight legs allows for greater speed.

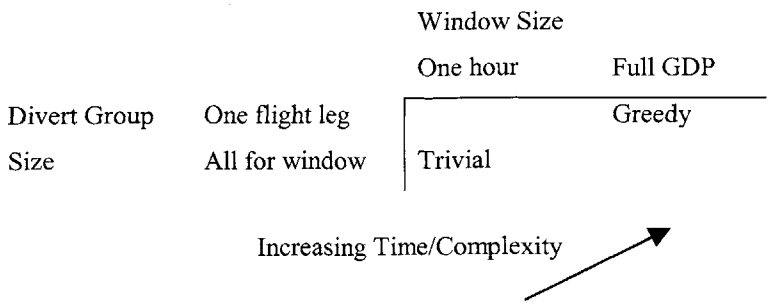


Figure 6.4: Window Size vs. Divert Group Size

Two Virtual Hub Model frameworks are implemented: trivial and greedy. The trivial framework is the fastest possible framework, while the greedy framework is the most complex. In the trivial framework, the score for each flight leg pair in the GDP is calculated exactly once. In the greedy framework, the score for each flight leg pair in the GDP is re-calculated after every change.

6.4.2 Trivial Framework

The trivial framework breaks flight legs into one hour windows, and selects flight legs to divert in the window all at once. It attempts to produce an algorithm similar to the work done by Karow on the virtual hub strategy.

For each time window, the trivial framework determines a group of flight leg pairs to divert. It first determines the number of flight legs to divert in the window based on the hub overcapacity and virtual hub under capacity. It then scores each flight leg pair, and the flight leg pairs with the lowest scores are diverted.

Time Windows

The trivial framework partitions the set of flight legs into subsets of one hour time windows so that the smaller groups of flight legs can be processed more quickly. In addition, it also ensures that diverted flight legs are evenly distributed throughout the entire GDP. One hour long windows are large enough that there is a large pool of flight

legs in each window to give flexibility in selecting the best flight legs from a group. But one hour is also small enough to make sure to produce an even distribution.
Flight leg pairs are placed into the window corresponding to its expected arrival time to the hub before the GDP was called. The pre-GDP time is used instead of the post-GDP time because this algorithm is attempting to restore flight legs to their pre-GDP times.

Number of Flight legs to Divert

In a given time window, the number of flight legs to divert is determined by the overcapacity at the hub and the under capacity at the virtual hub. The number of flight legs diverted is set so that either the hub reduces to full capacity or the virtual hub increases to full capacity.

The hub overcapacity for the airline is calculated to be the difference between the number of arrival flight legs for airline in the time window before the GDP call and the number after. For example, if there were 6 flight legs in the time window before the GDP call and 4 after, then the hub is 2 flight legs over capacity in the window. Were 2 flight legs to be diverted to the virtual hub, then the 4 other flight legs would be able to land in the time window, and no flight legs would carry over to the next time window. This calculation is only a rough estimation of the overcapacity. It implies that if a flight leg is diverted, the airline's next flight leg in the GDP will be able to take over the newly opened GDP slot. In addition, it only attempts to reproduce the hourly flight leg average of the original schedule. It doesn't take into account flight leg delay information, or the impact over the flight network.

The virtual hub under capacity for the airline is determined as the minimum of the tower under capacity and the gate under capacity. The tower under capacity is how many more flight legs could land at the airport, and is calculated to be the difference between the predicted flight leg capacity and the total number of arrival flight legs in time window. The predicted flight leg capacity is determined by the towerpareto frontiers for the airport. The frontier is set for an equal number of arrivals and departures. The gate under capacity is how many extra gates that the airline has available. At most airports, airlines lease gates, so this number is predetermined. It is assumed a flight leg will use a

gate for an hour. Since the time windows are each an hour long, the gate under capacity is the difference between the number of gates for the airline and the number of arrival flight legs for the airline in the time window.

6.4.3 Greedy Framework

The Greedy framework performs greedy flight leg pair changes. Instead of using time windows, it selects flight legs from the entire pool of candidates, until the virtual hub is full or there would be no gain in diverting more flight legs. A score is calculated for each flight leg pair of the airline in the GDP. The best scoring flight leg pair is then diverted, and the simulator updated. Scores are then calculated again, and the best flight leg is diverted. This loop continues until there are no more flight legs that meet the criteria for diversion.

7 Search Algorithm Module

The Search Algorithm Module uses all of the heuristics to score the flight leg changes associated with the virtual hub. The scores are based on expected changes in passenger and flight delays, and on the number of passengers who are able to reach their final destination. The Search Algorithm Module scores two types of flight leg changes: diversions and swaps. For each change, it uses a set of increasing complex scoring heuristics to analyze the direct impact of the change: trivial, one, two quick, and two. Each heuristic looks at a large portion of the network and will be explained in further detail in section 7.3. Each heuristic utilizes a list of previously unreaccomodated passengers generated by the partial passenger reaccomodation. All of the heuristics utilize a modified version of the list that contains information only for flight legs to or from the hub. The full heuristic uses the normal list, which contains the complete origin and destination of the passenger. In addition, the module also uses a heuristic to analyze the indirect impact on other flight legs and their passengers.

7.1 Diverting and Swapping

A pair of flight legs is diverted by sending the flight leg pair through the virtual hub instead of the hub. A pair of flight legs is swapped with another pair by sending the flight leg pair through the virtual hub, then swapping the destinations of the second legs of the two flight leg pairs. A swap can occur between two flight leg pairs that arrive at the hub within 30 minutes of each other. All of the future flight legs are also switched between planes.

