

The Intersection of Two Planes is a Line.

Airport Security and Delay Optimization Given Recently
Adopted Legislation

Team # 434

Abstract:

This report analyzes the optimal number of EDS baggage screening devices that airports should employ to comply with recently passed security regulations. We balance the cost and scarcity of the EDS machines against the need to minimize passenger delays. In addition, this report analyzes flight scheduling during the peak hour to minimize the delay time that scanning will cause. We identify three different scheduling strategies, and discuss the programming behind their development and the instances under which each should be implemented. We also include a program that allows an airport to determine the number of EDS devices that will be required based on its specific flight schedule and security requirements.

We also make specific recommendations as to the flight scheduling and number of EDS devices required by the two largest airports in the Midwest Region, airports A and B. We conclude that airport A requires 14 EDS machines and should vary its flight scheduling strategy based on the expected number of passengers coming through the airport on a given day. For busy days in which nearly all seats aboard the airplanes in airport A are occupied, we recommend that airport A employ the “uniform-bunched” scheduling strategy. For days in which an average number of seats aboard airplanes are occupied, we recommend that airport A employ the “normal” scheduling strategy. We conclude that airport B requires 16 EDS machines and should employ the “uniform-bunched” scheduling strategy regardless of the number of people expected on a given day. The specific attributes of each flight scheduling system and a program for determining the flight schedule for a specific airport is included and discussed in the report.

This report also assesses the financial impact that implementation of ETD devices would have on airlines revenue. We conclude that if 20 % of all bags are screened with ETD devices, as is suggested, airlines would lose about 1.14 % of their current revenue. Based on this, we recommend that use of ETD devices be reserved for high risk situations, and that they should not replace EDS devices for security screening.

Lastly, we analyze the impact that future technologies could have on security screening. We draw conclusions about the benefits and limitations that each technology offers. Based on our evaluation, we recommend that the Transportation Security Administration (TSA) and the National Research Council (NRC) focus research efforts on X-ray diffraction and millimeter wave imaging detection technology. We conclude that X-ray diffraction technology offers the greatest potential benefit and that the possible benefit of millimeter wave imaging technology should be investigated. We do not recommend that the TSA and NRC focus research efforts in the field of airport security on neutron-based detection technologies, microwave imaging, or quadrupole resonance based detection technologies at this time. We conclude that the limitations of each technology does not justify the development of these technologies for baggage security.

An Overview of Past and Present Airline Security

An Introduction to the Issues and Development of Airline Security

There is no demonstration of the heights of human ingenuity more convincing than the view from a window of a soaring plane. The ability to travel quickly, inexpensively, and safely through the skies binds the people of the world together. Air travel is a way of life in today's world. But the terrorist attacks on September 11, 2001 proved that the skies can also be a means to death and destruction.

Airplanes are vulnerable to attack for the same reasons they are useful. Their importance to the commerce of nations, and the large numbers of people they transport makes a successful attack against a plane a powerful blow. In addition, only a small disruption to the structure of an aircraft can destroy the entire plane. Thus a terrorist can bring down a plane with a relatively small amount of explosives, allowing him (or her) to remain as inconspicuous to law enforcement as possible. The vulnerability of airplanes necessitates that security systems protecting airports are as effective as possible.

Realizing the severity of the threat, the U.S. government passed a number of laws to strengthen airport security. The 107th U.S. Congress passed, and President Bush signed into law, the Aviation and Transportation Security Act of 2001 (S. 1447). This bill created the Transportation Security Administration, a government entity responsible for determining and enforcing security requirements in airports. A critical component of Aviation and Transportation Security Act of 2001, Section 108, Paragraph D, specifies that all airports must have sufficient Explosive Detection Systems (EDS) to screen all checked baggage by December 31, 2002.¹

The aim of this legislation is to prevent a terrorist from checking an explosive device as baggage. One of the easiest ways for a terrorist to bring down a plane is for him (or her) to check an explosive on the plane and detonate it at some point in flight. In the 1980's this type of attack became almost an epidemic. The worst disaster caused by the explosion of checked baggage was the 1985 bombing of an Air India airline, which killed 329 people.² The bombing of Pan Am Flight 103 over Lockerbie, Scotland also involved an explosive checked aboard a plane. In response to the Lockerbie bombing, the U.S. government undertook several policy changes. Among these changes is the Aviation Security Improvement Act of 1990, which allocated funding and directed research which led to the development of EDS.³

Universal application of EDS is the most effective means of preventing explosives from being checked as baggage on the plane. EDS employ the same technology used in medical computer-assisted tomography (CT) scans. The EDS takes advantage of the fact that explosives tend to have higher densities than other organic materials. The EDS scans a checked bag with X-rays from many different angles to determine the density of the contents of the bag. A computer program then analyzes the results from scanning at different angles and uses them to create a three-dimensional representation of the bag. The three-dimensional representation prevents the detection system from being fooled by objects placed around an explosive. The EDS then applies a mathematical algorithm that matches the density characteristics of the objects in the bag with those of known explosives.^{4,5}

There are different types of EDSs by many different companies. But the most widely used by far is the CTX 5500 DS made by InVision. The CTX 5500 DS meets has an accuracy of about 98.5 % and a Federal Aviation Authority (FAA) certified throughput rate of 264-362 bags an

hour. The official throughput rate is determined in the most ideal of circumstances and does not take into account some delays that would occur during use in an airport.^{5, 6} Research indicates that the actual rate of baggage checking by the CTX 5500 DS is 160-210 bags an hour.⁶

Despite the effectiveness of EDS in detecting explosives, the cost of a device can be prohibitive to airports. The typical EDS costs about one million dollars, and costs \$9.13 an hour to operate.⁶ Considering the fragile financial status that U.S. airlines are presently in, any additional costs are undesirable. Another obstacle to meeting the goal of complete EDS coverage is the inability of manufacturers to produce sufficient EDS to supply airports.

To best allocate the limited number of EDS for the Midwest Region of the U.S., The Transportation Security Administration has requested that our analysis team from the Office of Security Operations develop a model to determine the number of EDS required to service the two largest airports in the Midwest Region, and to extend the model for the specific airports to a general model to determine the optimal number of EDS for any airport. In addition to this task, our report also outlines how the model could be adapted to incorporate the use of Explosive Trace Detection Systems (ETDs), devices that may become essential to airport security in the future. The economic impact of our models on airlines, and recommendation on who will pay for the scanning are also assessed. Finally, the research and development possibilities in the field of airport security are analyzed, and recommendations as to the best allocation of research effort and funding are addressed.

Security Objectives of Airlines and Constraints Airlines Must Operate Under

A Brief Position on the Objectives of Airlines in Security Screening

Airport security procedures exist for the benefit of the airlines in addition to the benefit of the passengers who travel on them. If terrorist attacks involving airplanes became a common phenomenon, the demand for air travel would be drastically diminished, and airlines would lose large amounts of money, as evidenced by what occurred in the aftermath of the September 11th terrorist attacks. It is thus the objective of the airlines to prevent all terrorist attacks on their aircraft.

However, airlines cannot attain a 100% certainty that their aircraft will not be targeted in a terrorist assault. Attempting to achieve such a level is futile. In addition, the logistics and economics of air travel place constraints on the intensity with which airlines can pursue security screening programs.

The delays that extensive security screening programs cause passengers boarding planes and the costs of security programs limit the extent of security procedures. Just as fewer people would fly on airplanes if there were inadequate security procedures, fewer people would fly on airplanes if the ticket cost were to double, or if they had to show up at the airport terminal four hours before takeoff time. Thus airlines must find a balance between the accuracy of security measures, the cost of their implementation, and the delay that such measures result in.

As long as airlines can guarantee a high accuracy of detection, terrorists will be discouraged from attempting attacks against airlines. If an airline can detect 95% of all bombs that might be checked on board its aircraft, terrorists will almost certainly not target that aircraft. Terrorist organizations are under too great scrutiny to be able to acquire the materials for and produce twenty explosive devices in the hopes that an average of one of them will find its way on board an aircraft undetected. The discovery of a single attempt to smuggle an explosive on board an aircraft alerts authorities to the activity of a terrorist cell, and will likely lead to the arrest of the cell.

Airlines can give about a 98.5% certainty that a checked bag does not contain explosives by scanning it with an EDS. This should be a sufficient certainty to deter attempts to smuggle explosive devices on board aircraft from all but the most desperate or foolish of terrorist cells.

Scanning all baggage with EDSs requires analysis of the costs of such scanning, and a model to determine the best scheduling for flights so that the delay caused by scanning is kept to a minimum. The following models determine the best systems of scheduling and acquisition of devices to minimize the constraints of time delays and cost.

The Math Behind the Model

Queueing Theory: Reading Between the Lines

Queueing theory can mathematically simulate the flow of a line of customers at a server, given sufficient data. The basic shorthand for a model portrays the three most important figures of a queueing simulation: the probability distribution of the interarrival time between two arrivals of customers, the probability distribution of the time it takes to serve a customer, and the number of servers.

For our model, the customers translate to pieces of checked luggage and the servers to EDS machines. The queue represents the backup of luggage and at first sight queueing theory seemed to have formulas that could find the maximum queue length. This would stand for the maximum backup of a luggage for a specific flight, and the calculation of delay from this longest line length would be relatively easy.

After downloading a computer program that simulated queueing theory problems, we realized there existed aspects of the problem which could be demonstrated by queueing theory.¹⁹ The main two aspects are the batching of bags on people and the variability of server speed which would utilize the 160 to 210 bags per hour range that we had previously been estimating as 185 bags per hour.

However, we later ran into difficulties which appeared to be insurmountable with our current knowledge of queueing theory. One part of a queueing theory model is the probability distribution of the time between arrivals of customers, or for our problem, bags. That this distribution is assumed to be static with time leads to the problems with this model. The equations of queueing theory assume that this distribution is constant. In other words, the times between arrivals can be random, but the process that generates these times cannot change during the simulation. This assumption ignores a vital part of our airport model: that there are peak hours. The average time between arrivals of passengers is much longer at two hours before a flight than at one hour beforehand, but with a queueing theory model is could not be taken into account. The second problem with our adaptation of queueing theory does not prioritize customers, which turned out to be the key for our study.

Passenger Arrival Distribution

The passenger arrival distribution describes the spread of time during which passengers arrive for a flight. For each plane, the distribution for its passengers is a function of time that gives the rate of its passengers per minute that arrive at the airport. To find the distribution of the total number of passengers for any number of flights, we can sum the functions for the individual planes, making sure to offset the time by the difference of the time of takeoff. Therefore, the choice of this random variable will directly imply the calculated rate of luggage arrival to the EDS machines.

Our choice of the type of probability distribution for the passenger arrival rate of individual planes was the normal or Gaussian distribution. The main reason for this is that it matches previously taken data quite effectively.⁷ However, there are several implications of this choice. First, the parameters of the curve are simply the mean, μ , and the standard deviation, σ , so with a few estimates of when the travelers arrive we are able to generate an accurate and continuous

$$\int_{t_0}^{t_1} f(t)dt$$

approximating curve. Also, a few inconveniences come with the selection of the normal distribution. The function evaluates the rate of passengers arriving at any time. By the fundamental theorem of calculus, to determine the number of passengers who arrive during a period of time, the integral from the beginning to the end of that time will be the number of passenger that arrived during the period. So, if $f(t)$ is the passenger arrival rate, the number of passengers that arrive between t_0 and t_1 is:

The reason this is problematic comes from the form of the Gaussian distribution. The function of the normal distribution with average μ and standard deviation σ is:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x-\mu)^2}{\sigma^2}}$$

So to find the number of passengers, we would find the integral of this function. But because of the squared term in the exponential, this integral would have no symbolic answer. This means that the only way to determine the result is to approximate it with numerical methods. Due to this, we decided to drop the actual equation of the distribution and instead only use the formula to generate a set of data points, one per minute, that describes the average number of passengers to arrive during that minute. The additional reasons to change from continuous to point-wise and the other implications of the Gaussian distribution are discussed below.

The two parameters necessary to fully describe a normal distribution are its mean and its standard deviation. The mean represents the expected value of the outcome, in this case the arrival time of the typical passenger. Given that the typical arrival time falls between forty-five minutes and two hours before departure, the expected value of the time should be the average of these two extremes, since the normal distribution is symmetric about its mean. This does not correlate exactly with the data found, which determines the mean to be closer to an hour.⁷ But this is to be expected, since this is data from 1989, it does not take into account the changes invoked because of the events on September 11th, 2001. The heightened screening has initiated carry-on and passenger searches that are more thorough and therefore more lengthy. This in turn has caused airports to suggest that passengers arrive ninety minutes prior to takeoff. Assuming passengers are considering this information, the average time between arrival and takeoff should have increased towards ninety minutes, a confirmation of our average of 82.5 minutes. It should be noted that the previous conclusions drawn from this data does not depend on these alterations. The type of distribution would certainly remain a normal, regardless of the specific average time.

The standard deviation of a distribution represents the amount that the data varies, and the probability that a reading is a given distance from the mean. In our model, the higher this variance is the more passengers there are that arrive early or late. To match the bounds of two hours and forty-five minutes, a standard deviation of 18.75 minutes would put two standard deviations, or 95.44% of the passengers within this period. We found that this also is within one

minute of the deviation characteristic of previous data once the mean had been rescaled from an hour to 82.5 minutes.

With these two parameters, the rate of arrival of passengers for each plane is:

$$f(t) = \frac{N}{\sqrt{2\pi}(18.75 \text{ min})} \cdot e^{-\frac{(t+82.5 \text{ min})^2}{(18.75 \text{ min})^2}}$$

where N is the total number of passengers that board the plane and time is measured in minutes with $t = 0$ being takeoff. This formula still has a few small problems. Entering any positive amount of time will output a positive probability. This would represent the probability that a traveler *who fills a seat* arrives at the airport after the flight leaves. Similarly, there is a positive probability that a person arrives a day before his flight leaves. To eliminate these ridiculous possibilities, we changed the passenger arrival rate function to:

$$f(t) = \begin{cases} \frac{a \cdot N}{\sqrt{2\pi}(18.75 \text{ min})} \cdot e^{-\frac{(t+82.5 \text{ min})^2}{2(18.75 \text{ min})^2}}, & -120 \text{ min} \leq t \leq -30 \text{ min} \\ 0, & t < -120 \text{ min} \text{ or } t > -30 \text{ min} \end{cases}$$

Where a is the renormalization factor that makes the total integral of the rate equal N , so both the shape of the distribution and the total number of passengers is the same while fixing the problem of positive probabilities occurring where zero probabilities should.

The bounds of two hours prior to takeoff and thirty minutes before takeoff were chosen for the following reasons. By the data found people do arrive earlier than two hours before time of departure. However, by our model and method of prioritizing the luggage by flight schedule, whether a bag arrives this early or at two hours is inconsequential.⁷ A bag that arrives extremely early will either appear before or after when backup occurs. If it arrives before the luggage begins to pile up, then its arrival does not slow down the system at all because the EDS machines are not yet running at full capacity, so it can be processed without creating backup. If it arrives while the bags are amassing, then it is ordered in the line according to its flight. Since its flight is in over two hours, the piece of luggage will be near the back of the queue, so it will not be scanned in a short amount of time. Therefore, it makes no difference that the bag arrived slightly earlier than two hours or slightly later. Lastly, the assumption that no bag arrives earlier than two hours prior to embarking is a hindrance, so any slight miscalculation would be on the side of safety, since the more spread out the peak-hour luggage is the easier it is to deal with.

The other bound which requires all passengers arrive with thirty minutes before takeoff is also derived from logistical facts. Many airlines will not accept people at the check-in desk that arrive less than thirty minutes before the flight leaves, especially those with checked luggage. Also, the airport security associated with the passengers and carry-on luggage has become so exhaustive that thirty minutes is often not enough time to get to the plane's gate. Lastly, the constant length of time that we are using to judge how long luggage is being moved through the airport, which will be described in its entirety below, is twenty minutes. This implies that

accepting baggage with much less than thirty minutes until departure changes the reason of delay from the amount of backup to the time of the arrival of the last piece of baggage, which would be detrimental to the number of delayed flights.

Now we have the average rate of arrival for each individual flight. For the rest of the calculations we used, instead of this continuous function, a set of discrete data points. These data were generated by simply finding the value of the function at each minute, on the minute. Now to find the rate of passengers arriving for a certain flight per minute at time t_1 we had to find the entry in the appropriate table of the excel file (found in the appendix). Note that each minute, we have the average number of passengers that arrive during that minute, since the rate is measured in people per minute. This was done for the sole reason of ease in calculation. Since our points were one per minute, integrals involving the unintegrable Gaussian distribution function turned into summations of reasonably short lists of real numbers. This allowed easier analysis of the lists by our computer program.

The lack of precision incurred was trivial for a few reasons. The integrals (though they are computationally summations, our notation will refer to them as integrals because of notation preferences) that are approximated by these summations can also be represented by the area under the passenger arrival rate curve. The geometrical interpretation of the summation estimating of this area is exactly the way a Riemann Sum would compute the integral. Since the important integrals that determine the delay of the planes have bounds whose separation is between one and two and a half hours, by taking minute intervals, we are splitting the integral up into between 60 and 150 parts, which yields very accurate results. Also, the symmetry of the normal distribution about the mean allows for arguments that reduce the possible error of such summations; since the integrals that will be taken will span the mean, a slight loss on one side of the average will be cancelled by a similar gain on the other. Finally, takeoff schedules are never more strict than by-the-minute. Later analysis showed that precision of the takeoff times to more than one minute had an insignificant impact on the results of the model.

The Quest for a Schedule:

Making a Model out of a Mountain of Math

We considered several ways to generate a flight schedule from the available data. The ideas most carefully analyzed were:

- Writing a program to use brute force to generate all possible schedules and find the best one.
- Distributing the number of planes uniformly in time and planes of the same size uniformly in time. We will refer to this as “Schedule UU”.
- Distributing the number of planes uniformly in time and skewing the takeoff time of the planes with more passengers towards either side of the peak hour. We will refer to this as “Schedule US”.
- Skewing the takeoff times of the planes towards the beginning and end of the peak hour and skewing the takeoff order such that planes with more seats takeoff closer to the sides of the peak hour. We will refer to this as “Schedule SS”.

We quickly eliminated the first option, as the number of computations required would be too great for the time allotted. In addition, a general model based on brute force would be difficult to specify, and would give airports little idea of what their schedules would look like on a given day until they ran the program.

The other options all seemed promising. We first developed a computer program in C++ to generate a flight schedule that was as uniformly distributed as possible and kept large flights as far from each other as possible. We used Excel to store the specific data regarding the distribution of passenger arrival and size of planes and fed that data into the C++ program. We also used Excel to output the data generated by the C++ program, allowing us to use Excel’s built in graphing capabilities to analyze the results of our schedule. This schedule and the other two we designed are shown below.

This program is not computationally intensive, and functions by first determining the minutes between each takeoff required to get each plane type off the ground. The program then places each plane on the takeoff schedule at evenly spaced minute intervals. The program then analyzes locations in which it has placed two planes departing at the same time, and separates them by about a minute. To keep larger flights as far from each other as possible, the program also considers the largest flight scheduled in the minutes immediately prior to and following a flight. When the program separates two flights that depart at the same time, it places the larger of the two flights farthest from the last or next instance of the takeoff of a large flight.

Then we created another C++ program to create a graph of the rate of passenger arrival that results from this schedule. Using the schedule generated by the first program and the data regarding normal distribution stored in an excel file, we assigned each flight a distribution of people arriving at different times. After we ran the program, we graphed the distributions resulting from the scheduling strategies for both airports A and B, shown below:

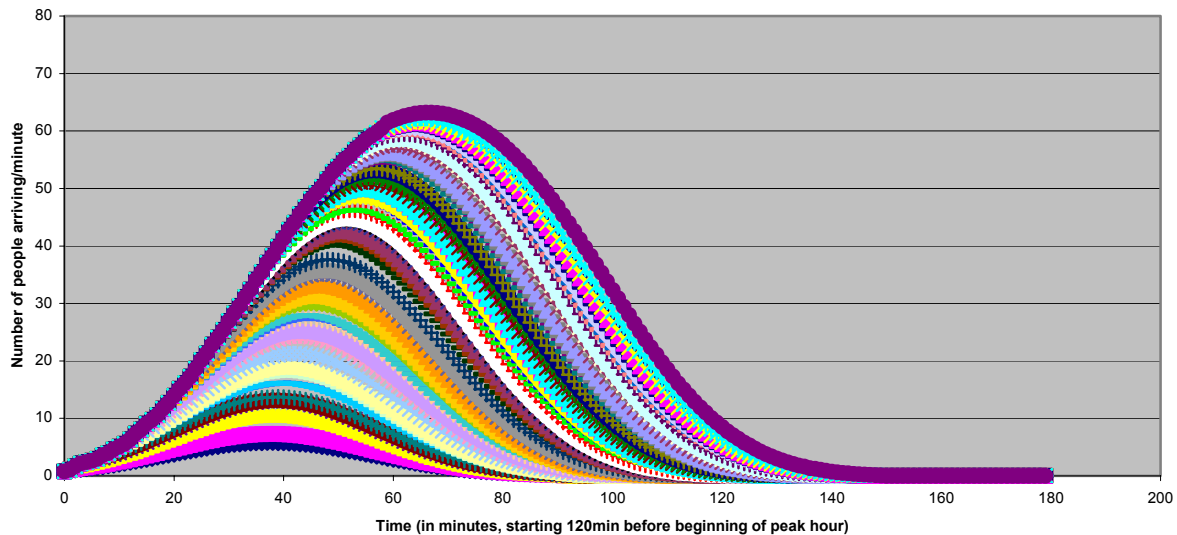
Considered Flight Schedules

Plane Type	Total Number of Seats
A	34
B	46
C	85
D	128
E	142
F	194
G	215
H	350

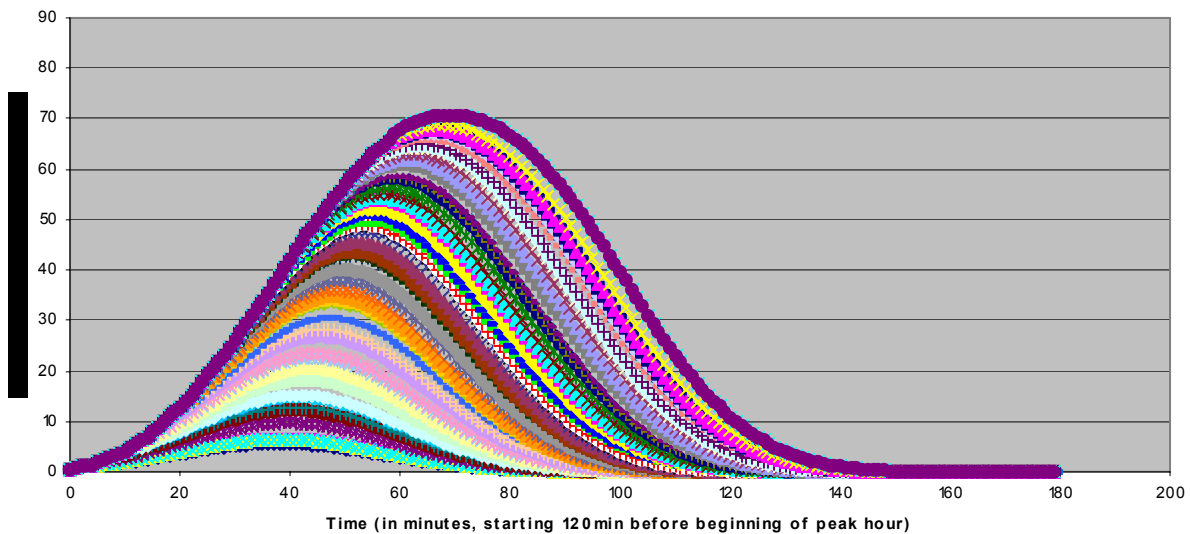
Time (mins since beginning of peak hour)	Plane Type Airport A			Airport B		
	UU Schedule	US Schedule	SS Schedule	UU Schedule	US Schedule	SS Schedule
0	H	H	H	F	H	H
1	C	F	F	D	G	G
2	F	F	F	B	F	F
3	E	F	F		F	F
4			E			F
5		E	E	F	F	F
6	A	E	E	E	F	F
7	E	E	E	A	F	E
8			E	C	E	E
9	E	E	E			E
10		E	E	B	E	E
11	A	E	E	F	E	D
12	E		E	E	E	D
13	F	E	D	D	D	D
14	B	E	D	A		C
15	E	E	D	G	D	C
16		D	B	C	D	C
17	A		B	E	C	B
18	E	D	A	F	C	B
19	C	C	A			B
20	D	B	A	B	C	A
21	E		A	A	B	A
22		B	A	C	B	A
23	A	A		E	B	A
24	E	A		D		
25	F			F	A	
26		A			A	
27	E	A			A	
28		A		A	A	
29	B			B		
30	E	A		E	A	
31	A	A		F	A	
32		A		C	A	

33	E	A		A
34				
35	F	A		A B
36	A	B		D B A
37	E	B	A	E B A
38			A	F C A
39	E	C	A	B A
40	C	C	A	C C B
41	D	D	A	E C B
42	A		B	F C B
43	E	E	B	A D C
44	B	E	D	
45	E	E	D	G D C
46		E	D	
47	A		E	C E D
48	F	E	E	D E D
49	E	E	E	E E
50		E	E	F E E
51	E		E	A E E
52		E	E	B F E
53	E	E	E	E F E
54	A	E	E	F F
55	E		E	
56	A	F	E	C F F
57	B	F	F	A F F
58	D	G	F	F G F
59	G		G	H G

Arrival Distribution as a Function of Time
Airport A, Avg Number Seats Filled, UU Schedule
 For flights taking off by minute 0, by minute 1, by minute 2, ..., by minute 59



Arrival Distribution as a Function of Time
Airport B, Avg Number Seats Filled, UU Schedule
 For flights taking off by minute 0, by minute 1, by minute 2, ..., by minute 59

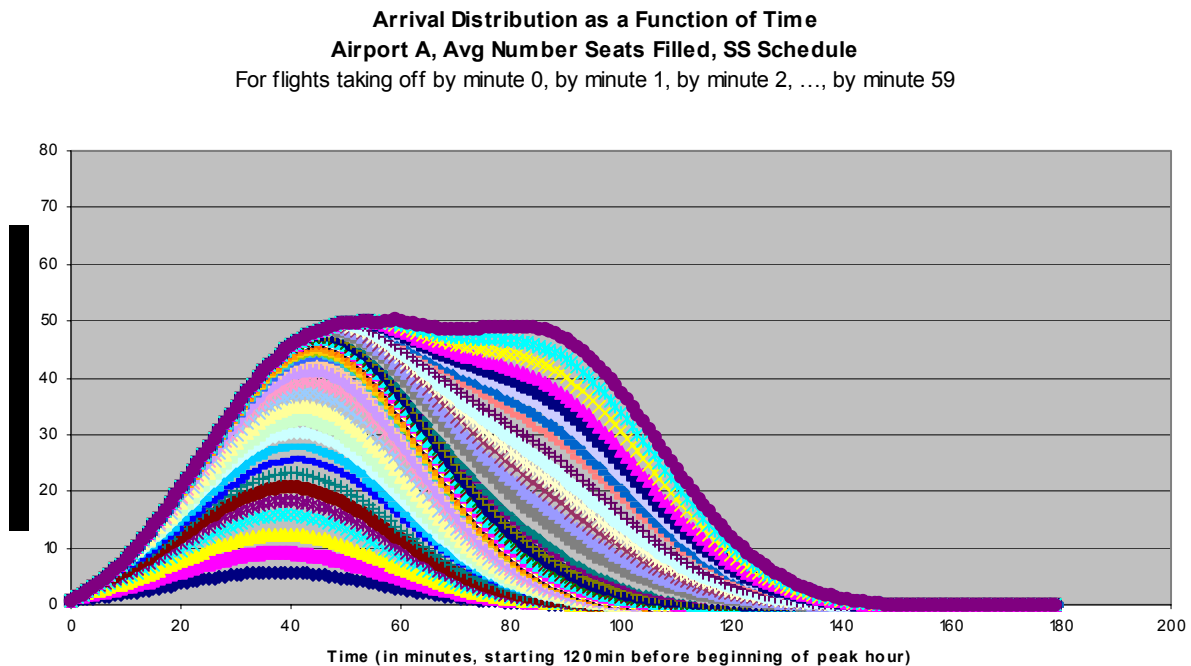


In these graphs, each differently colored distribution represents the arrival distribution for a different departing flight and the highest distribution, represented the total distribution for the overall arrival rate of people into the airport. Because of the resemblance of this graph to the normal distribution, we named this scheduling plan the “normal” scheduling plan.

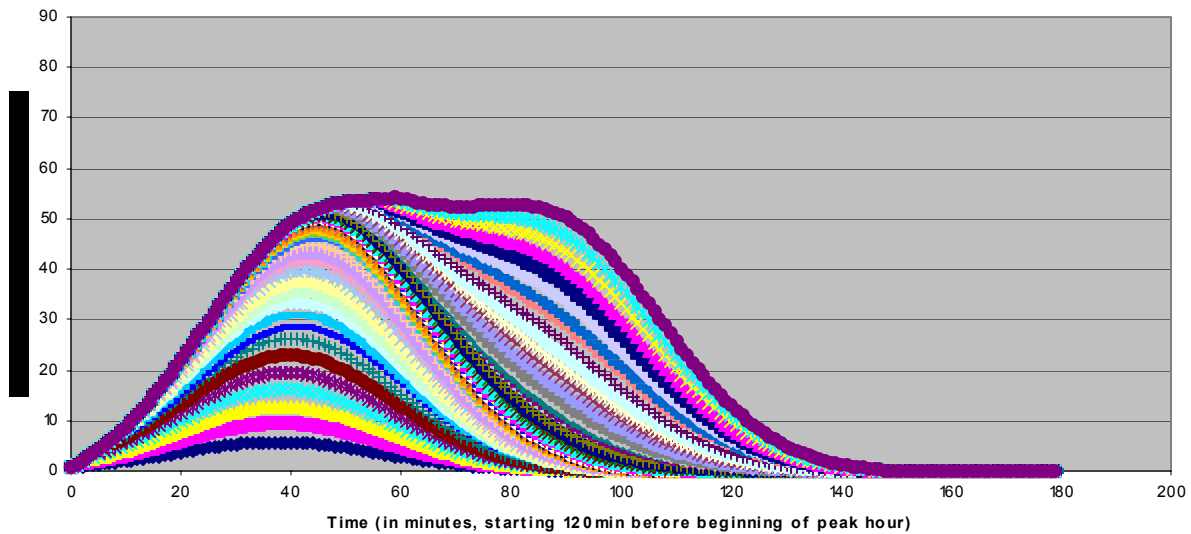
However, we were initially disappointed to see the resemblance of this graph to the normal distribution. It was our reasoning that delays would be minimized if the arrival rate of people is

as uniform as possible. As a result, we developed the other scheduling systems to see whether they would achieve a more constant arrival rate.

The next scheduling strategy we analyzed was to bunch all departing flights at the beginning and the end of the peak hour. We reasoned that given the resemblance to the normal curve of the first scheduling strategy, moving the flights to the ends would decrease the maximum rate of passenger arrival and achieve a more constant arrival rate. We then wrote a C++ program that would create a schedule that separates the flights into two groups of as close to equal size as possible. We specified that the program place the largest flights at the ends of the peak hour to further spread out the arrival rate of people. After running the program and sending the resulting data to Excel, the following graph was generated.



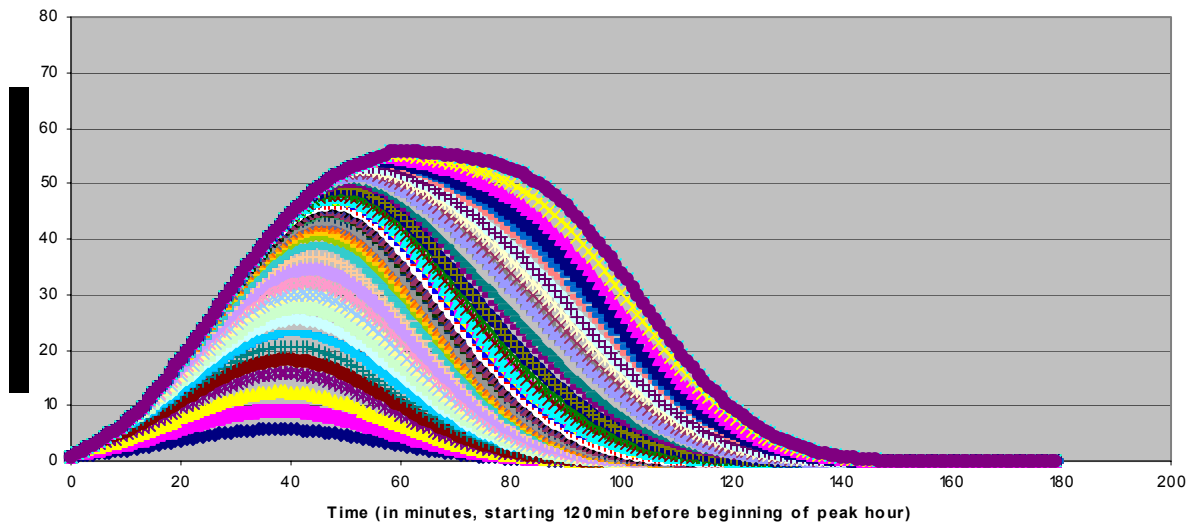
Arrival Distribution as a Function of Time
Airport B, Avg Number Seats Filled, SS Schedule
 For flights taking off by minute 0, by minute 1, by minute 2, ..., by minute 59



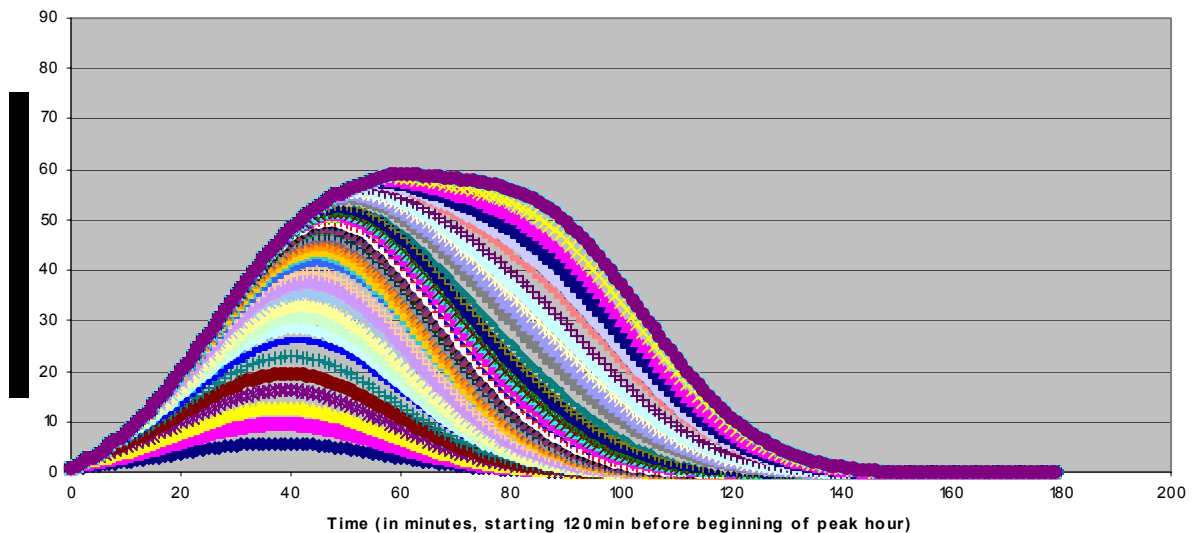
We were pleased to see that this approach spread the arrival rate of passengers out more uniformly. In particular, we were impressed that the SS distribution reduced the maximum passenger arrival rate by about 15 passengers per minute. We believed that a lower maximum arrival rate would reduce the baggage processing rate required by the EDS devices, and thus decrease the required number of EDS devices.

However, we were hopeful that we could generate a graph with an even more constant arrival rate by combining the two strategies. To do this, we wrote a program that uniformly spaced all the smaller planes (planes with 85 and fewer passengers) in a similar way that the first program uniformly spaced all flights. We then specified that the program treat all larger planes as it did in the way that the second program bunched flights. After running the program and graphing the data in Excel, we generated this graph:

Arrival Distribution as a Function of Time
Airport A, Avg Number Seats Filled, US Schedule
 For flights taking off by minute 0, by minute 1, by minute 2, ..., by minute 59



Arrival Distribution as a Function of Time
Airport B, Avg Number Seats Filled, US Schedule
 For flights taking off by minute 0, by minute 1, by minute 2, ..., by minute 59



This distribution turned out to have properties in between the UU and SS scheduling distributions. The maximum arrival rate of passengers is in between the maxima found in the UU and SS graphs. Although this distribution more closely resembles the UU curve, its standard deviation is greater than the standard deviation found in the UU scheduling distribution.

Determining Delay Time:

Now that we had a few schedules to choose from, we figured that more computation would lead to a final optimal selection of flight times for each airport, as well as a general method to create the best schedule for any given airport's peak hour statistics. The best way to measure the schedules against each other is to find which ones cause delays and to gauge the disturbance produced by each.

The delay of a flight due to scheduling is caused by the checked luggage of the passengers on that flight not being in the storage compartment of the plane at takeoff. This may be the effects of three things:

- an earlier backup of luggage at the EDS machines
- an unexpectedly high number of people on a flight
- a failure of an EDS machine.

All of these possibilities can be calculated and accounted for in our model.

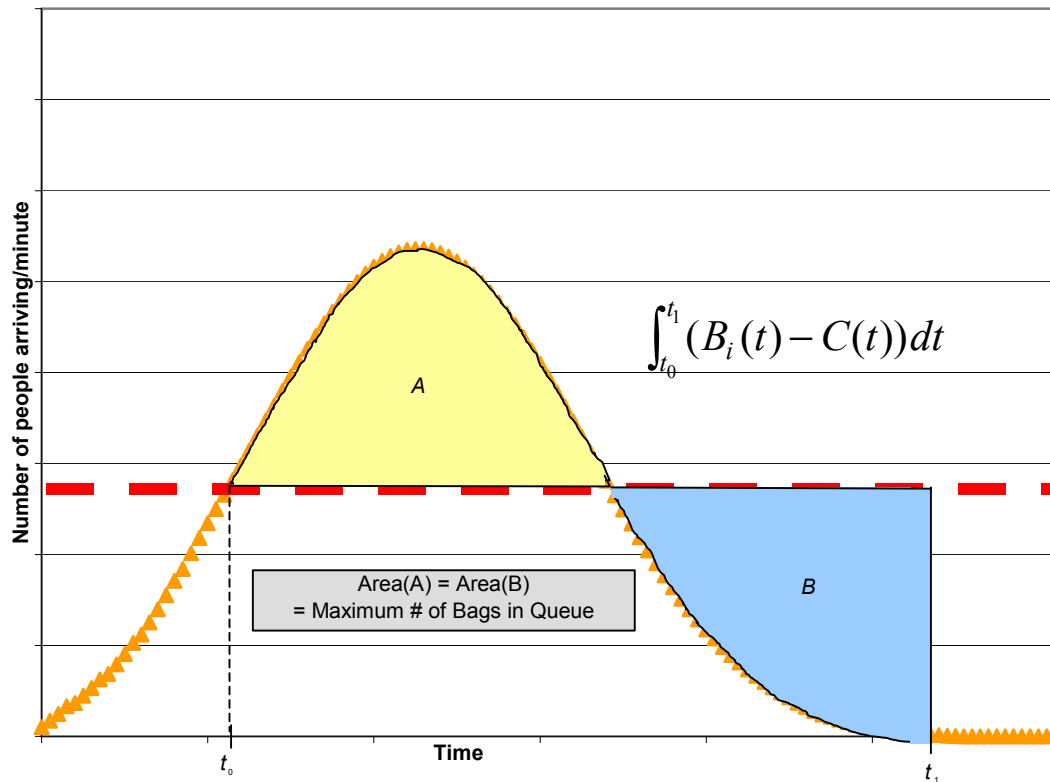
First, the backup caused by earlier flights can amass until there are not enough machines to process the luggage by departure. This delay is caused directly by the schedule. The first measure in preventing this setback is to prioritize the luggage.