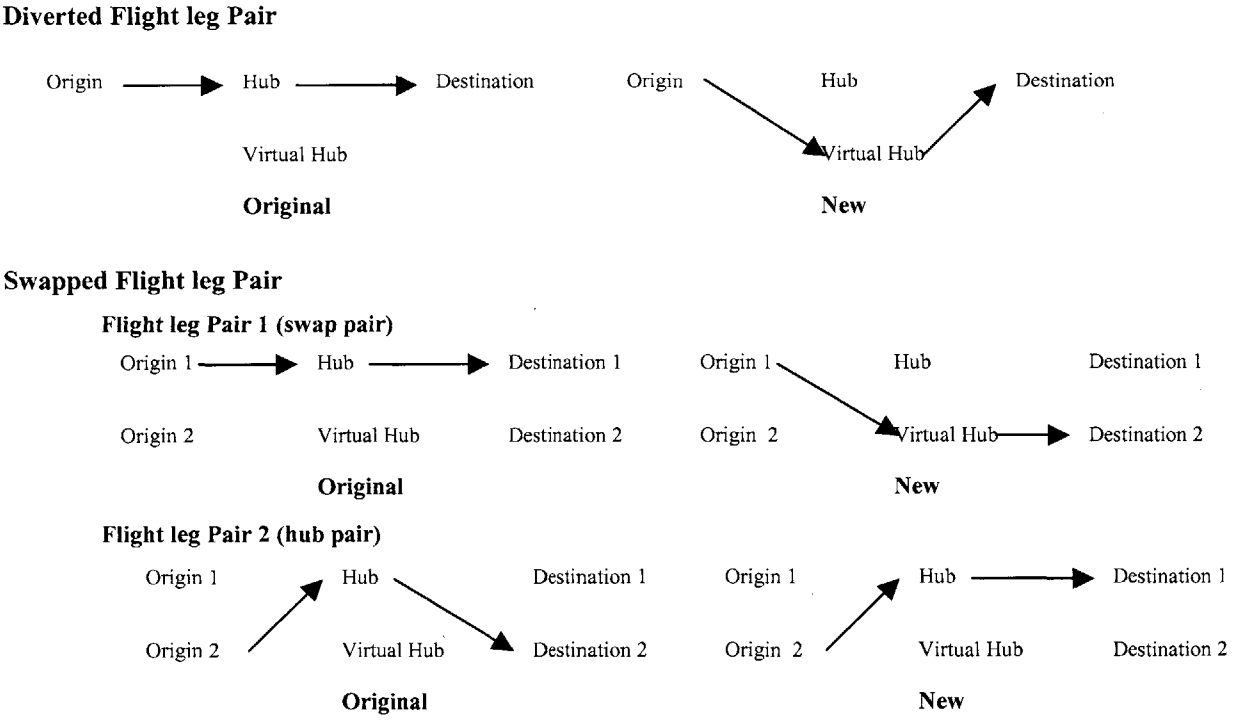


Figure 7.1: Diverting and Swapping

For a flight leg pair diversion, a passenger on both legs of the flight leg pair does not need to be reaccomodated. For a flight leg pair swap, the first leg of the hub pair is not changed, so passengers who are on that leg do not need to be reaccomodated. The second leg of the hub pair now covers the same airports as the original second leg of the swap pair. Thus, passengers just on the second leg of the swap pair are reaccomodated

onto the second leg of the hub pair, assuming that there is sufficient excess aircraft capacity.

A diversion results in a small change to the network. Passengers on just one leg of the flight leg pair have to be reaccomodated, but passengers on both do not. A swap causes a much larger disruption. Only passengers on the first leg of the flight leg pair through the hub (hub pair) do not have to reaccomodated. In addition, a swap results in more flight leg times being changed. A swap, however, also allows more flexibility. This flexibility is especially important in turning the virtual hub into a hub and spoke system. Since so many flight legs are being diverted to the virtual hub, it is very useful to be able to switch destinations to make sure as many passengers as possible at the hub and virtual hub end up where they need to be.

7.2 Scoring Metric

Each heuristic attempts to minimize flight and passenger delays and maximize the number of passengers that can reach their destinations. The optimization function is shown in Figure 7.2. A flight leg pair change must score past a threshold value to be considered a valid candidate.

Maximize

$$w_1 * (\sum f_1(P_{save} - P_{lost}))$$
$$+ \quad w_2 * (\sum f_2(P_{ds}))$$
$$+ \quad w_3 * (\sum f_3(F_{ds}))$$

P_{save} = passengers saved
P_{lost} = passengers lost
P_{ds} = passenger delay saved score
F_{ds} = flight leg delay saved score

Figure7.2: Search Objective

A passenger is lost if he is on a flight leg that is diverted or swapped and he can not be reaccomodated. A passenger is saved if he previously could not reach his destination, but after the flight leg pair change, he is able to. The passenger and flight leg delay scores are the difference between the original delay score and the delay score after

the flight leg pair change. Passengers who must be abandoned are not included in the delay scores.

Flight leg and passenger delays pass through a scoring function in order to facilitate finding an optimal network. Negative delays are not that significant, since the aircraft will most likely end up waiting to leave at the scheduled departure time. Delays over 15 minutes are very significant, since a flight leg or passenger is not considered "delayed" unless it is over 15 minutes late. The scoring function is thus biased for delays over 15 minutes and against negative delays:

$$\text{Score} = \begin{cases} w * \text{delay}, & \text{delay} < 0 \\ \text{delay}, & 0 \leq \text{delay} \leq 15 \\ \text{delay} + k, & \text{delay} > 15 \end{cases}$$

$$w < 1$$
$$k > 0$$

Figure 7.3: Scoring Function

7.3 Direct Impact

7.3.1 Trivial Heuristic

The trivial heuristic uses only minimal information in assessing a flight leg pair change. Passengers who are disrupted by the change are reaccomodated only for the disrupted leg, and the capacities of unchanged flight legs are not checked. Previously unreaccomodated passengers are recorded and organized according to their origin, destination, and origin departure time.

Diverting

Passengers originally on the diverted flight leg pair who must be reaccomodated are reaccomodated for the one disrupted leg. A search is performed to look for another flight leg that matches the departure and arrival airports of the disrupted leg and that leaves after the schedule departure of the disrupted leg. If one is found, then the passenger is reaccomodated. If one is not found, the passenger is considered lost.

Previously unreaccomodated passengers are then checked to see if they can be reaccomodated on the diverted flight leg pair. Reaccomodation is considered successful if the passenger origin and destination matches with the diverted flight leg pair departure and arrival airports, and the flight leg pair departs after the passenger's origin departure time. The flight leg pair aircraft capacity constraint is enforced.

Swapping

Passengers originally on one of the swapped flight leg pairs is reaccomodated for the disrupted leg. A passenger on the second leg of the hub flight leg pair is reaccomodated for the second flight leg. A passenger on both legs of the swapped flight leg pair is reaccomodated for the second flight leg. A passenger on just the first or second leg of the swap flight leg pair is reaccomodated for that flight leg.

Previously unreaccomodated passengers are checked to see if they can be reaccomodated on any of the new flight legs. Passengers can be reaccomodated on the swap flight leg pair, hub flight leg pair, first leg of the hub flight leg pair, or the second leg of the hub flight leg pair.

7.3.2 One Heuristic

The one heuristic adds a layer of complexity to the trivial heuristic by enforcing aircraft capacity constraints on all flight legs. Reaccomodation is performed in the same manner, i.e. only for the disrupted flight legs.

7.3.3 Two Quick

The two quick heuristic expands upon the one heuristic by expanding searches to all flight legs in the GDP and by utilizing the prediction passenger reaccomodation. The two quick heuristic first utilizes flight legs that are to or from the hub. It then searches for any new path through the hub or virtual hub that would reaccomodate the passenger.

Both the two quick and the two heuristic make use of prediction passenger reaccomodation. The prediction passenger reaccomodation uses the passenger reaccomodation module to predict how well a passenger could be reaccomodated by the flight leg pair changes. Changes are made as though the flight leg pairs were changed,

and the new flight leg pairs are temporarily added to the schedule. The passengers are then reaccomodated through the passenger reaccomodation model. Once the model completes, the results are measured. Then the changes are undone: passengers are reassigned to their original itineraries, and the new flight leg pairs are removed.

Previously unreaccomodated passengers are also checked to see if the new flight leg pair changes would allow them to be reaccomodated. Each changed flight leg is checked against the passenger to see if it allows a way for him to reach his destination.

7.3.4 Two

The two heuristic expands upon the two quick heuristic by expanding the search done on previously unreaccomodated passengers. It does not limit itself only to searching over the changed flight legs. Instead, it searches over all flight legs through the hub or virtual hub.

7.4 Indirect Impact

When a flight leg pair is changed, it has the indirect effect of reducing delay times for other flight legs. Later flight legs at the hub would be affected, as would the future flight legs of any affected aircraft. And any passenger on an affected flight leg would be affected. The indirect impact heuristics attempts to predict the reduced delay scores for indirectly affected flight legs and passengers.

7.4.1 GDP Flight Legs

When the selected flight leg pair is diverted, some of the airline's flight legs move to an earlier slot in the GDP. Thus, flight legs in the GDP later than the diverted pair are scored based on their reduced delays and number of passengers.