All the baggage that comes to the EDS machines has a tag identifying its plane and the takeoff time of that plane. Once the bags begin to pile up, a queue is formed. During approximately the first hour of backup, the bags are coming in at a faster rate than the machines can examine them, so the line grows. As this happens, the luggage must be sorted either by hand or automated according to the priority of time of takeoff. This is essential to the minimization of EDSs while avoiding delays. The luggage that needs to go with a certain flight must skip ahead of all the bags waiting to be scanned that are associated with later flights.

This priority allows the calculation of the delay of each plane individually. Let f_i be the i^{th} flight. Let the function $B_i(t)$ be the sum of passenger arrival rates of the first i flights. This measures the rate of flyers on the first i planes coming to the airport. So if $C(t)$ is the maximum rate at which passengers' bags can be dealt with, the backup of bags for flight f_i begins at t_0 , where $B_i(t_0) = C(t_0)$. This pileup continues to accumulate so long as $B_i(t)$ is greater than $C(t)$. To calculate the amount of backup left at time t_1 , the integral under $B_i(t)$ and above $C(t)$ from t_0 to t_1 . The integral doesn't start at the beginning of the period because there is no backup then, so no queue has been formed. The time at which the queue for the baggage going on flights f_1, f_2, \dots, f_i is dealt with is the point at which the integral of $B_i(t) - C(t)$ is zero.

Arrival Distribution as a Function of Time

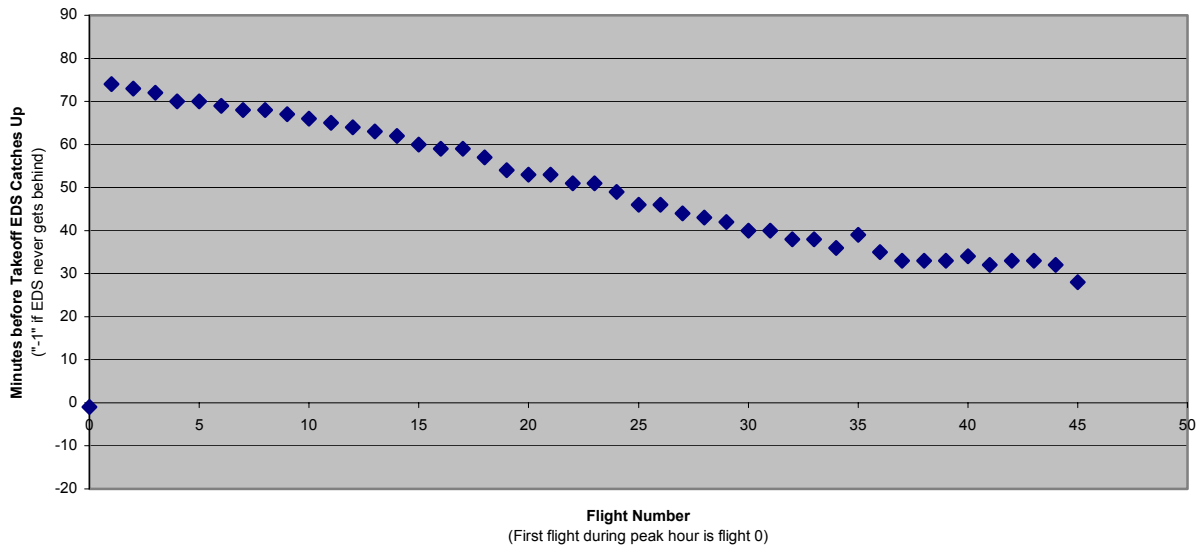
Calculation of Time EDS Luggage Backup Cleared



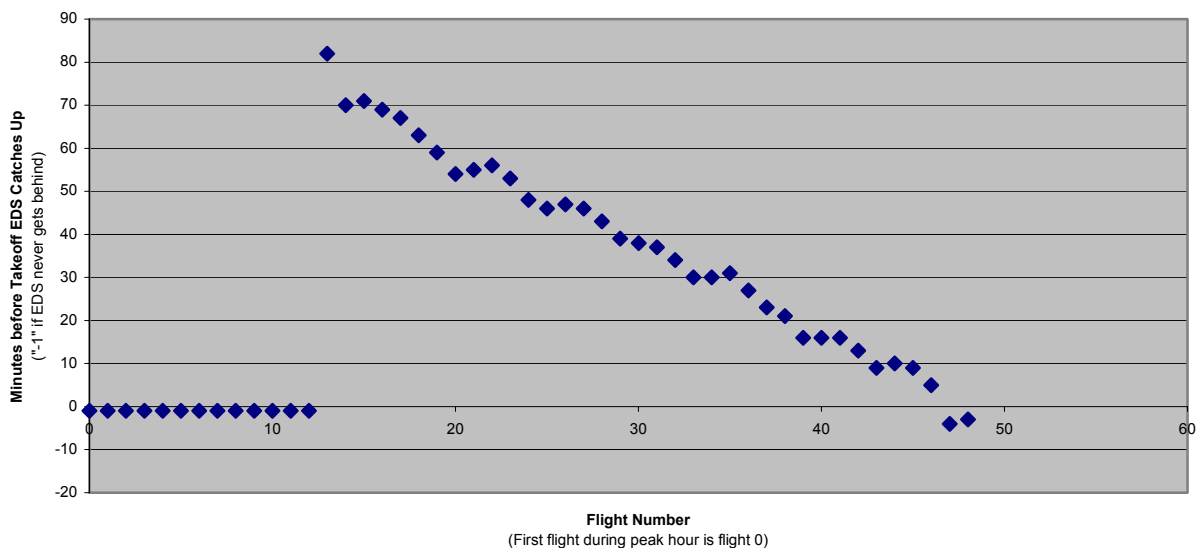
Since this can be easily integrated and the functions $B_i(t)$ simply calculated by summing the individual flight distributions, it wasn't difficult to write a C++ program to determine what effects the prioritization of luggage by takeoff time would have on the delays of each flight. The program sums the total number of passengers' baggage waiting to be scanned for each flight and calculates the time before or after takeoff required for the EDS machines to scan all of the baggage for each flight. We used the data produced by the previously written programs that determined the flights distributions under the three different scheduling strategies to generate data regarding the time that each flight in the distributions will be delayed. Because we were only interested in assessing the effectiveness of the schedules at this point, we assumed a constant rate of baggage checking by the EDS systems sufficient to check the baggage of 30 people each minute.

For the UU airline scheduling strategy, the program generated the graphs shown below, with positive delay time meaning the time between which the bags finish being delayed and takeoff:

**EDS Catchup with Incoming Baggage Times
Airport A, Avg Number Seats Filled, UU Schedule
EDSs at 30 people/minute**

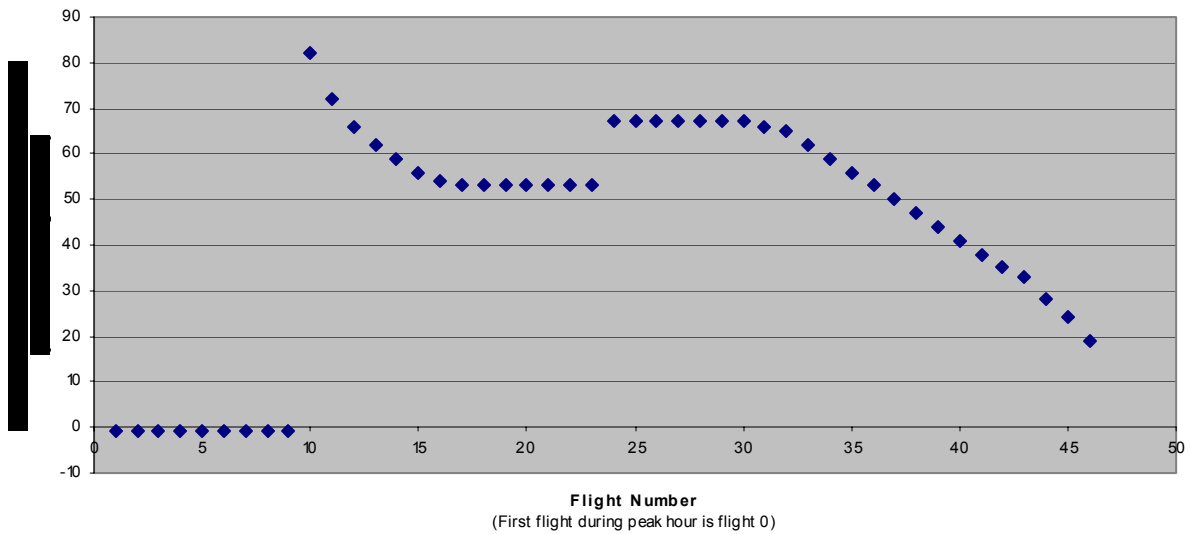


**EDS Catchup with Incoming Baggage Times
Airport B, Avg Number Seats Filled, UU Schedule
EDSs at 30 people/minute**

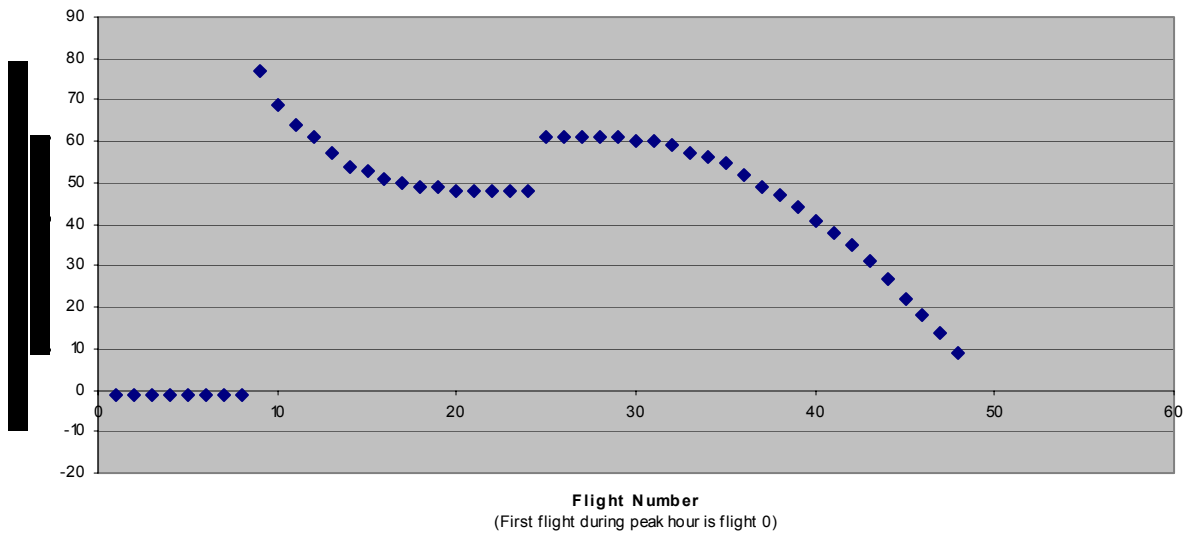


When the data generated for the SS flight distributions was applied to the same program, the following graphs were generated:

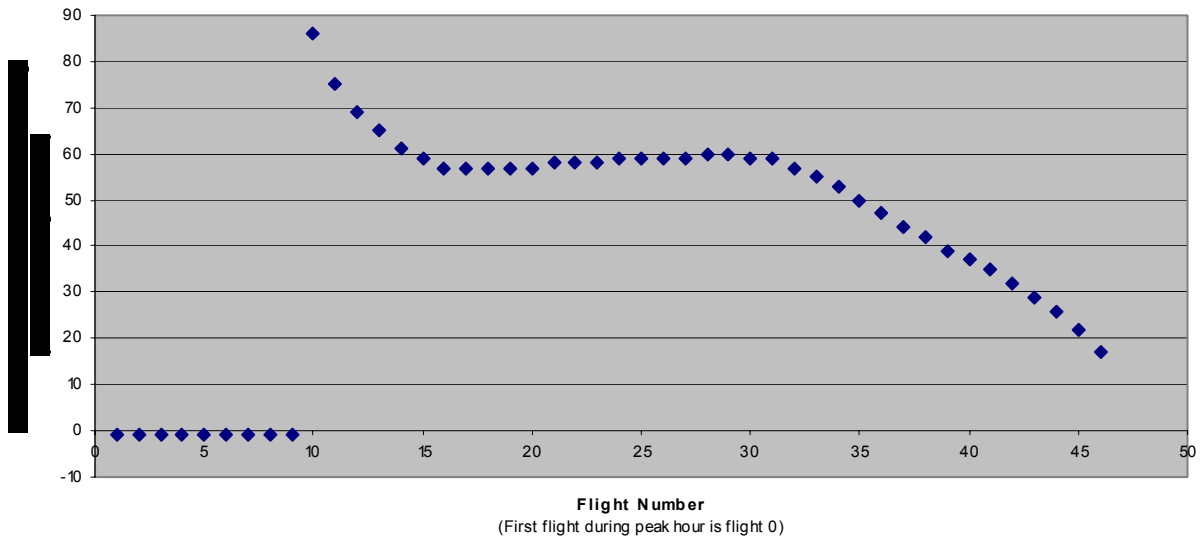
EDS Catchup with Incoming Baggage Times
Airport A, Avg Number Seats Filled, SS Schedule
EDSs at 30 people/minute



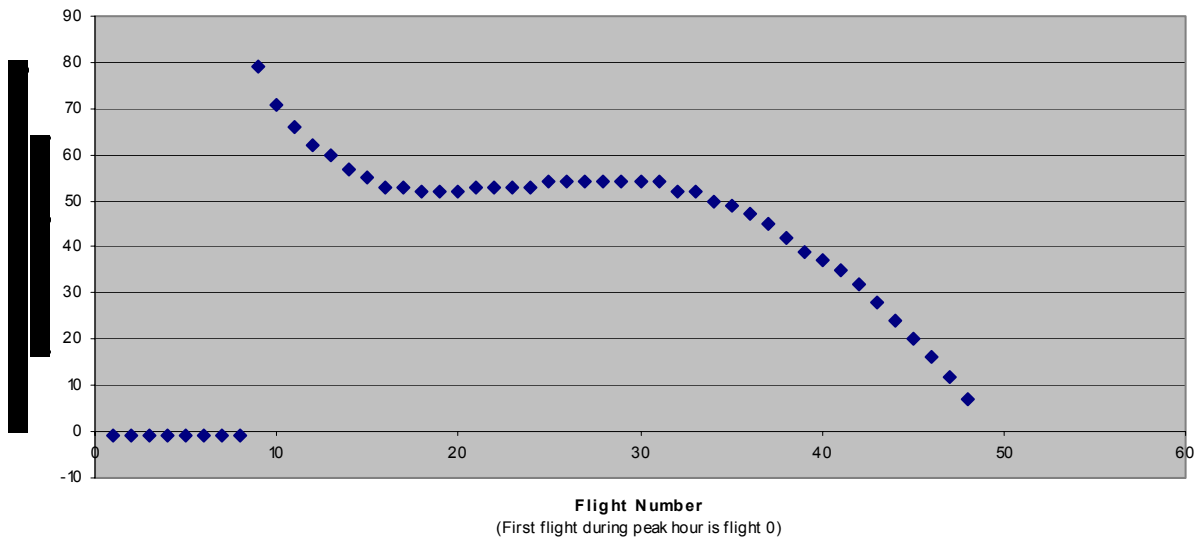
EDS Catchup with Incoming Baggage Times
Airport B, Avg Number Seats Filled, SS Schedule
EDSs at 30 people/minute



EDS Catchup with Incoming Baggage Times
Airport A, Avg Number Seats Filled, US Schedule
EDSs at 30 people/minute



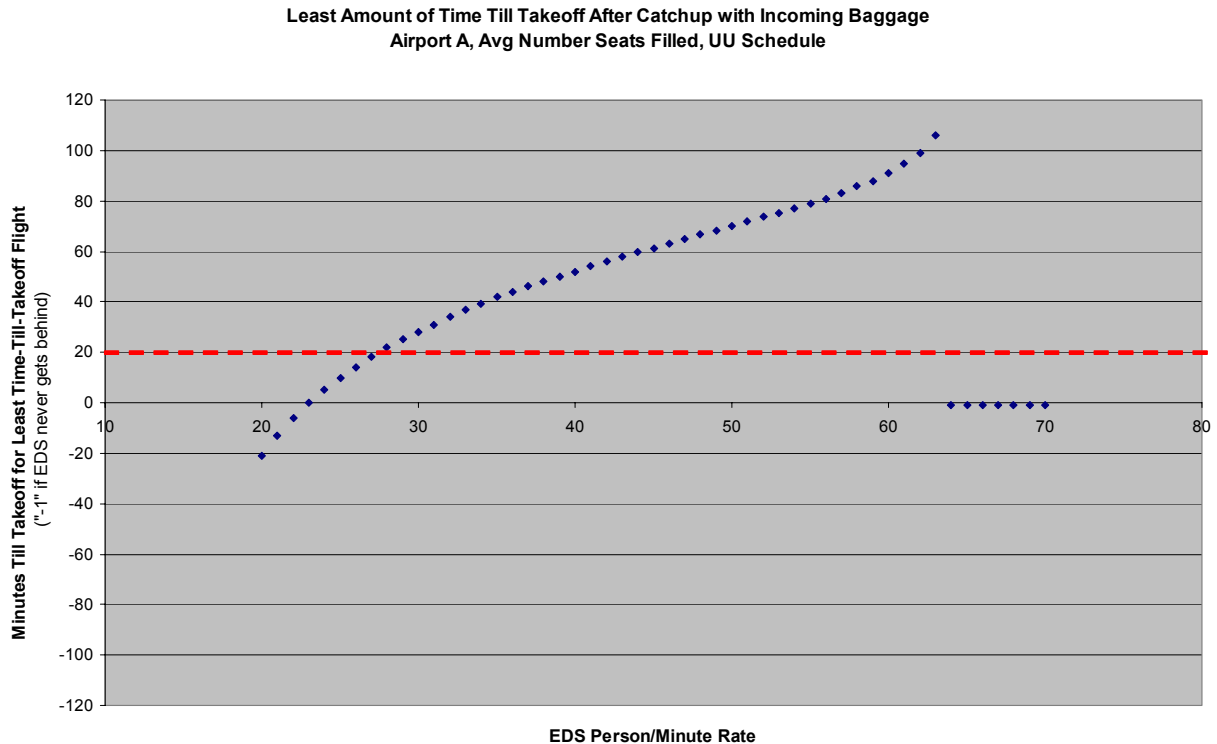
EDS Catchup with Incoming Baggage Times
Airport B, Avg Number Seats Filled, US Schedule
EDSs at 30 people/minute



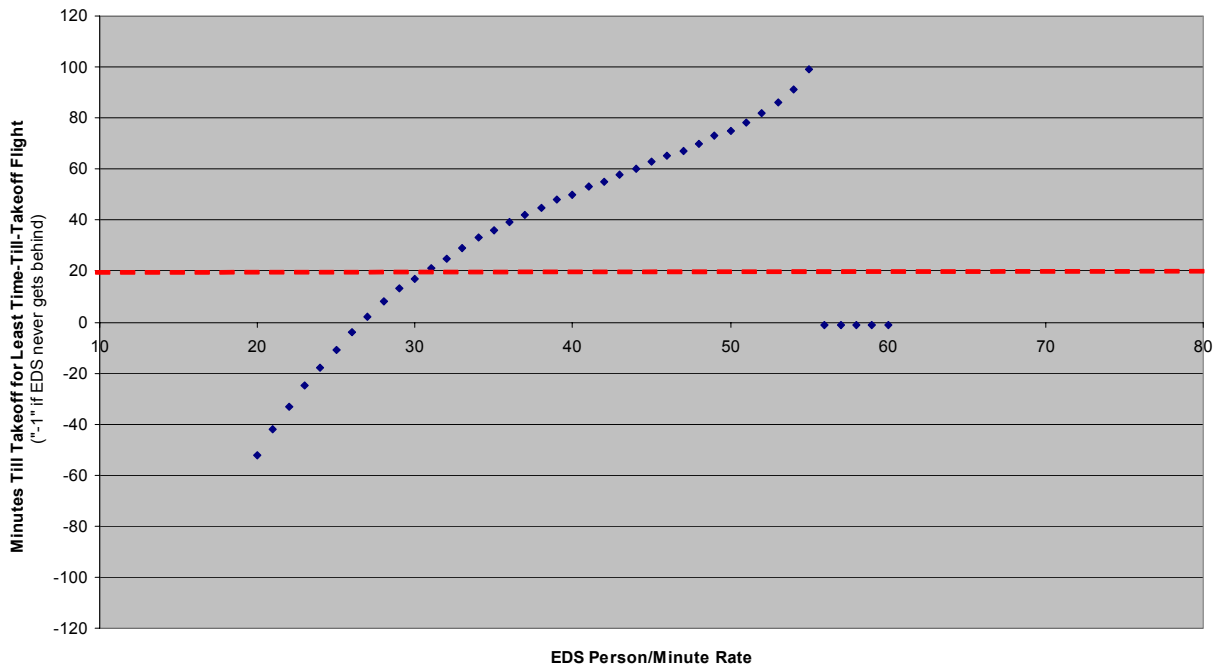
And similarly, when the data generated for the US flight distributions was applied to the same program, the following graphs were generated:

The difficulty arises with our selection of 30 passengers per minute. This was an arbitrarily chosen rate, and inputting another speed of processing would result in a different graph. For this program to help us decide what rate of processing was optimal, we realized there needed to be a

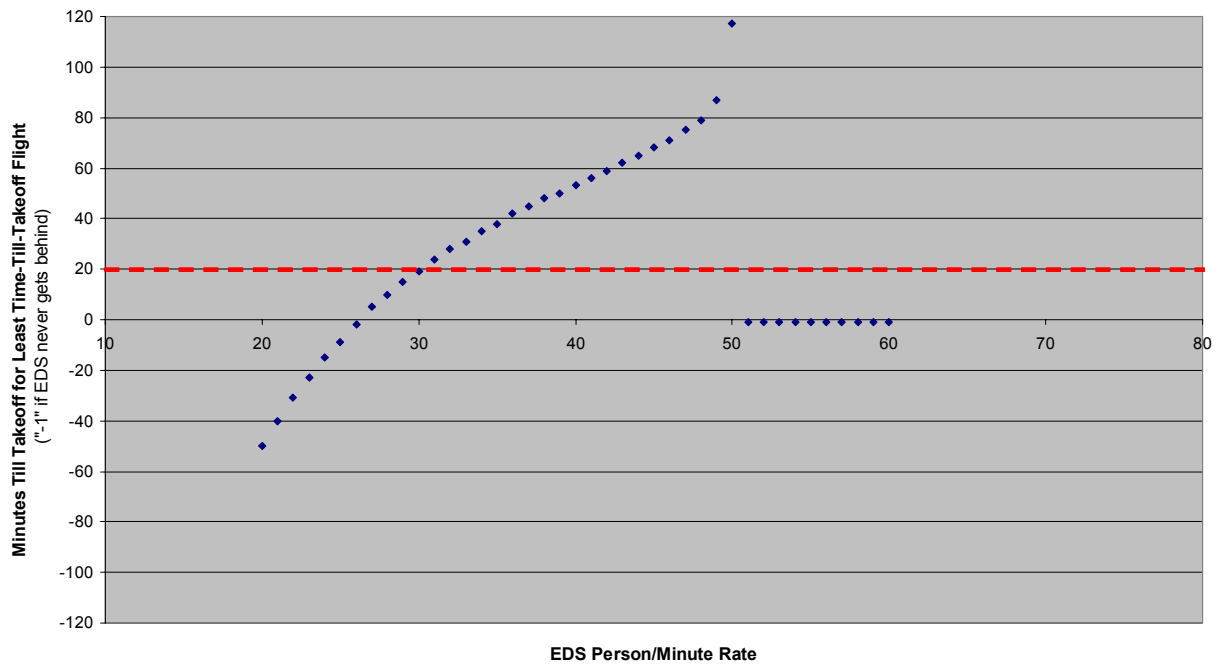
way to analyze the amount of delay of a rate with only one number. The amount of the worst delay of any plane (which is the equivalent of the last plane's delay for all of our models) seemed to be a good measure of the setback that any rate gives us, so we wrote another C++ program that would take a schedule and input processing rates between twenty and seventy passengers per minute and find the minimum y-value for each of the fifty graphs that this generated, which it would output as the worst delay. The resulting graphs are shown below (a lower y value is a longer delay):

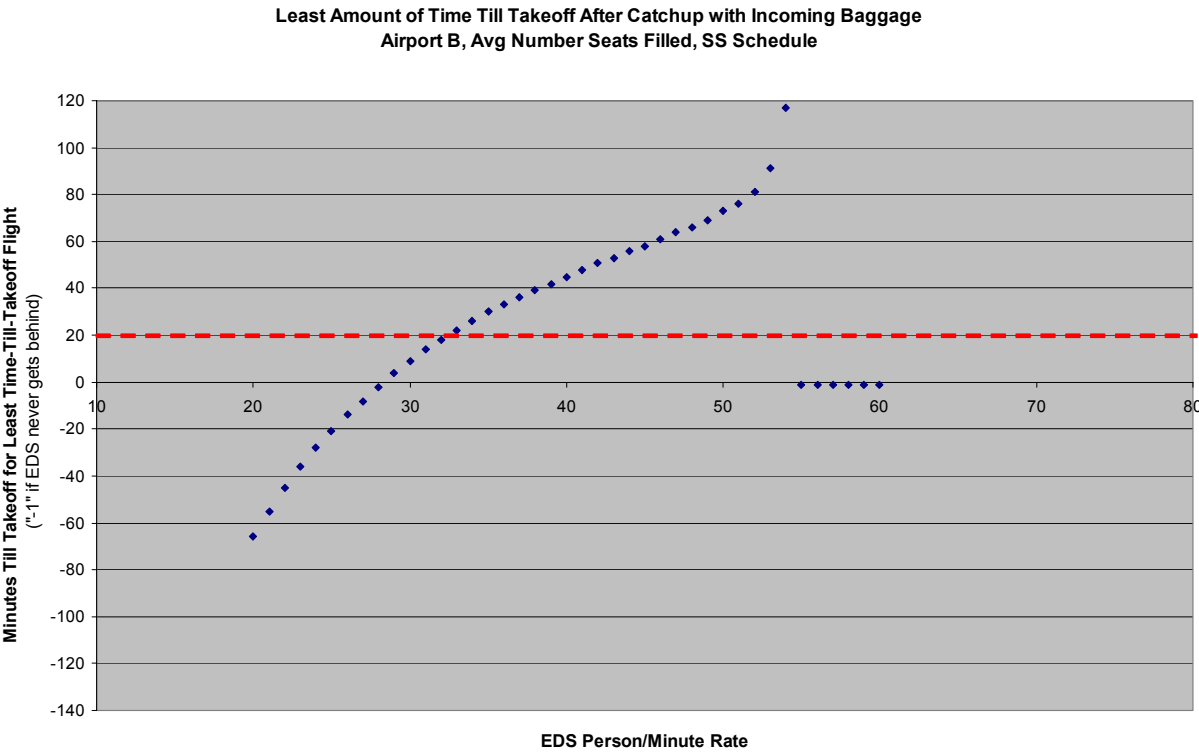
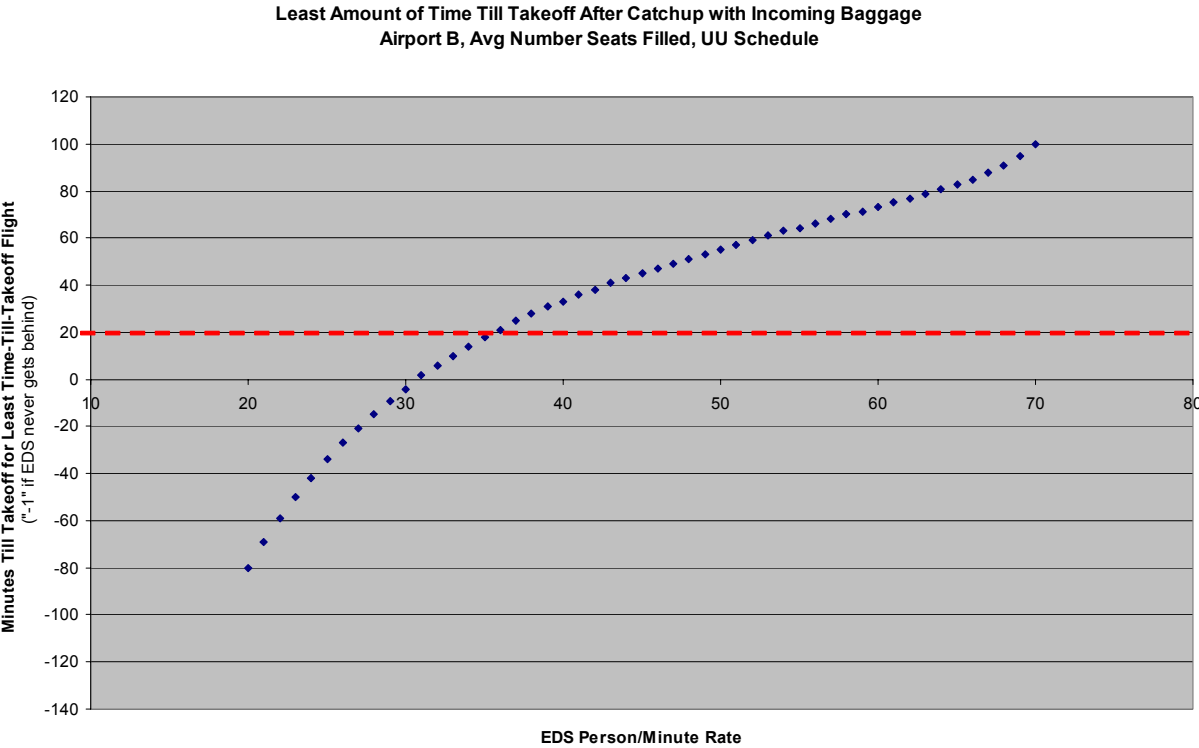


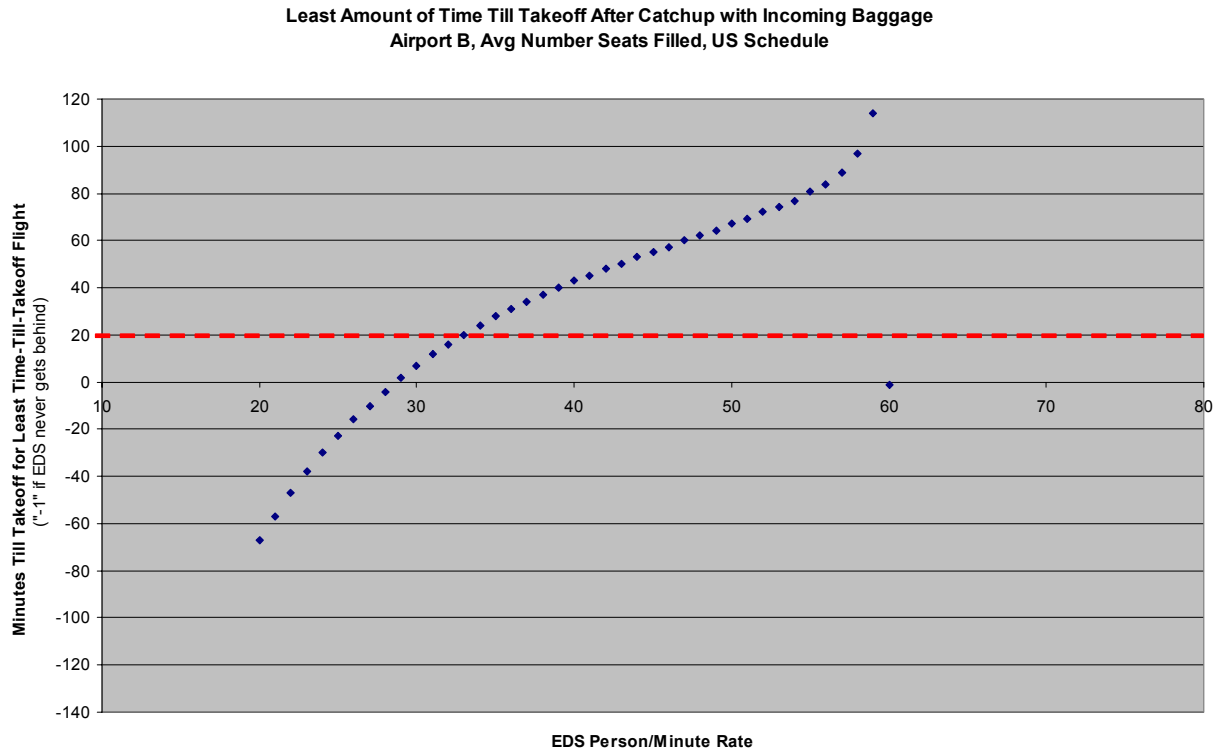
Least Amount of Time Till Takeoff After Catchup with Incoming Baggage
Airport A, Avg Number Seats Filled, US Schedule



Least Amount of Time Till Takeoff After Catchup with Incoming Baggage
Airport A, Avg Number Seats Filled, SS Schedule







So now we have a measure of the delay caused by each rate and we can analyze which pace of people per minute is optimal in cost and delay. The exact rate that was selected for each airport was based on several assumptions that we had to make.

- First, we assumed that on the completely average day, all planes should takeoff on time. If we suggested that the airport get fewer EDSs than the amount needed to handle an average day without a delay, that would cause the final flight of the peak hour to have a chance-of-delay percentage that was greater than 50%. The worst percentages we have seen for a flight is 60%, so this assumption seemed to be reasonable. If most days the flight is late, then it would make sense for the flight to be rescheduled, and since we are given that the flight must come within the peak hour, it must be on-time with a probability of over 50%.
- Second, we realized that our estimate was not taking into account a logistical part of the problem. For example, let our model predict a period of seventeen minutes between when the luggage is ready to get on a plane and the takeoff of that plane. So this appears to mean that this plane has a margin of error of seventeen minutes in which its luggage is packed into the baggage compartment but it has not taken off yet. This is not true however. What we used to generate these data were the rates at which the passengers *are coming into* the airport. We have not taken into account the time that the bag spends at the check-in desk and the travel time from the desk to the EDS queue, and later after the luggage is scanned the time that the bag is being moved from the EDS station to the airplane. We have very little information on this for these specific airports, but certainly this information is known by the managers of an airport, and therefore could be factored in to our equation. We estimated that since people can check baggage approximately thirty minutes before takeoff and still make the flight, an upper bound of thirty minutes is effectual for this travel time. We set the mark at twenty minutes for both airports. So in

other words, for our model, this window of seventeen minutes is now causing a delay of about three minutes, since the seventeen minutes did not take into account the twenty minute travel time of the baggage going through the airport when it is not in the queue at the EDS station. If the actual time varies from this estimation of twenty minutes, the arguments can still be followed with another number in mind.

- The last assumption that we make is that after these first two suppositions are considered, the cost of the system is minimized. In other words, the number of operational machines we are looking for is the number that just barely causes no delays on the average day with a twenty minute travel time for the luggage. Since some flights have as low as 60% on-time percentages, setting the on-time percentage of our flights that are most likely to be late to just above 50% looks to be an accurate way to determine the number of machines.

So the rate at which we would like to process can be seen as the intersection of this final graph and the line $y = 20$, since our travel time is twenty minutes. This is the reason the previous graphs have this shown as a dotted red line. We can determine both the type of schedule that is best for the average day and the minimum rate that will incur zero delays for that schedule.

For airport A, the UU distribution does much better than either the US or SS methods of scheduling by allowing the lowest processing rate that still gets each flight its baggage before takeoff, with the twenty minute period considered. The difference of three people per minute that airport A requires for the other two schedules converts (by the factors below) to at least a difference of one operational EDS. If the rate of processing is less than the recommended 27 to 28 people per minute, it should be noted that the US and SS distribution schedules become even worse when compared to the normal distribution schedule. This could happen, for example, when the computers malfunction at their 8% failure rate.

By this analysis, on the average day at airport A the rate at which passengers are processed should be about 28 people per minute and the best schedule to use is the normal distribution of plane sizes and evenly distributed intervals of time. To convert the passengers per minute units into number of EDSs is a simple computation:

$$\frac{28 \text{ passengers}}{\text{min}} \cdot \frac{1.4 \text{ bags}}{\text{passenger}} \cdot \frac{60 \text{ min}}{\text{hour}} \cdot \frac{\text{hour}}{185 \frac{\text{bags}}{\text{EDS}}} \approx 13 \text{ EDSs}$$

The 1.4 bags per passenger can be determined by a simple discrete probability average. Since 20% of the passengers have no bags, 20% have one bag, and the remaining 60% have two bags, the average number of bags per flyer is:

$$E\left[\frac{\text{bags}}{\text{person}}\right] = 0 \cdot 20\% + 1 \cdot 20\% + 2 \cdot 60\% = 1.4 \frac{\text{bags}}{\text{person}}$$

So the optimal number of operational EDSs for airport A is thirteen, which when we work backwards through the same conversions results in an exact rate of

$$13 \text{ EDS} \cdot \frac{185 \frac{\text{bags}}{\text{hour}}}{\text{EDS}} \cdot \frac{\text{passenger}}{1.4 \text{ bags}} \cdot \frac{\text{hour}}{60 \text{ min}} = 28.63 \frac{\text{passengers}}{\text{min}}$$

However, for airport B, we find a significant difference between the UU distribution and the US and SS schedules in the *opposite* way. The US and SS schedules are almost identical, with the SS schedule performing a marginal amount better. However, for the previously mentioned reasons of why we chose these three schedules, we suggest that these slight benefits of the model with the SS schedule is not worth the penalties incurred from selecting it. These small advantages would not result in a difference of a considerable fraction of an EDS anyway.