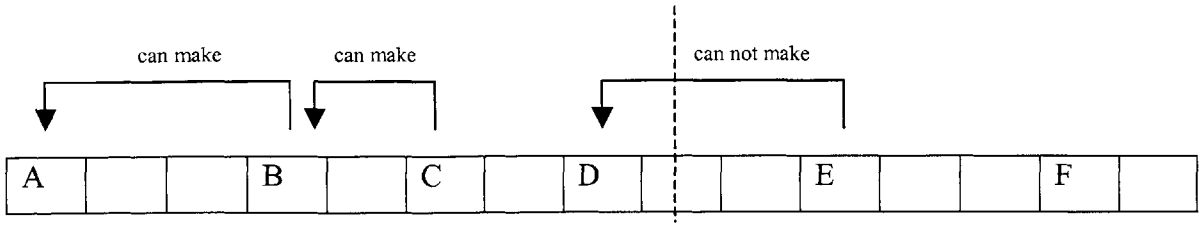


Figure 7.4: GDP slots

Flight legs move up a GDP slot until the next flight leg can't arrive in the slot time of the previous slot. In a real GDP, the airline would lose this slot, but get another later one. However, since it can not be known beforehand which later slot would be gotten, the algorithm ignores it and assumes that no later flight legs may be moved up. For each of the flight legs that do move up to an earlier slot, the reduced flight leg delay is calculated.

7.4.2 Future Flight Legs

For each affected flight leg in the GDP (including the changed flight leg pair), the future flight legs of that aircraft will also be affected. The reduced flight leg delay of the initial flight leg carries over to each of the future flight legs until there is no flight leg delay to be reduced. The flight leg delay may be reduced if there is an excess of connection time, which would allow a plane to depart with a smaller connection time.

7.4.3 Affected Passengers

The passengers on each affected flight leg (except the diverted flight leg pair, which has already been checked) are then scored using the expected delays and expected saved delays of the flight legs. If an affected flight leg is the last leg in the passenger's itinerary, the saved delay results in the passenger reaching his final destination sooner, thus reducing the passenger's delay. If the affected flight leg is not the last leg in the passenger itinerary, it is checked to determine if the saved delay gives the passenger a

better chance of making his next flight leg. The expected delay before diverting the flight leg pair is used to determine if the passenger was originally going to be able to make the connection to his next flight leg. It is assumed that the next flight leg leaves at its scheduled time. The expected saved delay is then used to determine if the passenger will now be able to make his connection. The saved delay is considered to save the passenger if before the diversion he was not expected to make his connection, but after he is. It is considered to lose the passenger if he was expected to make his connection before the diversion, but after he is not. Otherwise, the saved delay is neutral to the passenger.

8 Results

The Airline Module was tested on the same day and airline as Karow's work, using the same virtual hub candidate that she had identified. Weather, flight schedules, and the airline's passenger itineraries were inputted into MEANS. Since MEANS currently handles only the flight legs in the U.S. National Airspace System (NAS), international flight legs were filtered out of the passenger itineraries and their point of entry/exit was treated as an origin/destination.

Flight leg delays were measured according to two perspectives: relative and absolute. Relative delay is the amount of extra time the flight leg took, compared to the expected amount of time. It is a measure of how much delay the flight leg added to the system. Absolute delay is how much later the flight leg arrived than it was originally scheduled to. Thus, a flight leg might arrive late and have a high absolute delay, but also depart late and thus have a low relative delay

8.1 Passenger Reaccomodation Model

All three implementations of the passenger reaccomodation model were tested for the case where no virtual hub was created. The time for MEANS to run was also measured.

Passenger Data

	Non-Abandoned	Abandoned	Avg. Delay (m)	Time to Run (s)
Trivial	167293	1370	23.4	65.9
Greedy	167837	3352	24.2	131.4
Full	167644	1268	24.4	453.3

Figure 8.1: Passenger Reaccomodation Model Results

The results show that the greedy passenger reaccomodation outperforms the two other implementations. It produces the least amount of abandoned passengers. Although the greedy implementation has a higher average delay than the trivial implementation, it is marginally higher, and is most likely due to the greedy implementation reaccomodating "difficult" passengers that would have a high delay. Although the full implementation is more complex than the greedy implementation, it is still a greedy search (by flight leg instead of by passenger), and is not guaranteed to outperform the greedy search. The time to run follows the pattern of increasing complexity.

Since the greedy passenger reaccomodation model is the most efficient, it is used in the tests on the virtual hub models.

Passenger Data

		Abandoned	Avg. Delay (m)	% Delay > 15
Trivial	Hub	1225	83.0	81.6
	Total	3352	24.2	28.6

Flight leg Data

			Relative	Absolute		
		Cancelled	Avg. Delay (m)	Avg. Delay (m)	% Delay > 15	Time to Run (s)
Greedy	Hub	2	38.2	81.9	55.6	131.4
	Total	6	11.5	20.8	31.3	

Figure 8.2: No Virtual Hub Greedy PRM Results

8.2 Priority Constraints

When testing the virtual hub, it was found that the greatest constraint was the passenger itineraries. Flight legs are diverted to the virtual hub while trying to maintain passenger itineraries, virtual hub capacity, and reduce delays of indirectly affected flight legs and passengers. However, the direct reaccomodation/unreaccomodation of passengers on the changed flight leg pairs had an overwhelming priority over the two other constraints. On average, the virtual hub had an excess capacity for 20 flight legs per hour. However, none of the virtual hub algorithms ever diverted this many flight legs. In addition, the use of the indirect scoring heuristic was found to have no effect whatever on the selection of flight leg pairs to change when combined with any of the direct scoring heuristics. In light of these findings, the indirect scoring heuristic was not used during the comparative testing of the direct scoring heuristics.

8.3 Trivial Framework

8.3.1 Diversion

The trivial framework was tested for each heuristic. It was first tested allowing only flight leg pair diversion changes.

Passenger Data

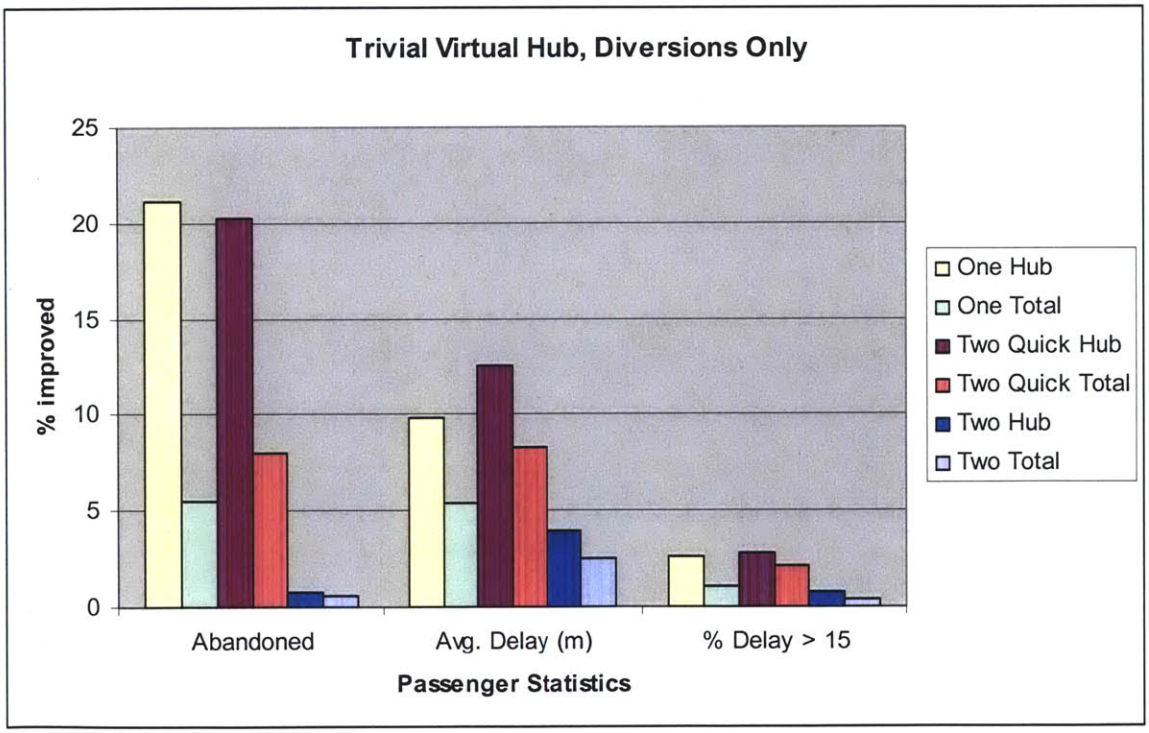
		Abandoned	Avg. Delay (m)	% Delay > 15
Trivial	Hub	1585	71	71.0
	Total	3831	21.9	26.5
One	Hub	966	74.8	79.5
	Total	3166	22.9	28.3
Two Quick	Hub	977	72.6	79.3
	Total	3083	22.2	28.0
Two	Hub	1215	79.7	81.0
	Total	3333	23.6	28.5