Therefore, these data imply that at airport B the processing rate of passengers should be right at 33 travelers per minute coupled with the schedule which involves skewed sizes of planes but distributing the flights evenly throughout the hour. The number of EDSs that this rate indicates is:

$$\frac{33 \text{ passengers}}{\text{min}} \cdot \frac{1.4 \text{ bags}}{\text{passenger}} \cdot \frac{60 \text{ min}}{\text{hour}} \cdot \frac{\text{hour}}{185 \frac{\text{bags}}{\text{EDS}}} \approx 15 \text{ EDSs}$$

Fifteen operational machines actually signify an exact rate of processing to be:

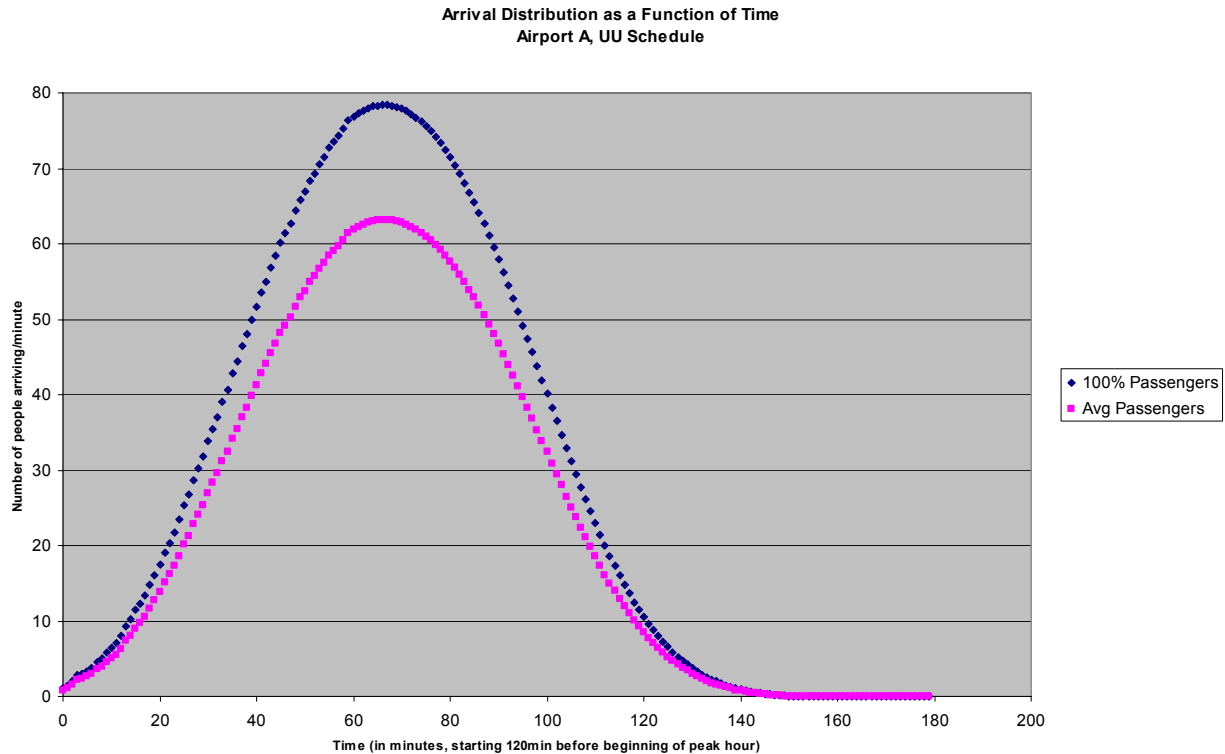
$$15 \text{ EDS} \cdot \frac{185 \frac{\text{bags}}{\text{hour}}}{\text{EDS}} \cdot \frac{\text{passenger}}{1.4 \text{ bags}} \cdot \frac{\text{hour}}{60 \text{ min}} = 33.04 \frac{\text{passengers}}{\text{min}}$$

Throughout this section we have carefully said operational machine for determining these figures. This is because each EDS has a failure rate of 8%. On the average day, if there are either 14 or 16 machines, one will be out of service. **Therefore, our model suggests that airport A purchase 14 EDSs and that airport B purchase 16 EDSs.**

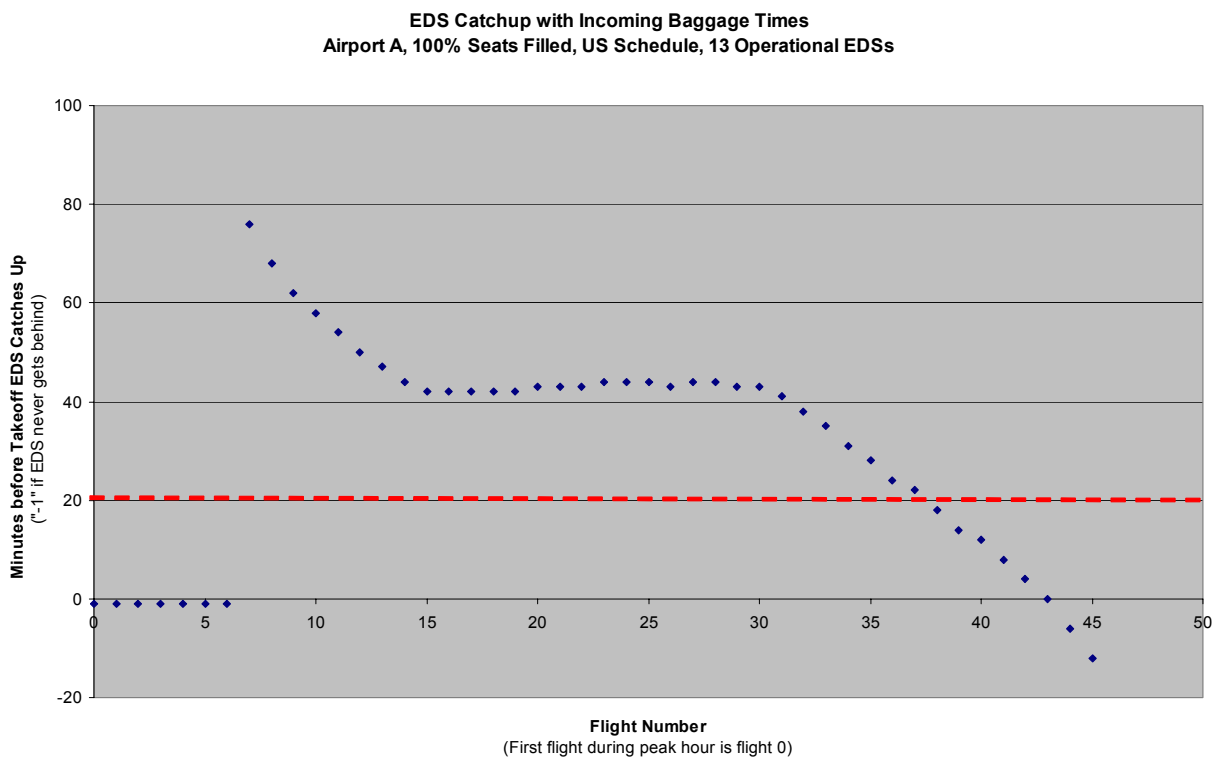
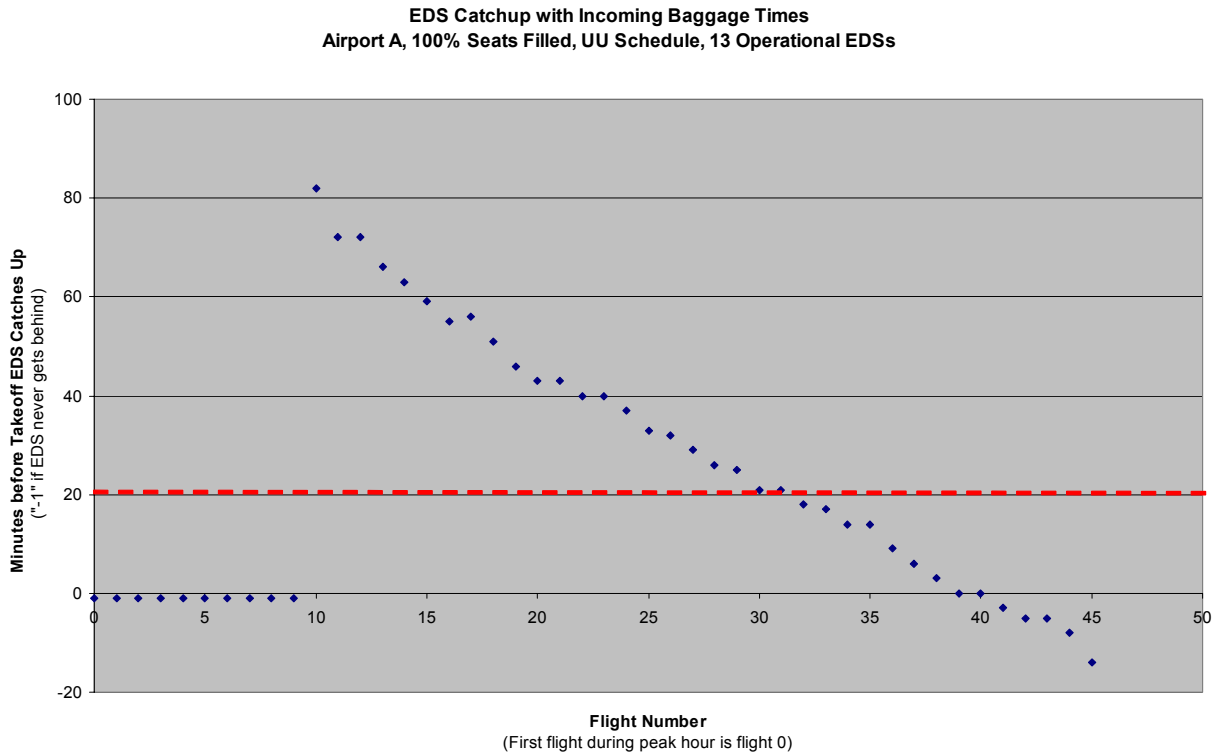
Now that we have the operational rates at which our EDSs will be processing, we can analyze the other two cases that cause delay: high load factors (percentage of seats of a plane that are filled) and failure of EDSs.

To test these schedules with regard to high load factors seems to certainly be worthwhile. There are notoriously tumultuous days for air travel in America, that can be associated with a long time period (June is the busiest month for airlines), holidays (the weekend after Thanksgiving is the most hectic couple of days of the year), or, more rarely, random occurrences. Since most of the demanding days can be predicted by analyzing historical statistics, the flight scheduling of those days could be altered in order to minimize delays.

To determine if changing the flight arrangement would reduce the delay during the peak hour of peak days, we altered our computer programs to accept the number of seats filled at 100% rather than the average. In other words, how long is the delay if every seat is filled on every flight? To give an idea of how much worse the rate of passengers is, this graph overlays the three types of scheduling, each with both the average day and the day in which every plane is full (the all seats filled day) at airport A.



As the graph portrays, the number of passengers is greatly increased. To determine the type of schedule that is best suited for these conditions, we can now take advantage of the fact that we know the operational rate of baggage processing to get more specific information about the amount of delay. With our selection of schedule and speed of scanning for the expected day, we only wanted to make sure that no flights were delayed, so observing the postponement of the most delayed flight was the only necessary factor. On the all seats filled day we know that we are going to have setbacks, but there is more information on this graph to analyze to determine the schedule that optimizes the flow of the flights during the peak hour. The following graphs show the delay for each of the planes at airport A given the two distributions of flights that are dispersed evenly throughout the hour.



There are several new pieces of important information that are depicted in this diagram. For example, by counting the number of data points lying under the travel time line of $y = 20$ minutes, we can find that the number of flights that are delayed. Also, the area under the travel

time line and above each delay curve gives us the total number of minutes all of the flights miss for that curve's respective schedule. Then we are able to determine the average delay of each flight and the average delay of each *delayed* flight, which are easily compared in the table below.

Airport A Delay Statistics

100% Seats Filled

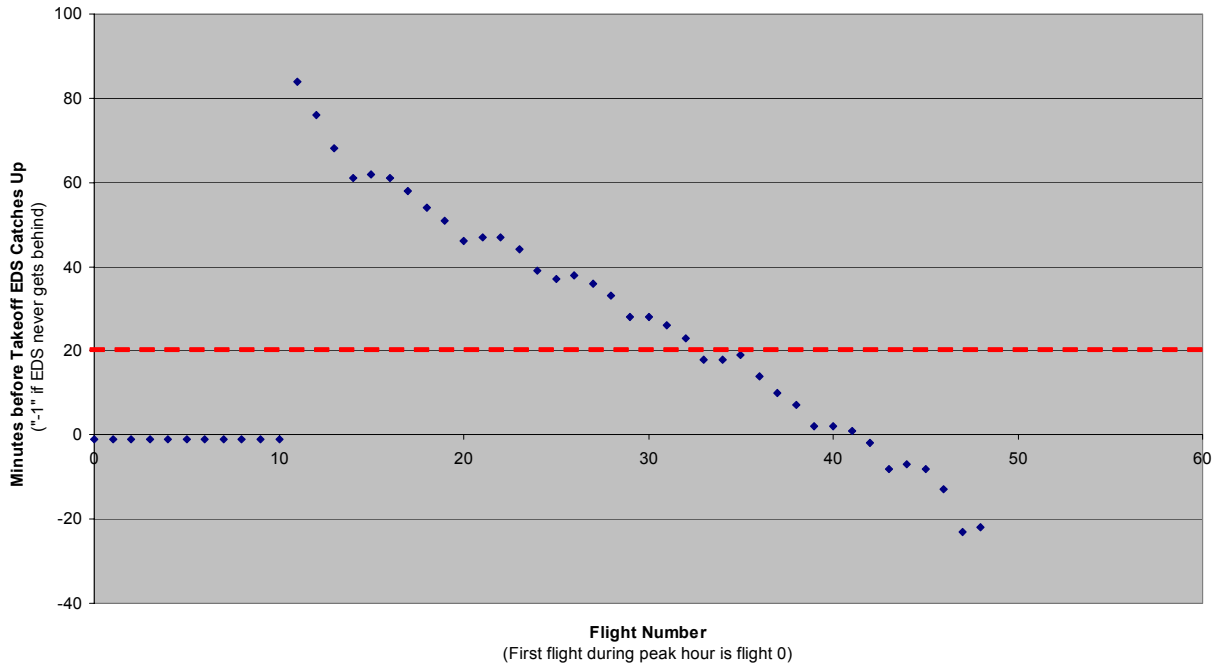
	UU Schedule	US Schedule
# delayed flights	14	8
Avg delay	5.09 min	2.65 min
Avg delayed flight delay	16.71 min	15.25 min
Total minutes of delay	234 min	122 min

This analysis clearly shows that for busy days, the US distribution is superior to the UU distribution. Also, we see that for tumultuous times the average number of delays seems acceptable for being the absolute worst day of overbooking. For most large airlines, on November 23, a notoriously busy day of the year, in the year 1999, over 14% of flights at large airports were delayed more than thirty minutes.¹⁸ The scaled percentage describing the peak hour of these flights would certainly be above the value determined above. A large airport delaying less than 20% of its airplanes during the worst hour of its worst day appears to be doing better than most airports under those circumstances, so this result is satisfactory.

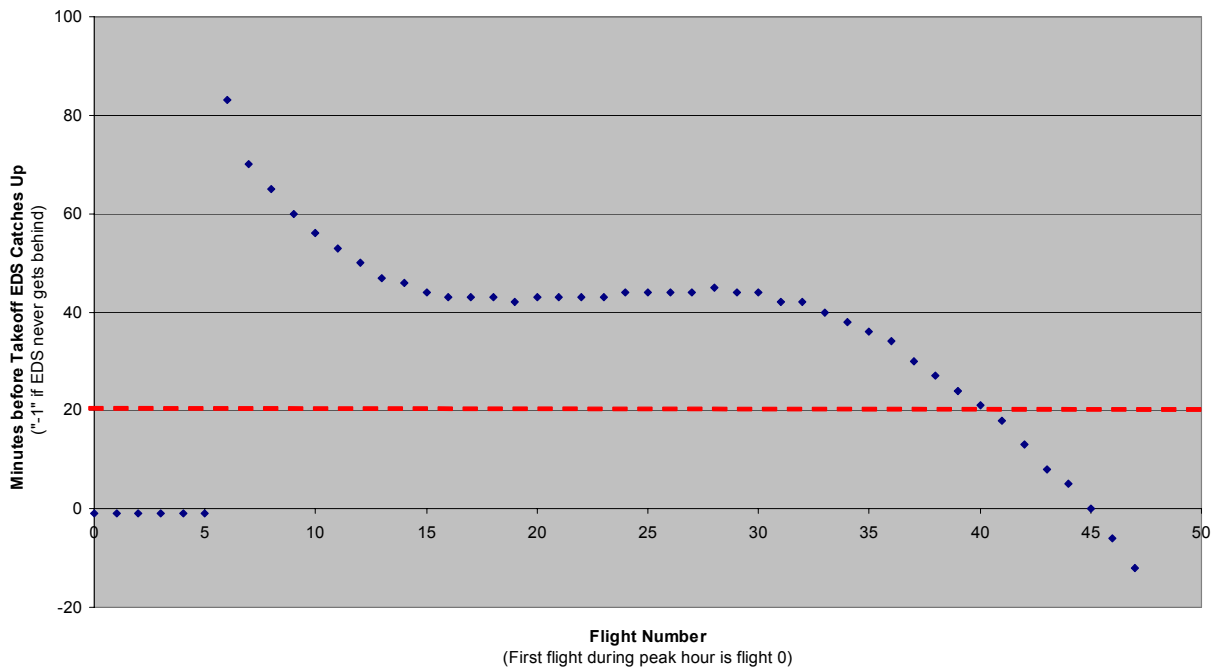
On these infamous days of demanding load factors, the percentage of seats on a plane that are filled, the US schedule should be prepared. Since many of the days that have the most passengers flying are easy to predict, a schedule can be designed weeks beforehand to match the US arrangement of our model. By adopting a different timetable for busy days, we effectively halve the average delay of planes on a busy day.

Airport B has the following graph and table:

EDS Catchup with Incoming Baggage Times
Airport B, 100% Seats Filled, UU Schedule, 15 Operational EDSs



EDS Catchup with Incoming Baggage Times
Airport B, 100% Seats Filled, US Schedule, 15 Operational EDSs



Airport B Delay Statistics 100% Seats Filled

	UU Schedule	US Schedule
# delayed flights	15	7
Avg delay	5.625 min	2.375 min
Avg delayed flight delay	18 min	16.29 min
Total minutes of delay	270 min	114 min

For similar reasons, airport B should also use the uniform distribution to arrange its flights on hectic days. However, since our model suggests the uniform schedule for average days with airport B, this airport does not need to change its flight plan for these busy periods.

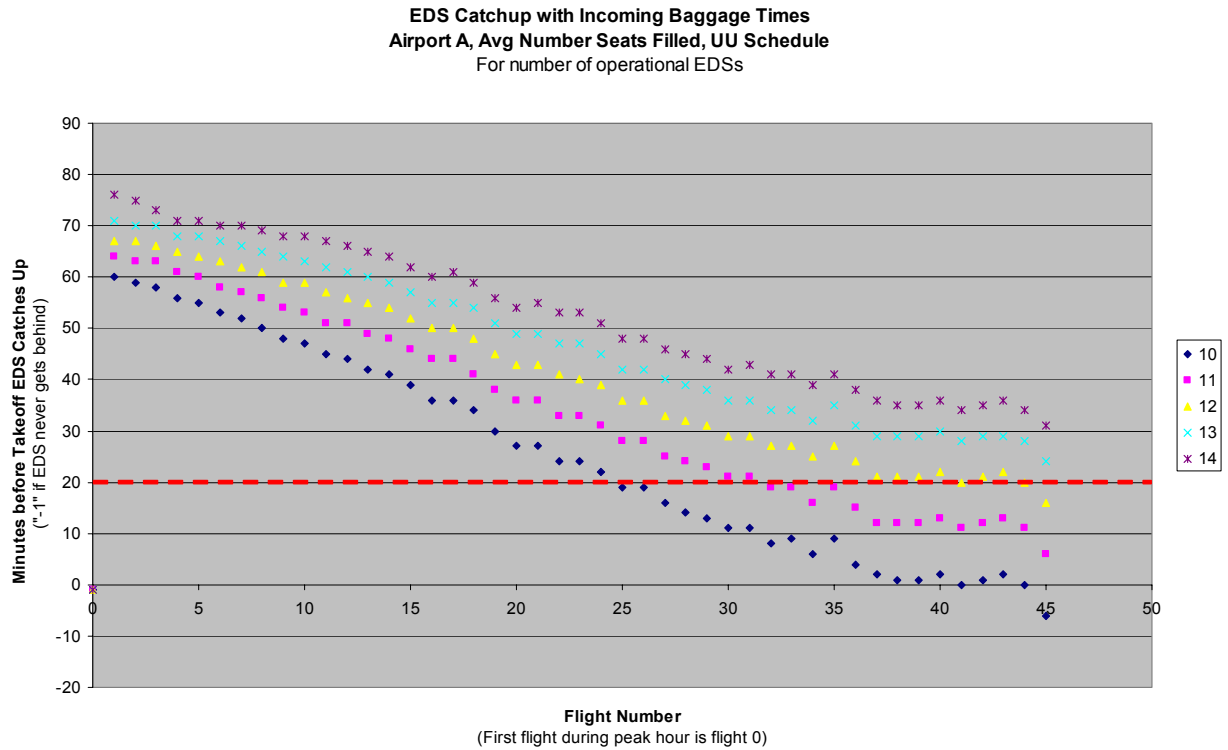
The final cause for a delay is the failure of one or more scanning machines. The operational percentage for the EDSs, 92%, can be interpreted a number of different ways. The major ambiguity of this figure is the length of time that the EDSs are inoperative. It could be that 92% represents time that the machine gets stuck or has to cool down, in which case the factor could simply be multiplied by the rate of processing to get the true scanning speed. In this case the time that it is down is very small, and the length of time that it spends without breaking is also relatively small. The percentage only tells us the ratio that any machine is operational. If instead both the time out of service and the time that a working machine goes without failure are long periods, the rate could not be modified, and a more accurate model would involve probabilities.

Our interpretation of this number was the latter, the probabilistic representation with the broken down periods long compared to an hour. If this was not the case it seems that the figure would have been already factored in with the speed of 185 bags per minute. The reason that it is important that the length of time that the EDS is out of service is much greater than an hour is that our model assumes that the number of machines that are operational does not change throughout the time we are studying each day. Instead, we take one reading for each peak period and rely on that reading for the entire period.

This results in a discrete probability distribution with a calculable expected delay. What is important to note is that for calculating this delay, we are looking at an average day. If we begin to combine all seats filled days with these probabilities, the chance of both occurring is the product of the chances of each and so these results would end be a very rare occurrence and therefore not worth computing. If there are N total machines, then the probability that exactly i of them are broken is a binomial distribution of the form:

$$P(i) = p^{(n-i)} \cdot q^i \cdot \frac{n!}{i!(n-i)!}$$

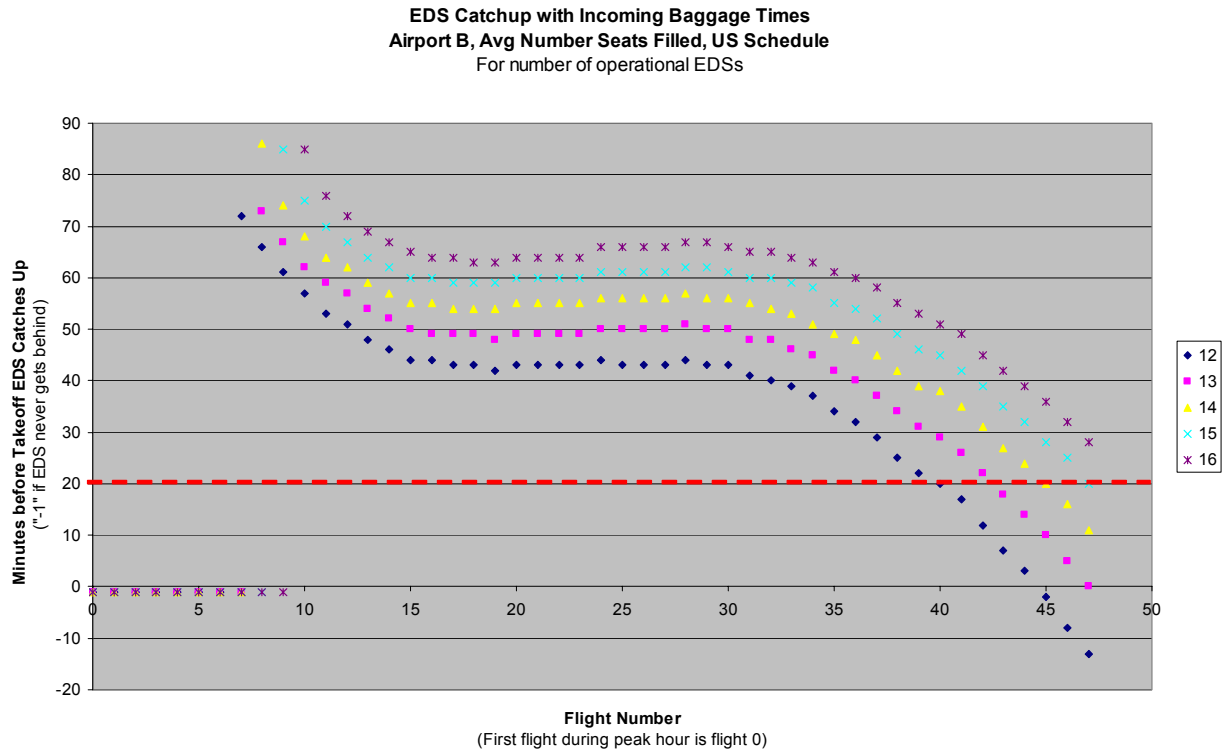
So for all of the following computations we assume that no more than 4 machines are broken at once, with a 99.65% confidence level for airport A with fourteen EDSs and a 99.32% confidence level for airport B. These are equivalent to saying no more than once a year in airport A are more than four machines going to be out of service, and no more than twice a year for airport B.



Airport A UU Schedule Delay Statistics

Avg Number Seats Filled

# operational EDS machines	10	11	12	13	14	Expected Value
# delayed flights	20	14	1	0	0	1.61
avg delay	5.8	1.88	0.08	0	0	0.26
avg delay of delayed flights	13.9	6.43	4	0	0	1.58
longest delay	26	14	4	0	0	2.36
total mins delayed	278	90	4	0	0	12.52



Airport B US Schedule Delay Statistics

Avg Number Seats Filled

# operational EDS machines	12	13	14	15	16	Expected Value
# delayed flights	6	5	2	0	0	1.13
avg delay	2.58	1.1	0.27	0	0	0.24
avg delay of delayed flights	20.67	10.6	6.5	0	0	3.15
longest delay	33	20	9	0	0	5.00
total mins delayed	124	53	13	0	0	11.65

For these ranges, the table above shows that the expected total number of minutes delayed (the sum of all the delay of all planes) due to machine failure is certainly acceptable for both airports.

Prioritization of Planes

This idea is based on an attempt to redistribute delay in order to minimize customer dissatisfaction. This is most important when regarding flights of different sizes. The delay of successive flights tends to accumulate, which is the reason that all of our graphs detailing the delay for each plane as a function of flight order are decreasing. This means that between two delayed planes, the plane that takes off after the other will be more delayed. The basis of prioritization of scanning bags could be changed to also include number of passengers on aircraft in addition to takeoff time. The only way to prevent the second, larger plane from being held up by the first smaller one is to actually switch the order of the takeoffs. This results in the small plane having to wait for the all of the luggage of the larger plane to be scanned before its luggage

can finish being scanned. This method has the chance to dissatisfy fewer customers (those flying in the earlier, smaller plane), but we believe that a small amount of lateness of many customers is preferable to no delay to some customers but a much greater delay to others. The primary reason behind this is the fact that many flyers will need to make connecting flights, and if this delay is maximally distributed, fewer connections will be missed because of it.

An assumption that we made was that flight schedules should be largely static, not allowing the rescheduling of flights a small time prior to the scheduled takeoff time.

Is the Price Right?

Analyzing Political and Financial Limits on the Implementation of ETD devices

Although the security of air travel is paramount to its viability as a means of transportation, other factors also impact whether air travel remains practical for widespread use. The capacity to travel through the skies is useless if no one wants to. If airport delays become sufficiently lengthy, or if the price of air travel rises too high, commuters will seek other means of transportation. All of the important aspects of air travel must be balanced and optimized for society to reap the full benefits.

The current financial situation of airlines is precarious at best. Bankruptcy looms over the head of all the major airlines. The airlines were dealt a double blow by the terrorist attacks of September 11: The drop in air travel caused a large shortfall in revenue, and the fees that will be added to provide airport security will further discourage people from flying.

This section of our model assesses the economic losses that would be caused by adopting a variable number of EDS and ETD machines. Because the federal regulations must be met, this section of the model does not seek to determine the financially ideal number of EDS machines, nor does it attempt to balance financial requirements against the need for safety. Instead, this section of the model determines the impact that acquiring and operating a fixed number of EDS and ETD machines will have on the financial status of airlines. Using the data from the ETD graph, airports (or the Transportation Security Administration) can determine for themselves the ideal deployment of ETD machines.

Section 118 of the Aviation and Transportation Security Act of 2001 specifies that the cost of implementing federal security regulations will be paid for by the revenue from a security service fee imposed on each passenger. The maximum fee that can be imposed is \$2.50 per passenger per flight segment, not exceeding \$5.00 per flight. If this proves insufficient to cover the cost of the security regulations, the Under Secretary of the Transportation Security Administration may impose a fee directly onto the air carriers.¹

A large portion of the cost of air travel is already consumed by fees imposed on passengers. For a \$200 domestic ticket, government fees account for about \$55.⁸ These fees are particularly detrimental to the financial situation of airlines because of the elasticity of demand for airline tickets.

The elasticity of demand relates the percentage change in the price of a product to the percent change in the amount of product bought. Specifically, the elasticity equals

$$e = \frac{\frac{\Delta Q}{Q}}{\frac{\Delta P}{P}} = \frac{\Delta Q \cdot P}{Q \cdot \Delta P}$$

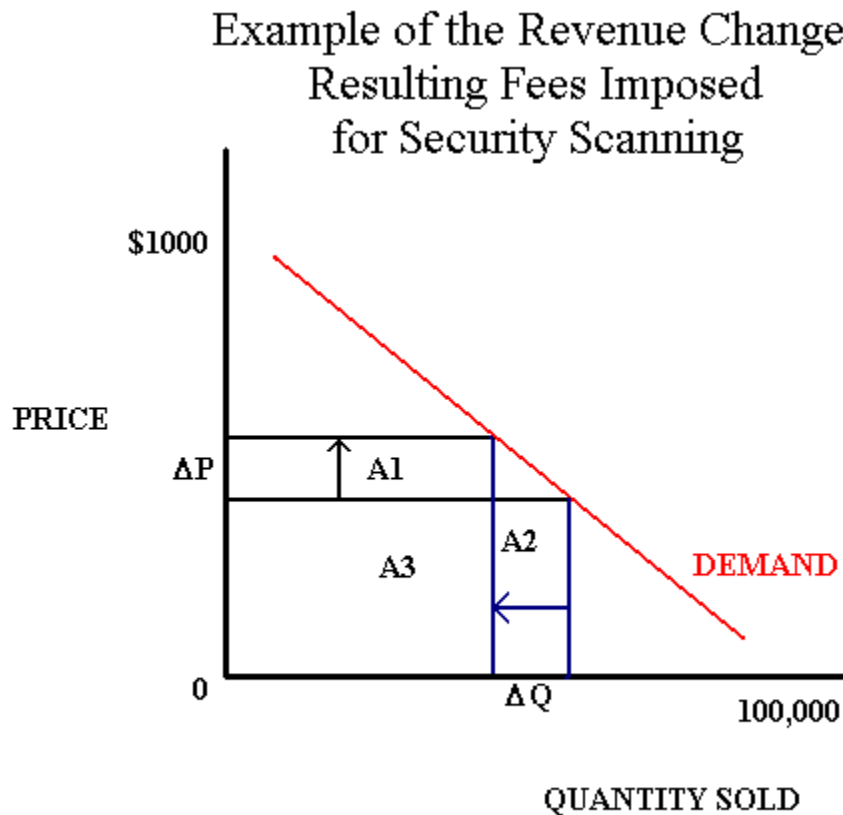
Where P is price, Q is quantity, ΔP is the change in price, and ΔQ is the change in quantity sold that results from the change in price. Elasticity always has a negative value because if the price increases the quantity bought will decrease, and if the price decreases the quantity bought will increase.

If the elasticity of a product is high, buyers of that product are more sensitive to changes in the price of that product. For example, if the elasticity of a product is 2 and the price of the

product increases by 10 %, then the quantity of the product bought will decrease by 20 %. Conversely, if the elasticity of a product is .5 and the price of the product increases by 10 %, then the quantity of the product bought will decrease by 5 %.

The Elasticity of a product depends on the characteristics of the product and the industry making it, and can be determined by researching the changes in quantity of a product sold when different price changes are put into effect.⁹

The change in revenue that a price change causes is obtained by looking at a demand curve for the industry. A sample demand curve illustrates how this is accomplished:



The revenue received before the price increase is given by the area of rectangle A3 added to rectangle A2 (this is true because Price per unit multiplied by the number of units sold equals the revenue). The revenue after the price increase is given by the area of rectangle A3 added to rectangle A1. The relative magnitudes of the ΔP and ΔQ arrows is given by the elasticity of demand for the product. The change in revenue is given by the area of rectangle A1 subtracted by the area of rectangle A2. Because small values for the elasticity of demand minimize the magnitude of the ΔQ in comparison to the ΔP arrow, for a product with an elasticity of demand less than one, an increase in price results in an increase in revenue. Similarly, large values for the elasticity of demand maximize the relative magnitude of the ΔQ arrow. As a result, for a product with a high elasticity of demand an increase in price results in an increase in revenue.

Data for the aircraft industry indicates that the overall price elasticity of airline tickets is 1.5. Because airline tickets have an elasticity greater than one, the air travel industry is particularly sensitive to changes in price caused by the government imposed security fee.⁸ To calculate the

specific effect the screening of passengers will have on the price of an airline ticket and the revenue of airlines, and number of parameters must be known:

Parameters Required for the Analysis of Revenue Change:

- The cost for the average one-way air ticket per flight segment.
- The cost of operating a single EDS machine per passenger.
- The cost of operating a single ETD machine per passenger.

Data Regarding the Required Parameters:

- The average cost of a one-way air ticket per flight segment was \$99.62 in 2002.¹⁰
- The cost for operating an EDS system is about \$219.12 a day.⁶

We assumed that the ETD machine costs about ten times as much to operate a day based on the information given in the problem packet. In addition, because our model analyzes the revenue loss for each person, we assumed that the number of ETD and ETD machines required were dictated only by the number of people checking baggage during the peak hour. We then assumed that of the cost for operating an EDS machine was about 92.6 % due to the cost of actually using the machine and about 7.4 % due to fixed maintenance costs and depreciation. We arrived at this assumption by dividing the cost of operating the EDS machine per day by the cost of screening a single passenger with the EDS machine. As a corollary to this assumption, the cost incurred by the machine per person is about 7.4 % dependent on the number of machines required for the peak hour.¹¹ Because ETD scanning is labor intensive and requires limited capital, the cost incurred by the machine per hour is completely due to the use of the machine.

These assumptions allow us to determine a cost per person for the EDS machines so that we can add that value to the cost of a one-way ticket. We also assumed an average rate of baggage sorting as 185 bags per hour, halfway between the given range of 160 to 210 bags per hour.

Assumptions Regarding the Economic Model:

- The cost for operating and EDS system is about \$2191.20 a day.⁶
- The number of EDS and ETD machines required is dictated by the number of people during the peak hour.
- The fixed maintenance costs for the EDS and account for 7.4 % of the cost of operating the machine.⁶
- The cost of operating an ETD machine is essentially 100 % correlated to the time of the machine's use
- The EDS machine processes 185 bags an hour.
- The ETD machine process 45 bags an hour.
- Each person carries an average of 1.4 bags.

By converting the cost per day to a cost per person, the additional cost each individual has to pay for EDS scanning can be computed:

$$\frac{\text{Cost}}{\text{Person}} = \frac{\$219.12}{\text{Day}} \cdot \frac{\text{Hour}}{185\text{Bags}} \cdot \frac{1.4\text{Bags}}{\text{Person}} \cdot \left(\frac{0.074\text{FixedCost}}{\text{Hour}} + \frac{0.926\text{Day}}{24\text{Hours}} \right) \approx \frac{\$0.19}{\text{Person}}$$

From this value, the percentage change in the price of the average air ticket can be obtained. When this information is combined with the data on the demand elasticity of airports, the cost of security screening on airlines can be computed. **From this, we derive a total revenue loss of about 0.286 %.**

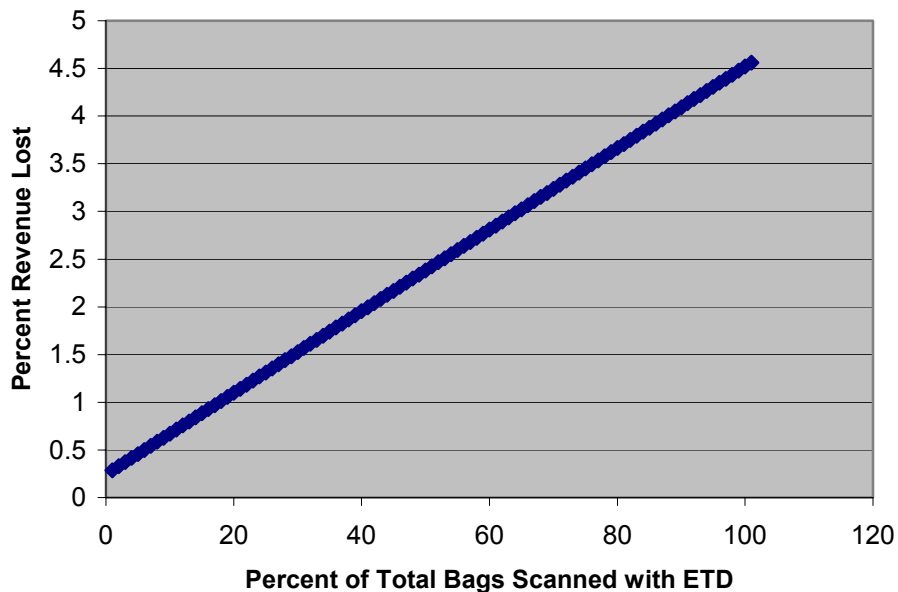
In addition to the cost of running the EDS machines, the cost of running the ETD machines can be analyzed as a function of the percentage of all bags that the ETD machines scan. The additional price paid by the consumer for a certain percentage is given by the equation:

$$\frac{\text{Cost}}{\text{Person}} = \frac{\$2191.20}{\text{Day}} \cdot \frac{\text{Hour}}{45\text{Bags}} \cdot \frac{1.4\text{Bags}}{\text{Person}} \cdot \frac{\text{Day}}{24\text{Hours}} \cdot \frac{n}{t} \approx \frac{\$2.84}{\text{Person}} \cdot \frac{n}{t}$$

Where the notation $\frac{n}{t}$ signifies the percentage of bags that are scanned by the ETD machines.