Flight leg Data

				Relative	Absolute		
		Changes	Cancelled	Avg. Delay (m)	Avg. Delay (m)	% Delay > 15	Time to Run (s)
Trivial	Hub	35	0	28.3	47.8	49.0	293.3
	Total		4	9.3	12.9	27.3	
One	Hub	12	0	34.5	73.2	55.9	190.9
	Total		4	10.5	18	29.9	
Two Quick	Hub	15	0	33.4	68.1	53.3	209.0
	Total		4	10.1	16.9	29.8	
Two	Hub	15	0	35.8	78.2	54.4	148.9
	Total		4	11.1	19.7	31.0	

Figure 8.3: Data Trivial Virtual Hub Airline, Diversions Only

The percentage improved over the non-virtual hub airline agent was calculated and plotted. The trivial implementation was omitted because its results were so poor.



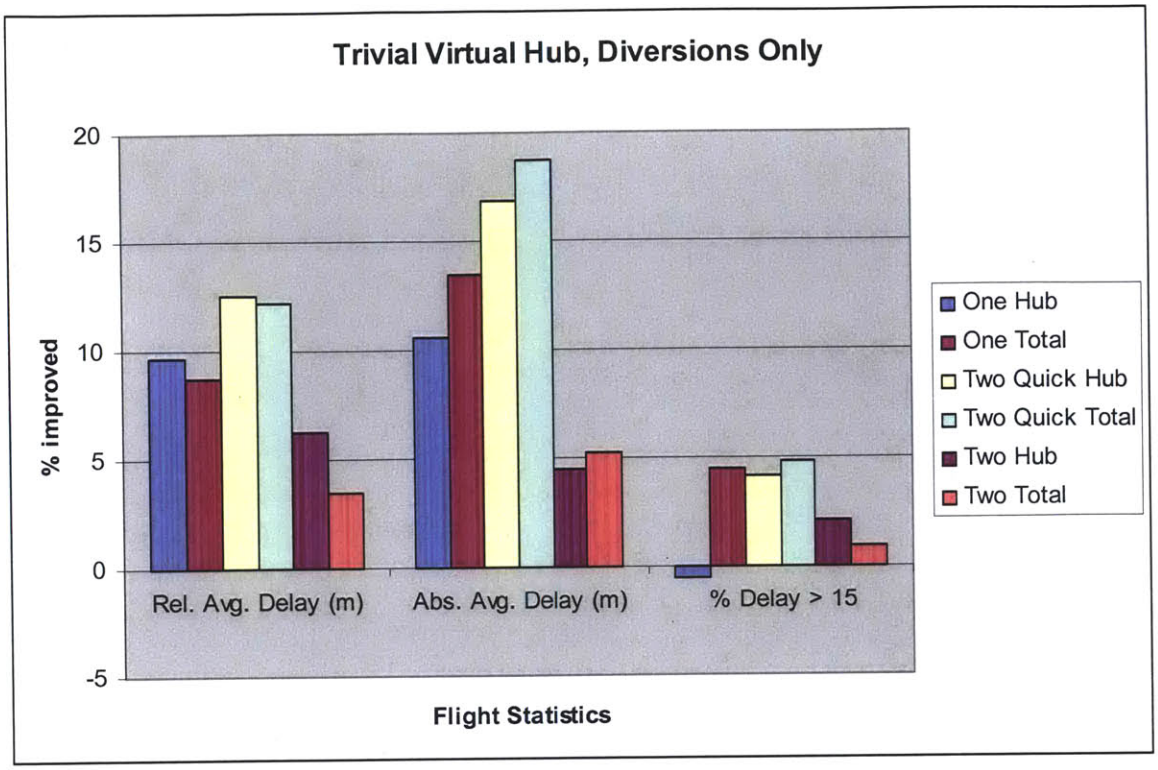


Figure 8.4: Plot Trivial Virtual Hub Airline, Diversions Only

The One Heuristics performed well, resulting in a 21.1% reduction in the number of abandoned passengers originally going through the hub (and 5.5% overall), and a 9.7% reduction in flight leg delays through the hub (and 8.7%) overall. The Two Quick Heuristic also performed well, and the Two Heuristic received marginal gain.

The Trivial Heuristics resulted in a 29.4% increase in the number of abandoned passengers originally going through the hub. This poor result is due to the heuristic not checking the capacities of flight legs to which it tries to reaccomodate passengers. Thus, it may divert several flight legs that it should not, and try to reaccomodate hundreds of passengers on the same flight leg. In addition, the Trivial Heuristic takes a much longer time to run, even though it is the simplest heuristic. The additional time to run is the result of the poor choices made by the Trivial Heuristic, resulting in a large number of passengers that become unreaccomodated. These passengers must be searched over for each scoring, and attempted to be reaccomodated between each window. This Trivial

Heuristic produces even poorer results for the other scenarios tested, and in general should be ignored as an oversimplified algorithm.

8.3.2 Diversion and Swapping

The Trivial framework was tested allowing for both diversions and swaps.

Passenger Data

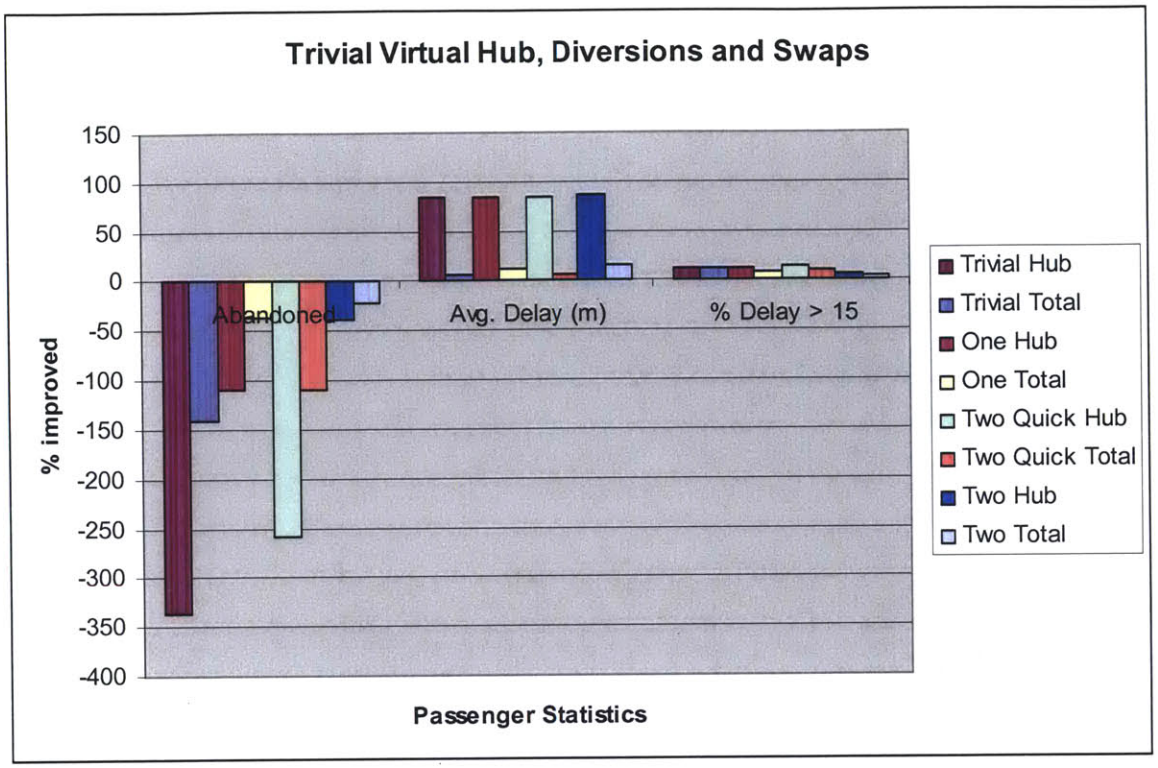
		Abandoned	Avg. Delay (m)	% Delay > 15
Trivial	Hub	5352	12.3	72.0
	Total	8072	22.8	25.3
One	Hub	2551	11.9	72.3
	Total	4579	21.1	26.4
Two Quick	Hub	4391	12.4	70.9
	Total	7006	22.5	25.8
Two	Hub	1700	11.1	76.4
	Total	4055	20.5	27.5