When combined with the data regarding demand elasticity for the air travel industry, a graph relating the percentage of bags scanned by the ETD to the loss of revenue for the airline can be produced.

The Percent Revenue Lost Due to Security Fees



The equation of the line in the graph is determined by the equation: $y \approx .04276X + .2861$

This yields a revenue loss of about 1.14 % if one-fifth of all bags are scanned with an ETD (as is proposed). This is not an insignificant value. For example, a 1.14 % revenue loss would cause Southwest Airlines to lose an additional 63.84 million dollars a year based on its 2001 revenue. This would reduce the profit of Southwest Airlines by about 12 % a year.¹¹ Such a loss in earnings would make airline stocks even less appealing than they already are and cause a flow of capital from the air travel industry into other sectors at a time when airlines are experiencing a great need for capital.

This model allows airlines and airports to assess the potential financial impact of adopting a policy that specifies a certain percent of bags that must be screened by an ETD. Airports can tailor their security policies regarding the ETD based on the financial health of the airlines they service and the current threat of terrorist attacks. It is our recommendation that under high levels of security threat, the percentage of bags screened by ETD could be increased to 20 %, or even to a higher percentage for a short period of time. As long as the overall average percentage of bags screened for the year is kept around 20 %, the financial loss of revenue for airlines will be kept to an acceptable 1 %. A revenue trade off of 1 % on the part of airlines to prevent the financial catastrophe that would result from another terrorist attack using a plane is also probably in the best interest of the airlines, although this cannot be assessed without analyzing the effect an additional terror attack would have on revenue. We recommend that except in extreme circumstances, the overall revenue loss due to the security service fee be kept at close to 1 %.

Implementation of the ETD at Airports A and B

A Specific Application of the General Economic Model Presented Above

During periods with a high risk of terrorist attacks, it may be required to scan about 20 % of all checked baggage with an ETD device. The cost of this on the airlines has already been established. Now, the number of ETD devices required for the major airport hubs under our jurisdiction, Airport A and Airport B, must be determined. In addition, the effect that additional scanning procedures have on flight delays and flight scheduling must be considered.

First and foremost, we must consider whether we should accept additional delays on high risk days. A significant determinant of this consideration comes from our analysis of the nature of the ETD machine itself. During high risk circumstances when two machine types are used to scan bags, the speed with which one of the machine types processes the baggage will prove the rate limiting factor for the speed that baggage is loaded onto the plane. In essence, because either the ETD or the EDS will require the most time to process the baggage for one flight, the total rate of baggage processing of either the total number of ETD machines or the total number of EDS machines will determine the delay of the flights.

To consider which machine type we should choose to be the limiting factor we consider two aspects of each machine:

- The year round need for the machine.
- The cost incurred by having unused ETD or EDS machines during times when they are not needed.

The acquisition cost of an ETD system is minimal compared to the acquisition cost of an EDS, which would suggest that the EDS system should be the limiting factor if such a choice is possible.

However, because the impact of a relative shortage of EDS machines will impact the delay on far more days than the impact of a shortage of ETD machines (because EDS machines must operate constantly and ETD machines are used only for high risk days) we conclude that there is also a compelling reason to choose the ETD system as the limiting system.

To resolve the issue, we examine the cost incurred by having unused EDS or ETD machines. Because the EDS system is far more capital intensive than the ETD system, the EDS suffers from a far greater fixed maintenance cost. The ETD system, on the other hand, is labor intensive and thus receives the majority of its cost of ownership from the active operation of the machine. As a result, the cost of having the EDS machines as a limiting factor is far greater than the cost of having the ETD machines as the limiting factor. Because there is a fixed number of bags to be scanned by the operating team of the ETD, an increased number of ETD machines should not greatly effect the total operational cost of screening all the bags on a day where ETD is required.

For these reasons, we conclude that the operation speed of the EDS should be the rate limiting factor on days where the ETD is required. As a result, the rate with which the ETD machines scan baggage must be greater than the rate with which the EDS machines scan bags. A second result of specifying the EDS as the rate limiting system is that the schedule and delay times determined as a result of the EDS scanning are also the schedule and delay times that occur

when ETD scanning is required. Thus, we submit no additional modification to the data given for the schedule and delay times when only the EDS machine is used to screen baggage.

After this conclusion has been reached, the number of ETD machines required for Airport A and Airport B can be considered. In order to guarantee that the rate of EDS machine scanning is the limiting factor, the rate of ETD scanning must be kept above the rate of EDS machines at all times.

In order to accomplish this, the number of ETD machines required to keep pace with the number of EDS machines must be determined. This relationship is obtained by relating the processing rate of the EDS machine to the processing rate of the ETD machine and the percentage of all bags scanned by the ETD:

$$ETD > EDS \cdot 0.2$$

Where ETD is the rate of ETD scanning, EDS is the rate of EDS scanning, and .2 is the proportion of all bags scanned by ETD. Using the data acquired about the rate of ETD and EDS scanning, this equation is equivalent to:

$$q \cdot 45 = n \cdot 185 \cdot 0.2$$

Where n is the number of operational EDS machines and q is the number of ETD machines. Simplifying this yields the result that for every functional EDS machine, about .822 ETD machines are required.

From this point, an easy solution would be to round the number of ETD machines required up to an integer value. This would provide a rate of ETD scanning constantly greater than the rate of EDS scanning if the number of EDS machines functional was constant from day to day. However, EDS machines are only operational 92 % of the time. As a result, we take into account the confidence interval for the number of EDS machines that are operational on a given day. In addition, we must express our answer for the number of ETD machines required as a confidence interval as well. Specifically, the result will come in the form that we are confident that a certain percentage of the time, our specified number of ETD machines will process baggage faster than the number of operational EDS machines.

For Airport A, we have a confidence interval of about 50 % that there are at least 13 EDS machines operational on any given day. For Airport A, this means that on average, there is one EDS machine broken each day. If we desire about 50 % confidence that the rate of ETD scanning exceeds the rate of EDS scanning, then the required number of ETD machines is 11. If we desire a greater confidence that our rate of ETD scanning is not the rate limiting function, then we assess the probability that a greater number of machines are operational. In the example of Airport A, where there are only 15 EDS machines total, this is largely irrelevant, as we can easily become 100 % confident that our rate of scanning by assuming one additional EDS machine is operational. If the expected number of broken EDS machines in airport A were greater, we could become progressively more confident that the rate of ETD scanning exceeds the rate of EDS scanning. In the specific case of Airport A, we are 100 % confident that the rate of ETD scanning exceeds the rate of EDS scanning on any given day when the number of ETD scanners is 12.

By similar calculations, we can be 100 % confident that our rate of ETD scanning exceeds the rate of EDS scanning in airport B on any given day when the number of ETD machines is 14. In summary, we conclude that

Conclusions Regarding the Number of ETDs Required in Airports A and B:

- The recommended number of ETD machines to be deployed in Airport A is 15. With this number of ETDs, we are 100 % confident that the rate of EDS scanning will be the rate limiting function for the clearance rate of baggage
- The recommended number of ETD machines to be deployed in Airport B is 16. With this number of ETDs, we are 100 % confident that the rate of EDS scanning will be the rate limiting function for the clearance rate of baggage.
- No change in the flight schedule or estimated time of delay is required due to the implementation of ETD testing with the prescribed number of ETD devices.

Generalizing the Methods Used to Determine Ideal Number of ETD Machines

In order to extend this model to more general airports, the Appendix graph “Relationship Between the Total Number of EDS Devices and the Probability that a Given Number are Broken on Any One Day” details the probabilities that a certain number of EDS machines from a specific total will be broken on any given day. The confidence that the rate of ETD scanning will exceed the rate of EDS scanning is equal to the sum of the probability values for all cells equal to or to the right of the number of machines broken.

This graph was generated through the binomial distribution, which determines probability that $n-i$ machines will be operational given the probability that the machine will be operational on a given day, p , and the probability that the machine will be broken, q . The specific equation is:

$$P(N - I) = p^{(n-i)} \cdot q^i \cdot \frac{n!}{i!(n-i)!}$$

The graph in the appendix should be consulted for producing confidence intervals that the rate of ETD scanning will outpace the rate of EDS scanning on a given day. For airports with few ETD machines, the confidence interval will always be 100 %, but for larger airports, confidence intervals as low as 90 % can be produced and accepted.

Back to the Future:

An Analysis of Promising New Technologies

and their Potential Impact on Security

This section of our report focuses on how to best allocate the resources of research and development between the promising technologies of the future. Specifically, we analyze the advantages and disadvantages of using systems based on the current system of X-ray computer tomography, and future systems based on X-ray diffraction, neutron-based spectroscopy, millimeter wave imaging, quadrupole resonance, and microwave imaging technologies.

It is our belief that research into X-ray diffraction offers the greatest potential benefit to security screening. We conclude that X-ray diffraction could easily surpass X-ray tomography as the next generation of security screening systems. Thus we recommend that the majority of research efforts be allocated to the development of X-ray diffraction.

We also recommend that the potential of millimeter wave imaging be investigated through research due to the uncertainty as to the degree of benefit such a system would bring. We believe research directed into this field should at the very least investigate the potential of this technology. The low cost that a millimeter wave explosive detection system would entail makes this technology a potential means of driving down security fees in the future.

However, we do not recommend the allocation of resources to the development of neutron based spectroscopy, quadrupole resonance, or microwave imaging. The systems do not provide compelling benefits beyond those of current systems, and have disadvantages that make their implementation problematic.

Our specific analyses of each technology are detailed below, as is the reason for our conclusion:

Background Technology and Analysis of Advantages and Disadvantages of the EDS

Currently employed EDS technology relies on measuring the attenuation of X-rays that pass through objects in the baggage. The attenuation of an X-ray depends on the energy, density, and atomic number of the material it passes through. The two physical properties responsible for X-ray attenuation are the photoelectric effect and Compton scattering. Compton scattering alters the energy and path of the X-rays, whereas the photoelectric effect absorbs the X-ray entirely. For elements with a low atomic number, such as elements found in organic compounds, Compton scattering is dominant because more X-ray energy is required to boost the electrons of small elements from their orbitals. For larger inorganic elements, the photoelectric effect is dominant. By computing the influence of the photoelectric effect as compared to the influence of Compton scattering in an object, the EDS can discriminate between organic and inorganic compounds. The EDS uses computer tomography to analyze the density of the objects in the bag. A computer program then compares the densities of the organic objects in the bag to the densities of known explosives.⁵

The advantages of current EDS technology are that detection machines do not involve commonly produced cathode ray tubes and computer tomography systems. Due to the widespread use of constituent technologies, improvements in the technology behind EDS will likely improve with little additional research due to advances in the fields of its constituent technologies. In fact, the profitable state of the air travel security industry implies that sustained

advances from corporate research and development will occur regardless of government funded research.

In addition, an X-ray based system requires fewer safety standards than accelerator-based systems. The high degree of automation also gives the EDS an economic advantage of technologies such as ETD, where the costs of labor intensive operation are prohibitive.

Unlike some technologies, EDS cannot identify the specific type of explosive in a bag. In addition, the relatively high cost and weights of a single device: one million dollars and about 8 tons respectively, can place limits on the financial and spatial capabilities of airports.

Background Technology and Analysis of Advantages and Disadvantages of Neutron-Based Detection Systems

Current prototypes for neutron-based detection involve Pulsed Fast Neutron Transmission Spectroscopy (PFNTS). PFNTS uses a particle accelerator to generate neutrons with a specific total energy. A beam of accelerated neutrons are then passed through the bag. Each element possesses a different energy dependent neutron attenuation curve (i.e. each element scatters neutrons of different energies in different ways). A PFNTS system compares the scattering of a neutron beam passed through baggage to the energy dependent attenuation curves of elements commonly found in explosives: hydrogen, carbon, oxygen, and nitrogen. The PFNTS system then compares the chemical composition of objects in the bag to the chemical composition of known explosives, specifically looking for a high nitrogen and oxygen content in relation to the content of other compounds. The PFNTS system rates each bag on how suspicious its contents are based on its comparison of chemical composition between the objects in the bag and known explosives.⁵

A blind study of PFNTS at the University of Oregon in 1996 assessed the accuracy of the PFNTS in detecting explosives hidden in baggage. Although the PFNTS system was able to positively identify explosives in a bag with a comparable accuracy to the EDS system, the PFNTS system did have problems detecting a specific category of explosives (Class A) which is required for FAA certification. The system produced a false alarm on about 4.5 % percent of all bags in the University of Oregon trial. The high positive detection frequency (for a prototype) and the low false alarm rate shows that PFNTS has potential to be more accurate than current EDS systems.⁵

However, even if accuracy proves to be a strength for PFNTS, the technical, financial, and regulatory weaknesses of using such a device are extensive. The acquisition cost of each device would be several times that of an EDS system, estimated at ten to twenty times the cost of an EDS system. In addition, the size and weight of a PFNTS machine would be several times that of the rather bulky EDS. If several PFNTS machines were required by an airport (as would often be the case), the demands placed on airport space would be significant. In addition, the regulatory concerns of having a particle accelerator in an airport, and the safety need for radiation shielding makes use of PFNTS technology a logistical nightmare. Furthermore, because the constituent technology behind PFNTS is not widely employed, an extensive research program would be required to develop the technology.⁵

Given the financial, developmental, and logistical obstacles, it is our recommendation that neutron-based detection technologies are not promising for development at this time. There is presently greater promise offered by alternate detection technologies.

Background Technology and Analysis of Advantages and Disadvantages of Millimeter Wave Imaging Detection Systems

Primary interest in the security of millimeter wave imaging detection systems focuses on their use in metal detectors to screen passengers entering the airport. Millimeter wave detectors measure light wavelengths in the one millimeter to ten millimeter range, which includes the thermal emissions from human bodies. There is currently great interest in adapting millimeter wave detectors to function like metal detectors. Such a system would be able to identify weapons that a metal detector would miss, such as ceramic or carbon fiber knives, or guns refitted specifically to avoid a metal detector. Millimeter wave detectors are also used to detect land mines.¹²

Current millimeter wave detection employs passive scanning: using the device solely as a detector. This method of scanning measures the thermal emission of the scanned object. The value obtained by the detection system depends on the temperature and emissivity of the object scanned. Organic materials have high emissivities, whereas inorganic materials, particularly the metal wirings of explosives, have emissivities that approach zero.¹²

This detection system functions excellent in situations where there is a sharp contrast between thermal emissions, but could be less accurate when applied to scanning the contents of a bag, which would have relatively uniform thermal emissions. In addition, passive millimeter wave detection systems have difficulty penetrating barriers, such as a wall, or a well placed book concealing an explosive hidden in a bag. For this reason, active millimeter wave scanning would have to be developed for use in baggage scanning. Development of this technology is already underway, because of the potential of easily identifying insect infestations within the walls of a house.¹³

Millimeter wave detectors have low acquisition costs: a system could cost as little as \$100,000. Depending on the development of the scanning system, the procedure could either be automated or labor intensive. The potential for a cheap, automated scanning system is a compelling argument for the development of millimeter wave detectors. In addition, research into millimeter wave detection would provide benefits to other security programs, including passenger screening airport security. Millimeter wave technology also has commercial applications, which would enhance the effectiveness of government research by the research going on in the private sector. One possible advantage that millimeter wave imaging has over X-ray and neutron based detection is that millimeter wave frequencies cause minimal damage to sensitive organic or biomedical cargo, such as food or liquid medication. As a final point, there is a great deal of uncertainty as to the potential of millimeter wave detection for baggage screening. This implies that millimeter wave detection devices should be researched to at the very least determine their potential usefulness.

Considering the wide range of potential security applications for millimeter wave technology, and in light of the potential advantages in cost of screening, it is our recommendation that the Transportation Security Administration or the National Research Council pursue and develop millimeter wave detection systems for use as an airport security system. With regard to scanning checked baggage, we recommend that research focus first on developing active scanning millimeter wave detectors. After these detectors are developed, the next stage of research should involve investigating the potential accuracy of millimeter wave detectors. After the accuracy of

millimeter wave detectors is determined, the National Research Council or Transportation Security Administration can decide whether to continue developing millimeter wave detection technology.

Background Technology and Analysis of Advantages and Disadvantages of X-Ray Diffraction Detection Systems

X-ray diffraction technology has already been deployed in baggage screening in some airports. For example, X-ray diffraction devices (HDXs) were installed in Munich in 2001 to complement the EDS machines already in use there. HDX machines are more precise than EDS machines. In Munich, they are used to scan bags that the EDS identifies as suspicious. The HDX machine will either clear a bag identified as suspicious, or it will confirm the conclusion of the EDS machine and send the bag to be manually investigated. This system is completely automated, and greatly reduces the time required to scan baggage because it reduces the number of false alarms that must be investigated manually.¹³

X-ray diffraction works in a manner similar to the computer tomography of the EDS machine, but it also includes some concepts from X-ray crystallography. X-rays pass through an object in baggage and scatter with an angle dependent on the Compton scattering and photoelectric effects. The angle of diffraction detection at various depths of the bag indicates the different composition of the objects in the bag at that depth. When the data from multiple scans are analyzed by a computer, a three dimensional location for sources of scatter spectra is produced, and the contents of each object is analyzed based on that information.¹⁴

The advantages for X-ray diffraction technology are similar, but also potentially greater than, EDS technology. Both systems are automated, have high accuracies, are based on widely used technologies, and minimize operational cost. Research into X-ray diffraction would be relatively easy because machines employing the technology already exist, and the effectiveness of research suggestions could be readily tested. Similarly, X-ray diffraction and EDS technology have similar disadvantages. Both involve large, heavy machines with high acquisition costs.¹⁴

We conclude that research into X-ray diffraction technology offers the greatest potential benefit off all the technologies detailed here. Refinements in the accuracy of HDX machines, and reductions in the cost of a machine could allow X-ray diffraction technology to completely surpass current EDS technology in the near future. The low rate of false alarms would allow a much higher throughput rate if the HDX machines could be improved to process bags at the same rate as a current EDS device. It is our recommendation that the Transportation Security Administration and the National Research Council aggressively pursue the development of X-ray diffraction technology as the next generation of baggage screening explosive detection systems.

Background Technology and Analysis of Advantages and Disadvantages of Quadrupole Resonance Detection Systems

Quadrupole resonance (QR) detection technology is already used in commercial weapons detection devices. The QR based scanner Qscan made by Quantum Magnetics is used to scan for land mines and explosives in carry-on luggage. The Department of Defense has also authorized \$35 million for research into QR technology.¹⁵

QR works by scanning objects with low frequency radio waves. These waves respond in characteristic ways to the nitrogen nucleus, indicating the nitrogen concentration in a scanned object. The concentration of nitrogen measured in a scanned object is compared to the nitrogen composition of known explosives.¹⁵

One benefit of QR is that it can detect and identify over 10,000 substances. Despite the wide range of substances it can identify, QR has difficulty with some common substances. For example, a QR scanner will identify leather as a nitrogen based explosive. Additionally, QR is incapable of scanning through metal, although it will indicate when it cannot scan a compartment of a bag for this reason. Unfortunately, the issues with leather and metal will lead to a large number of false positive which will require a more labor intensive system to manually investigate all the false positives, and which will lead to increased delays in baggage scanning. QR is also incapable of detecting certain explosives, such as liquid explosives, amorphous plastic explosives, and explosives without QR active nuclei.¹⁵

This technology can be coupled with Nuclear Magnetic Resonance (NMR) to greatly increase the accuracy of scanning. NMR is sensitive to the explosives that QR is not, so that when the two systems are combined they can cover the entire spectrum of explosives. However, NMR cannot resolve the false positive caused by leather, nor can it scan through, metal. Although the entire spectrum of explosives can be covered by a combination of NMR and QR, the vexing issue of false positives remains.¹⁵

In light of the limitations on the QR detection, even when combined with NMR technology, it is our recommendation that the Transportation Security Administration not pursue research into quadrupole resonance at this time. The physical limitations of the technology that must be overcome do not, in our judgment, justify the potential for a QR and NMR explosive detection system, especially when compared to the promise of X-ray diffraction and millimeter wave technologies. If research undertaken by the Department of Defense refines QR technology enough so that it can distinguish leather from explosives, then research into QR technology by the Transportation Security Administration would be justified.