Flight leg Data

				Relative	Absolute		
		Changes	Cancelled	Avg. Delay (m)	Avg. Delay (m)	% Delay > 15	Time to Run (s)
Trivial	Hub	56	0	27.5	46.8	79.5	507.0
	Total		4	9.6	12.6	21.523.5	
One	Hub	29	0	30.1	48.5	82.6	278.1
	Total		4	9.4	13.4	25.2	
Two Quick	Hub	49	0	30.2	48.4	80.9	650.4
	Total		4	9.6	13.2	23.5	
Two	Hub	21	0	30.3	50.9	83.9	495.5
	Total		4	9.9	14.5	26.6	

Figure 8.5: Data Trivial Virtual Hub Airline, Diversions and Swaps

The percentage improved over the non-virtual hub airline agent was calculated and plotted.



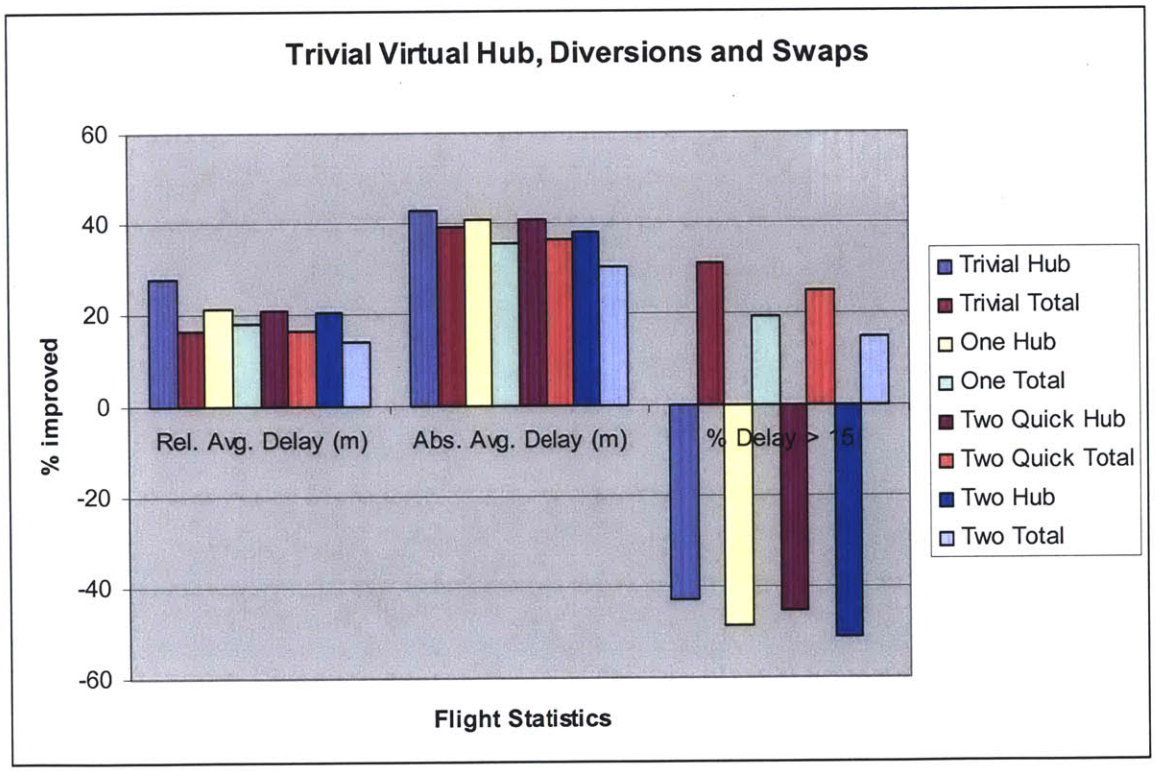


Figure 8.6: Plot Trivial Virtual Hub Airline, Diversions and Swaps

The results for the allowing diversions and swaps were consistently poor. Every implementation resulted in additional passengers being abandoned. Although the average flight leg delay was reduced 35-45% for each algorithm, it does no good if there are no passengers on the planes.

8.4 Greedy Framework

8.4.1 Diversion

The greedy framework was tested for each heuristic. It was first tested allowing only diversion changes.

Passenger Data

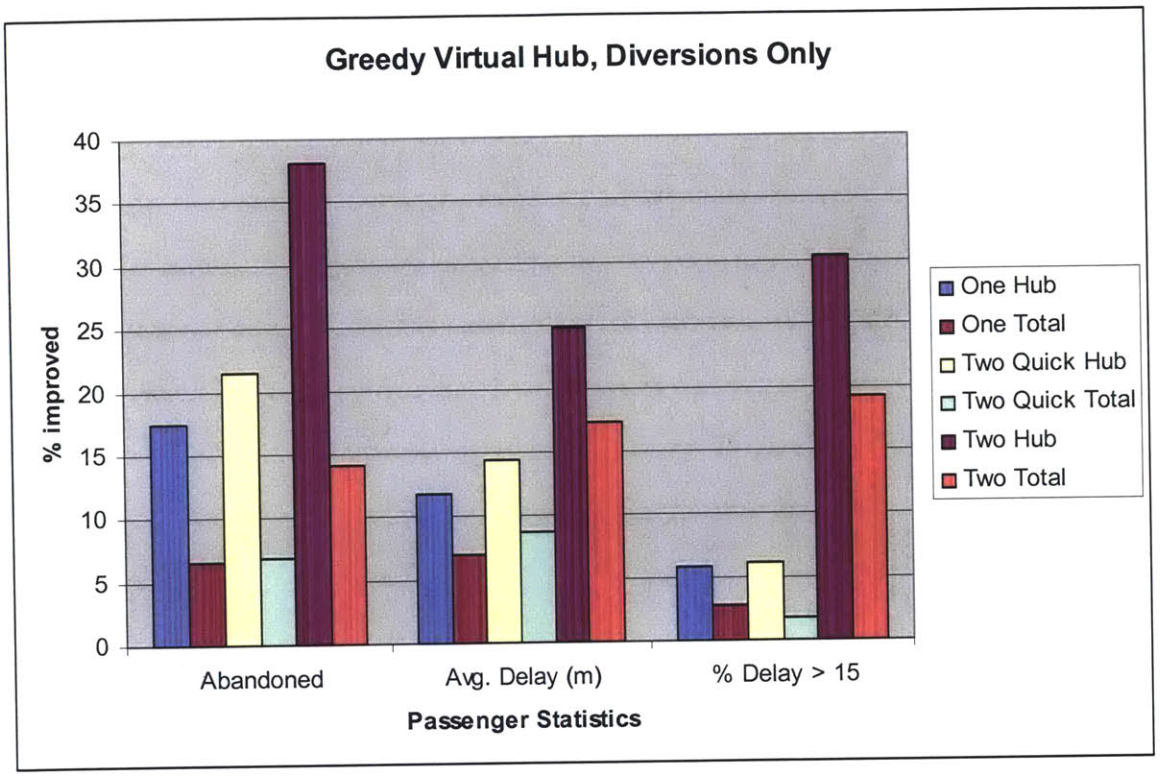
		Abandoned	Avg. Delay (m)	% Delay > 15
Trivial	Hub	5350	110.0	52.1
	Total	7492	27.7	22.0
One	Hub	1012	73.2	76.9
	Total	3131	22.5	27.8
Two Quick	Hub	962	71.1	76.6
	Total	3122	22.1	28.1
Two	Hub	759	62.4	56.8
	Total	2877	20.0	23.1

Flight leg Data

				Relative	Absolute		
		Changes	Cancelled	Avg. Delay (m)	Avg. Delay (m)	% Delay > 15	Time to Run (s)
Trivial	Hub	109	0	38.2	33.3	41.1	2531.3
	Total		4	11.5	8.7	16.1	
One	Hub	21	0	18.6	67.7	86.0	286.2
	Total		4	7.0	16.6	26.8	
Two Quick	Hub	23	0	31.1	62.3	85.9	378.3
	Total		4	10.0	15.5	26.8	
Two	Hub	46	0	20.4	39.9	64.6	215.6
	Total		4	7.2	10.5	20.0	

Figure 8.7: Data Greedy Virtual Hub Airline, Diversions Only

The percentage improved over the non-virtual hub airline agent was calculated and plotted. The trivial implementation was omitted because its results were so poor.



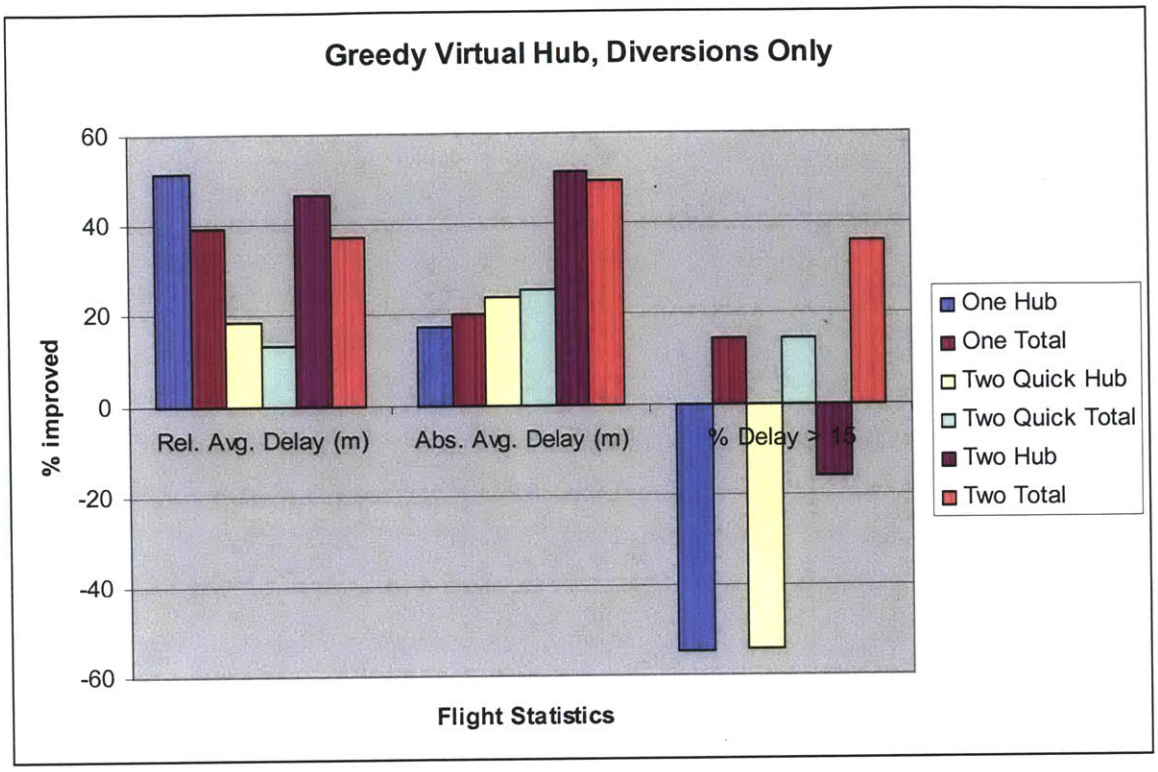


Figure 8.8: Plot Greedy Virtual Hub Airline, Diversions Only

The Two Heuristic showed a large improvement of 38.0% for abandoned passengers originally through the hub (and 14.2% overall). It also resulted in a reduction of 46% of the average flight leg delay through the hub (and 37.4% overall). Although the percentage of flight legs considered delayed (delay > 15 minutes) increased at the hub by 16.2%, the overall percentage decreased by 36%. The Two Quick and One Heuristics also performed well, producing significant reductions in abandoned passengers and delays.

8.4.2 Diversion and Swapping

The greedy framework was then tested allowing both diversions and swaps.

Passenger Data

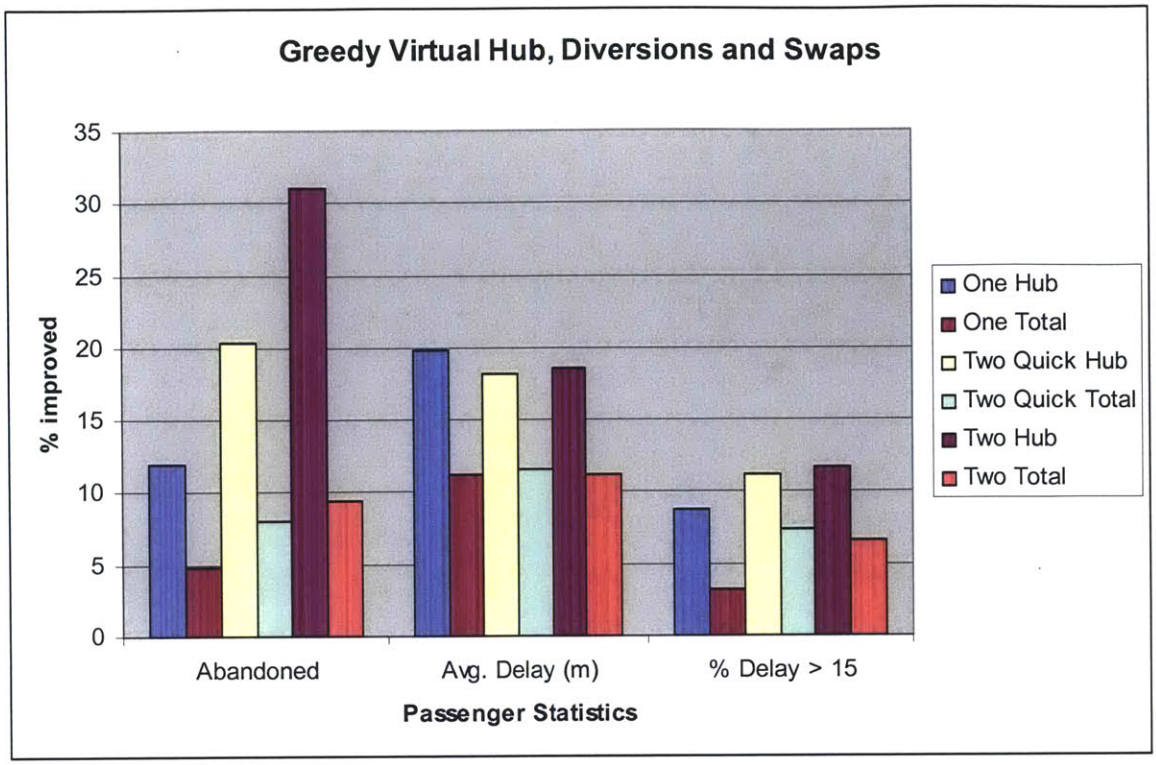
		Abandoned	Avg. Delay (m)	% Delay > 15
Trivial	Hub	5269	120.0	53.5
	Total	7643	29.5	22.4
One	Hub	1078	66.5	74.4
	Total	3192	21.5	27.7
Two Quick	Hub	975	67.9	72.5
	Total	3082	21.4	26.5
Two	Hub	845	67.6	72.0
	Total	3035	21.5	26.7

Flight leg Data

				Relative	Absolute		
		Changes	Cancelled	Avg. Delay (m)	Avg. Delay (m)	% Delay > 15	Time to Run (s)
Trivial	Hub	104	0	18.4	46.7	46.5	2402.1
	Total		4	7.2	10.3	16.9	
One	Hub	25	0	28.9	54.5	83.0	317.1
	Total		4	9.7	14.7	26.3	
Two Quick	Hub	33	0	31.4	54.0	77.7	734.4
	Total		4	9.6	14.3	24.8	
Two	Hub	36	0	27.3	51.4	81.0	768.5
	Total		4	9.1	13.6	24.7	

Figure 8.9: Data Greedy Virtual Hub Airline, Diversions and Swaps

The percentage improved over the non-virtual hub airline agent is calculated and plotted. The trivial implementation was omitted because its results were so poor.



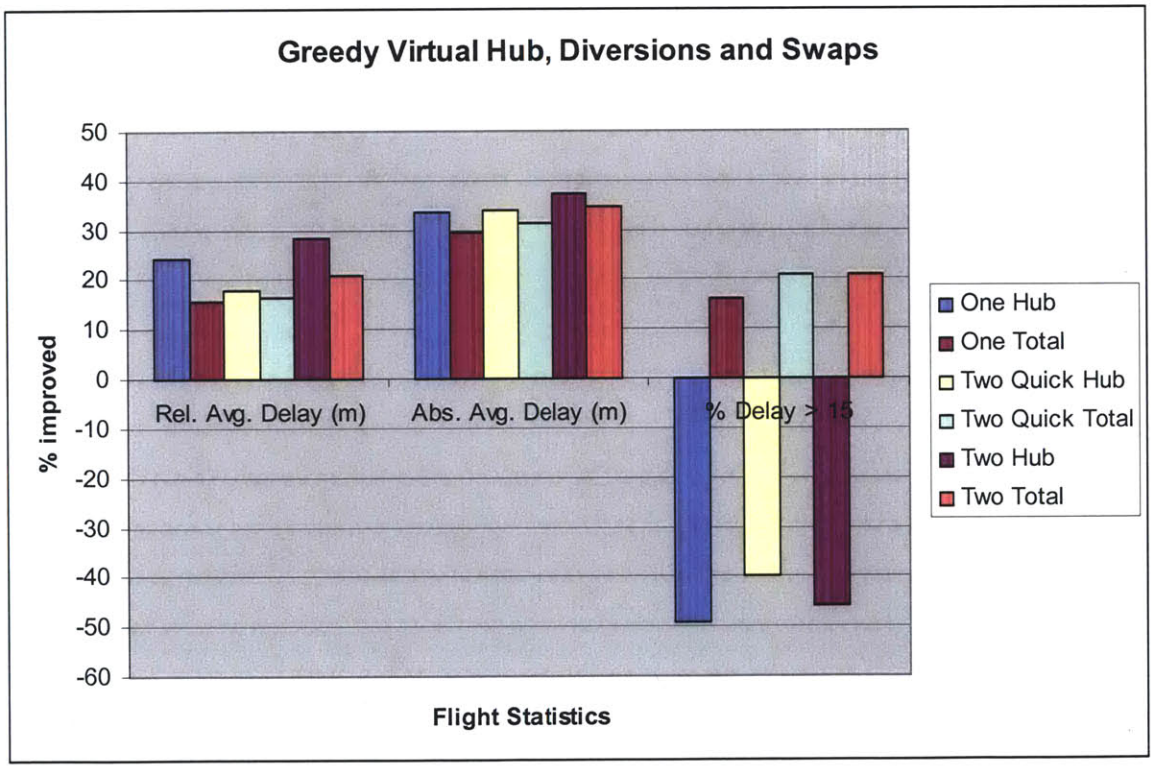


Figure 8.10: Plot Trivial Virtual Hub Airline, Diversions and Swaps

The Two Heuristic showed the most gain, with a reduction of abandoned passengers originally through the hub by 31.0% (and 9.5% overall), and reduction of the average delay through the hub by 28.5% (and 20.9% overall). The other algorithms also produced significant gains.

9 Discussion

Three of the four heuristics produced good results. Both versions of the Greedy Framework provided significant improvements, as did the Trivial Framework with Diversions only. The Trivial Framework with Diversions and Swaps resulted in more passengers being abandoned. These results are due to the simple design of the Trivial Framework. Since all flight leg pair candidates in a window are changed all at once, the framework would only work if the flight leg pair candidates have minimal affect on each other. For the case of just diversions, the change to the network is minimal, and the Trivial Framework is able to select flight leg pairs well. But for the case of diversions and swaps, a swap produces much more change than a diversion. It causes a larger change in the network, and affects more passengers. The Trivial Framework is unable to handle the effects between flight leg pairs, and so produces a poor selection of flight leg pairs to change.

Each heuristic must also be analyzed for its efficiency- how long it takes to get a good result. The time for the simulator to run is plotted against the percent reduction in abandoned passengers for each framework and heuristic.

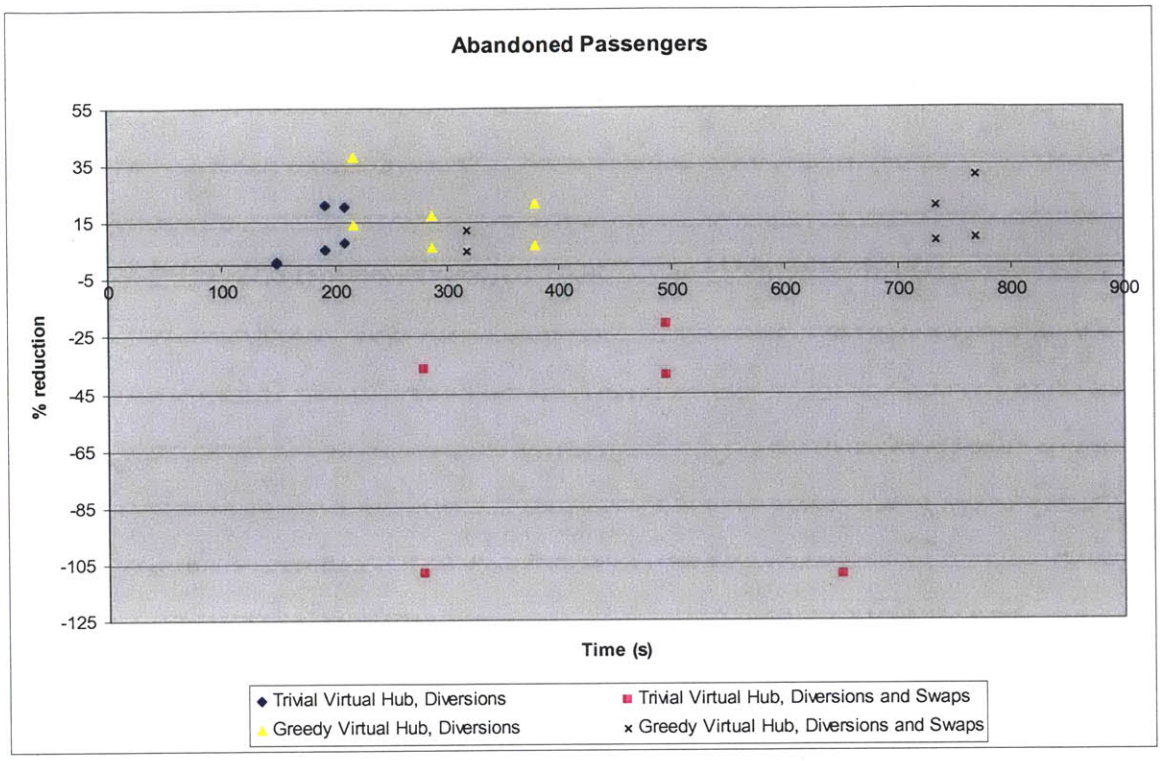


Figure 8.11: Plot of Heuristic Efficiency

The plot shows that the Greedy Virtual Hub Framework and the Trivial Virtual Hub Framework with Diversions only all produce about the same optimal solution. The Greedy Virtual Hub with Diversions only, however, produces the highest gain of 38%. The time required is consistent with the complexity of each algorithm. The Trivial Virtual Hub with Diversions tends to take the least amount of time, while the Greedy Virtual Hub with Diversions and Swaps tends to take the most amount of time.

The Trivial Virtual Hub with Diversions and Swaps produces poor results, and the time required is most likely lengthened by the cost of handling additional unreaccomodated passengers due to those poor results. Also notice that the best solution (produced by the Greedy Virtual Hub with Diversions) also takes the least amount of time for that heuristic. Since it produces fewer unreaccomodated passengers, the simulator can run faster.

10 Future Research

Although several heuristic-based models are evaluated in this thesis, it would also be advantageous to perform benchmarks for the virtual hub strategy and Passenger Reaccomodation Model. These benchmarks could be used to determine how close the heuristic models were to the best answer, and better understand the tradeoffs between computation time and accuracy.

In addition, more thorough algorithms could be devised to produce more optimal results. The purpose of this thesis was to find some fast, efficient algorithms that produce good results. This thesis uses only greedy-search algorithms, which are only guaranteed to find local maximums. Flight legs are scored independently of each other, rather than as a batch. And the weights were manually set, rather than trained on the data. More complex, time-consuming algorithms could be developed.

The Virtual Hub Model and MEANS could also be expanded to handle other real-world constraints. The largest constraint not handled would probably be crew assignments. Diverting flight legs to the virtual hub would affect crew schedules; either flight legs would have to make sure crews ended up where they needed to, or crews would have to be reassigned. Another large constraint would be the availability of ground crew at the virtual hub. An airline would need to have enough ground crew to handle the additional planes that are diverted to the virtual hub. And it may be difficult for the airline to assemble extra ground crew on such short notice.

The Virtual Hub Model could also be expanded to include a cost-based model. Currently, the Virtual Hub Model attempts to reduce flight and passenger delays and cancellations. With a cost-based model, the Virtual Hub Model could create a flight diversion plan that would minimize the airline's costs, in terms of operational costs and passenger refunds. A passenger may be a better candidate to divert because he has paid the least for his airfare, or a plane might be a bad choice to divert because of additional fuel and crew costs. A cost-based model has recently been developed for MEANS, and in the near future could be integrated into the Virtual Hub Model.

The Virtual Hub Model and MEANS could also be expanded to handle more interaction between airlines. Currently, each airline creates its own diversion plan

independent of other airlines. More realistic and more optimized solutions may be found by allowing airlines to communicate and cooperate. This may include things such as reaccomodating passengers on a different airline or coordinating recovery strategies during GDP.

11 Conclusion

The virtual hub strategy has shown itself to be a simple, yet effective recovery strategy to compensate for the frailty of the hub-and-spoke system. The best performing virtual hub heuristic was able to reduce the number of abandoned passengers going through the hub by 38%, the number going through the hub with significant delay by 30.4%, and was able to reduce the average flight leg delay at the hub by 51.3% (and 49.5% over the entire network). The virtual hub strategy would allow more passengers to get to their destinations more quickly, while also reducing flight leg delays. These reductions would ultimately result in reduced operating costs for the airline.

The virtual hub strategy's open simplicity allows many different formulations. Airlines would have tremendous freedom in deciding on the selection of flight legs to divert, the adjustment of the flight network, and the reaccomodation of passengers. In this thesis, we developed a set of heuristics of varying complexity, which were then tested for effectiveness. The heuristics that were developed have been shown to be very powerful in getting good results much more quickly than a more thorough analysis. In fact, the best performing heuristic was not even the most complicated or time-consuming one. The majority of the heuristics ran in 15 minutes or less, and produced significant reductions in passenger and flight leg delays.

There are some drawbacks to the virtual hub network. Not every airline may have good virtual hub candidates. And some airlines may find the gain not worth the disruption to the network. Furthermore, the analysis of the virtual hub strategy relied heavily on MEANS. Deeper analysis or even implementation of the strategy may produce less beneficial results or additional flaws. However, this thesis has shown that

the virtual hub strategy could be a powerful tool for the airline industry, and merits serious consideration as a recovery strategy.

Bibliography

Clarke, Michael D. D. "Development of Heuristic Procedures for Flight Rescheduling in the Aftermath of Irregular Airline Operations." Doctor of Science in Flight Transportation Thesis. Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA. (1997)

Hall, William D. "Efficient Capacity Allocation in a Collaborative Air Transportation System." Doctor of Philosophy in Operations Research Thesis. Department of Electrical Engineering and Compute Science, Massachusetts Institute of Technology, Cambridge, MA. (1999)

Karow, Michelle J. "Virtual Hubs: An Airline Schedule Recovery Concept and Model." Masters of Science in Transportation Thesis. Department of Civil and Environmental Engineering, Massachusetts Institutes of Technology, Cambridge, MA. (2003)

Lan, Shan. "Planning for Robust Airlne Operations: Optimizing Aircraft Routings and Flight Departure Times to Achieve Minimum Passenger Disruptions." Doctor of Philosophy in Transportation Thesis. Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA. (2003)

Lewis, Damon Marcus. "Evolving Efficient Airline Schedules." Master of Engineering in Electrical Engineering and Computer Science Thesis. Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA. (2000)

Melconian, Terran. "Effects of Increased Nonstop Routing on Airline Cost and Profit." Master of Science in Aeronautics and Astronautics Thesis. Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA. (2001)

Mitra, Trin A. "Measuring the Causes of Airline Customer Dissatisfaction." Master of Science in Transportation Thesis. Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA. (2001)

Morin, Massimo. "Metrics and Methods for Improving Airline Schedule Reliability." Master of Science in Transportation Thesis. Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA. (2001)