Although we recommend that this technology not be developed so that research effort can be focused on X-ray diffraction and millimeter wave detection systems, we also conclude that of the technologies we failed to recommend, we judge this to be the most promising. If the National Research Council or the Transportation Security Administration deems that researching only two technologies provides an insufficient depth of development, then we recommend that this technology be developed as opposed to neutron based and microwave imaging detection systems.

Background Technology and Analysis of Advantages and Disadvantages of Microwave Imaging Detection Systems

Microwave imaging relies on a tunable series of scans to create visual cross sections of a checked bag. By using different frequencies of microwaves, the imaging system visually distinguishes and displays layers of the bag, as if the operator were visually slicing through the bag and observing the contents of each slice. This system is operated by a human, and is thus subject to human error as well as the high operation cost of paying an operator. This device is better than the X-ray imaging used to screen carry-on baggage because it can see metal objects hidden behind substances of lower atomic nuclei.¹⁶

A principle advantage of the microwave imaging system is that it causes minimal damage to objects in the bag sensitive to higher energy probing techniques (such as neutron or X-ray based detection techniques). Another advantage is the low rate of false positives and the high throughput rate that such a system could attain. However, the disadvantages of microwave imaging are extensive. The crucial weakness is the low positive detection rate of the system. The quality of this scanning system is limited to the skill of the operator interpreting the information. In addition, it may be impossible for even the best of operators to determine that an object they are seeing is an explosive device if it is well disguised. To have an accuracy comparable to current automated detection systems, an operator would have to be able to identify at least 98 out of every 100 bags that contain an explosive device. That is a tall order even for the most devoted and well trained of employees.¹⁶

It is our conclusion that the age of visual interpretation by a human operator in the field of security is nearing an end. The accuracy of a user operated and interpreted system is too low, and the cost of such a system is too great to justify its application no matter how effectively the technology can be refined. Perhaps in the future, when artificial intelligence has advanced enough to easily compete with a human in terms of visual interpretation, research of this technology might be justifiable. However, at the present point in time, it is our recommendation that the Transportation Security Administration not pursue microwave imaging as a security tool for screening checked baggage.

Assessing the Strengths and Weaknesses of the Model:

Strengths of the Model:

- We believe that the greatest strength of our model is the wide range of factors and solutions that we consider. Because we generate three different scheduling strategies and provide programs to generate specific schedules and determine which of the schedules is best. In addition, we consider confidence intervals that a certain number of EDS machines will be operational and provide information for determining the number of total EDS machines required for a given level of confidence. The wide range of solutions that we consider and offer as possibilities allows our model to be adapted to a great variety of considerations individual airports may have. This allows each airport to consider its individual characteristics to the maximum extent.
- An additional advantage of our model comes from the way in which the data behind our model was derived. Because we used computer programming to generate and analyze the data behind our model, it is easy to generate a vast amount of specific data with a minimal change to the initial source data or to the program.. For example, after the data and graphs for airport A was generated, all that was required to generate the data for airport B was to change 8 cells in an Excel spreadsheet to correspond to the number of planes of each type that depart during the peak hour of airport B. The speed and power of our computer programming allows each airport to determine the effects of many different scheduling plans and EDS machines in a minimal number of time. This minimizes the possibility that a specifically generated model will be inaccurate due to an error in data entry.
- We also believe that our model is very robust to deviations from our assumptions. We analyzed the worst case scenario for our scheduling and EDS number models, which replaced all our assumptions about the number of passengers per plane, bags per person, and EDS machine operational with the most extreme assumptions possible (completely packed planes, two bags per person, and the number EDS machines broken on the worst day of the year. We found that, although delays on the worst days are vexing, the delays are acceptable. Thus, if our assumptions about the average number of passengers or bags per passenger is inaccurate, only the delay on average days is effected. Although our basic assumptions may be off to a significant degree, our worst case scenario places an absolute upper bound on the delay times. Because the delays on even the worst of days is acceptable, we conclude that our model will be acceptable even for large deviations from our assumptions.

- Another strength of our model is the ease that it can be adapted to changing government requirements for security screening. For example, if another terrorist attack occurs and Congress specifies that all baggage must be scanned by two separate EDS machines, then our model can be altered to consider the effects of double screening by doubling the number of operational machines and calculating the new number of machines required as dictated by the desired confidence interval. The additional delay could easily be calculated by shifting the new rate of baggage checking to the corresponding level such a number of EDSs would be expected to have.
- We believe that our model of the financial effects of ETD and EDS scanning is also strong. We believe that our approach of examining the financial impact from the point of view of the airlines, who are currently in the greatest financial peril, as opposed to the government or passengers, offers the most accurate analysis of the impact of security scanning. One particular strength of our economic model is that airlines can predict how their revenues will change when a certain security procedure is implemented, allowing them to make immediate changes to their ticket pricing or fiscal outlays, as opposed to having implement more drastic policies when they experience an unexpected revenue shortfall.

Weaknesses of the Model:

- One weakness of our model is that a person must be familiar with the programming language C++ in order to understand what the function of our source code is. C++ is not as well known as simpler methods of data computation, such as Excel. However, we believe that the strengths offered by programming in C++ outweigh this disadvantage. For a matter as important as airline security, it is important to produced the most comprehensive report possible, even if the resources of a computer programmer may be required to understand the specifics of the source code. The systems engineers that would be most interested in the recommendations of our model would probably know C++, and be able to easily adapt the source code if needed.
- Another weakness of our model is that it is specific to the level of technology currently employed in airport security systems. The development of better scanning technology, whether it is slower, but cheaper and more accurate, or of the same accuracy only faster, will change our recommended number of EDS machines, and possibly which scheduling strategy should be used by an airport. As a result, this model will likely remain valid for the next two to five years. There is no way to avoid this weakness. Although the promise of future detection systems can be evaluated and recommendation made as to which technologies should be developed, the specific features of future detection systems are impossible to predict with a reasonable amount of certainty.

- We believe that uncertainty about the future is also a potential weakness regarding our recommendations as to which technologies should be developed by the TSA and NRC for use in baggage screening. Many of the technologies we assessed are also being researched by the private sector for commercial applications. If private sector research makes a breakthrough in one of the fields we have not recommended to be researched, that field of research could become far more promising as a future technology. For example if the Department of Defense of private sector finds a way to refine quadrupole resonance so that it can distinguish leather from an explosive, quadrupole resonance becomes a far more appealing technology. The recommendations for which technologies should be pursued are based on assessments of the capabilities and limitations of the technology at this time. These assessments, and thus the recommendations based on them, may no longer be valid in the future. As a result, we recommend frequent evaluations (perhaps every two to three years) of the changing potentials of future technologies.

Testing the Accuracy and Validity of the Model

The best way to test our model is to actually apply the various flight scheduling strategies to an airport. Unfortunately, this would require actually purchasing and installing the prescribed number of EDS machines for an airport and then assessing the delays caused by them with various flight scheduling. If this is done and the delays predicted by the various flight schedules coincides well with the actual delays when the schedules are implemented, then our model functions well in determining the ideal number of EDS machines required and the best flight scheduling strategy to employ. Because the number of people departing from the airport would vary from day to day, a trial period of about a month (about 10 days with each flight schedule) would be required to accurately assess how well our models predictions fit the actual behavior of the airport.

A less expensive way to test the model would be to develop a simulation for the behavior of a specific airport. By applying the results of our model to the simulation and comparing the results of the simulation to the predictions of our model, the accuracy of our recommendations could be estimated using the simulation. The advantage of this approach is that it would not require actually purchasing EDSs to test the model. The implications that govern the accuracy of the accuracy of the model in the case of actual testing hold true for the case of the simulation.

A combination of the two methods could also be used. The simulation could be used to provide a rough estimate for the accuracy of our model. If the simulation's assessment of our model shows our model to be roughly accurate, then the model could be tested in an actual airport.

Method Improvements

Steps That Could be Taken to Improve the Model

Validating the Scheduling Methods with More Seat Filling Scenarios

We have looked at two scenarios in our analysis, all seats on all airplanes being filled and all planes having the average number of seats filled. Additionally, a look at other scenarios might be helpful in choosing the number of machines desired; for example, we could look at random number of seats filled for each plane. However, the change in delays is a fairly predictable function, and so the pros to using the random method (finding odd scenarios one has not thought of) are few. Instead, closely studying a few targeted cases would be useful in more acutely choosing the number of scanners. One example is to selectively distribute the load factors of planes and using this to find confidence levels for number of bags to be scanned. While this would be helpful, planning for the typical day and looking to avoid overly-extreme delays on days with much larger load factors, as we have done, requires much less knowledge of the distribution of load factors and is in itself sufficient to recommend the number of scanners.

Creating an Algorithm to Generate Our Normal Schedule

In our analysis of flight schedules we have considered the three methods mentioned, US, SS, and UU. We developed computer programs which deterministically generate US and SS schedules given the number of flights and number of people expected on each flight and approximated the UU schedule by hand. This being an approximation should not alter the actual utilization of scanners significantly because of the characteristics of the normal distribution, but creating schedules for other times for this airport and other airports would be more easily done if a computer algorithm were used instead.

References:

1. United States. Congress (107th, 1st session: 2001). Aviation and Transportation Security Act : conference report (to accompany S. 1447).
<<http://purl.access.gpo.gov/GPO/LPS16675>>.
2. US Department of State, 1986. Patterns of Global Terrorism.
< <http://www.state.gov/>>.
3. Rodney, Wallis. Lockerbie: the Story and the Lessons. Westport, Conn. Praeger. 2001.
4. Gerárd Vichniac. X-Ray Vision Defuses Terrorist Threat. Mercury Computer Systems.
www.mc.com/literature/literature_files/RTC4-99X-rayVichniac.pdf
5. National Academy Press. The Practicality of Pulsed Fast Neutron Transmission Spectroscopy for Aviation Security.
<<http://books.nap.edu/books/030906449X/html>>.
6. Upali Vandebona. Composition of Baggage Screening at Airports. University of South Wales.
<<http://www.civeng.unsw.edu.au/conferences/CAITR2002/Caitr%202002%20Upali%20paper.pdf>>.
7. International Air Transport Association (IATA). Airport Terminal Reference Manual 7th Ed. 1989. Montreal, Canada. <<http://www-civil.eng.monash.edu.au/people/centres/its/WorkshopsSeminars/PastActivities/caitr/ahyudanari.pdf>>.
8. Buchholz, Todd G. From Here to Economy: A Shortcut to Economic Literacy. Plume. 1996

9. Timothy L. Jacobs, Richard M. Ratliff and Barry C. Smith. Soaring with Synchronized Systems: Coordinated scheduling, yield management and pricing decisions can make airline revenue take off. Institute for Operations Research and the Management Sciences. Lionheart Publishing, Inc. 2000.
10. Jet Blue Airlines Corporation. Assessment of Business Strategy.
<<http://www.kbcc.cuny.edu/UCVE/JETBLUE%20CASE.PDF>>.
11. Southwest Airlines. Southwest Airlines Fact Sheet.
<http://www.iflyswa.com/about_swa/press/factsheet.html>.
12. Murray, Charles J. Wanted: Next-Gen Tech for Weapons Detection. EETimes. September 2001. <<http://www.eetimes.com/story/OEG20010914S0035>>.
13. RheinmetallGroup. Major Order for X-Ray Inspection Systems.
<http://www.rheinmetall.com/englisch/pdf/profil_3_2001.pdf>.
14. US House of Representatives (107th Congress). Rizkalla.
<<http://www.house.gov/transportation/aviation/hearing/03-16-00/rizkalla.html>>.
15. Denise Forscht. Nuclear Quadrupole Resonance. Naval Explosive Ordnance Disposal Technology Division.
<http://216.239.51.100/search?q=cache:ckN2vCylWl0C:www.onr.navy.mil/sci_tech/ocean/reports/docs/oe/00/oe2fors2.pdf+%22quadrupole+resonance%22+airport+security&hl=en&ie=UTF-8>.
16. APS News Online. Technologies to Counter Terrorism.
<<http://www.eps.org/aps/apsnews/whitepapers/wp010306.html>>.
17. Ross, Sheldon. A First Course in Probability 6th Ed. Prentice Hall. 2002.

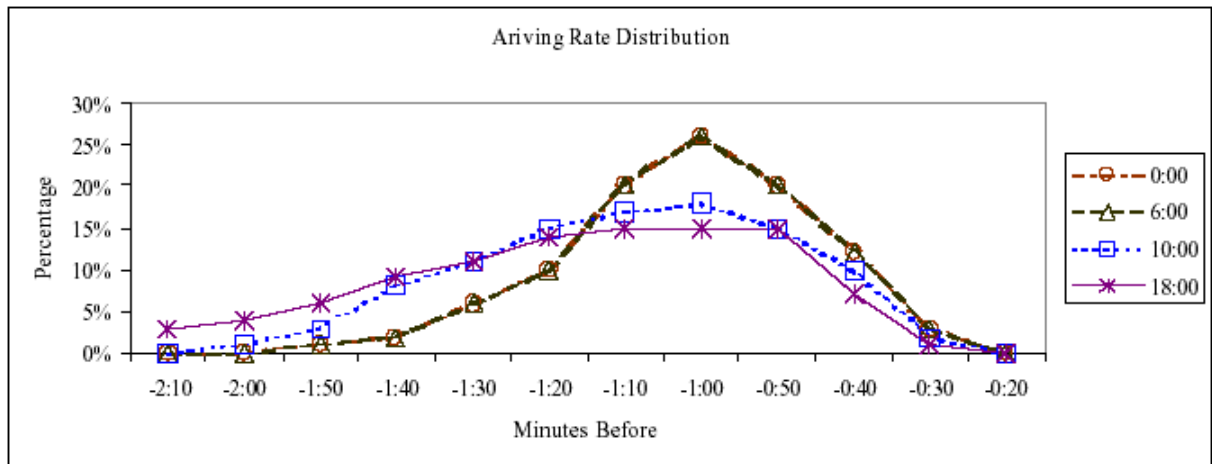
18. U.S. Department of Transportation. Thanksgiving Travel Delays.

<<http://www.bts.gov/oai/holidaytravel/carriers.cfm>>.

19. Chang, Yih-Long. WinQSB v. April 2000. <<http://my.fit.edu/~dhott/WinQSB.htm>>.

Appendix

Graph 1. Data Regarding the Observed Arrival Time Distribution of Passengers (associated with citation number 7).



Probability that a number of EDS machines are broken on a given day

total

41	0.03276	0.11678	0.20310	0.22960	0.18967	0.12205	0.06368
40	0.03561	0.12384	0.21000	0.23130	0.18605	0.11648	0.05908
39	0.03870	0.13125	0.21684	0.23256	0.18200	0.11078	0.05459
38	0.04207	0.13900	0.22361	0.23334	0.17754	0.10498	0.05021
37	0.04572	0.14711	0.23027	0.23360	0.17266	0.09909	0.04596
36	0.04970	0.15558	0.23676	0.23333	0.16739	0.09315	0.04185
35	0.05402	0.16442	0.24305	0.23248	0.16173	0.08719	0.03791
34	0.05872	0.17361	0.24909	0.23104	0.15570	0.08123	0.03414
33	0.06383	0.18315	0.25482	0.22897	0.14933	0.07531	0.03056
32	0.06938	0.19305	0.26019	0.22626	0.14264	0.06946	0.02718
31	0.07541	0.20328	0.26514	0.22287	0.13566	0.06370	0.02400
30	0.08197	0.21382	0.26961	0.21881	0.12843	0.05807	0.02104
29	0.08909	0.22467	0.27351	0.21405	0.12099	0.05260	0.01830
28	0.09684	0.23579	0.27679	0.20860	0.11337	0.04732	0.01577
27	0.10526	0.24714	0.27937	0.20244	0.10562	0.04225	0.01347
26	0.11442	0.25868	0.28117	0.19560	0.09780	0.03742	0.01139
25	0.12436	0.27036	0.28211	0.18807	0.08995	0.03285	0.00952
24	0.13518	0.28211	0.28211	0.17990	0.08213	0.02857	0.00787
23	0.14693	0.29387	0.28109	0.17110	0.07439	0.02458	0.00641
22	0.15971	0.30553	0.27896	0.16172	0.06680	0.02091	0.00515
21	0.17360	0.31700	0.27566	0.15181	0.05940	0.01756	0.00407
20	0.18869	0.32816	0.27109	0.14144	0.05227	0.01454	0.00316
19	0.20510	0.33886	0.26520	0.13068	0.04545	0.01186	0.00241
18	0.22294	0.34894	0.25792	0.11961	0.03900	0.00950	0.00179
17	0.24232	0.35822	0.24919	0.10834	0.03297	0.00746	0.00130
16	0.26339	0.36646	0.23900	0.09698	0.02741	0.00572	0.00091
15	0.28630	0.37343	0.22731	0.08565	0.02234	0.00427	0.00062
14	0.31119	0.37884	0.21413	0.07448	0.01781	0.00310	0.00040
13	0.33825	0.38237	0.19950	0.06361	0.01383	0.00216	0.00025
12	0.36767	0.38365	0.18349	0.05318	0.01041	0.00145	0.00015
11	0.39964	0.38226	0.16620	0.04336	0.00754	0.00092	0.00008
10	0.43439	0.37773	0.14781	0.03427	0.00522	0.00054	0.00004
9	0.47216	0.36952	0.12853	0.02608	0.00340	0.00030	0.00002
8	0.51322	0.35702	0.10866	0.01890	0.00205	0.00014	0.00001
7	0.55785	0.33956	0.08858	0.01284	0.00112	0.00006	0.00000
6	0.60636	0.31636	0.06877	0.00797	0.00052	0.00002	0.00000
5	0.65908	0.28656	0.04984	0.00433	0.00019	0.00000	
4	0.71639	0.24918	0.03250	0.00188	0.00004		
3	0.77869	0.20314	0.01766	0.00051			
2	0.84640	0.14720	0.00640				
1	0.92000	0.08000					

#

broken

0

1

2

3

4

5

6

To determine confidence interval that n machines are operational on any given day, sum the probabilities that more than the given number are operational on a day for each

row. For example, how to find the confidence interval that 30 machines are operational is demonstrated below

								Confidence
39	0.03870	0.13125	0.21684	0.23256	0.18200	0.11078	0.05459	96.67249
38	0.04207	0.13900	0.22361	0.23334	0.17754	0.10498	0.05021	97.07415
37	0.04572	0.14711	0.23027	0.23360	0.17266	0.09909	0.04596	97.44180
36	0.04970	0.15558	0.23676	0.23333	0.16739	0.09315		93.59141
35	0.05402	0.16442	0.24305	0.23248	0.16173			85.56975
34	0.05872	0.17361	0.24909	0.23104	0.15570			86.81535
33	0.06383	0.18315	0.25482	0.22897				73.07713
32	0.06938	0.19305	0.26019					52.26165
31	0.07541	0.20328						27.86851
30	0.08197							8.19662
	0	1	2	3	4	5	6	

Thus if a 90 % confidence that at least 30 machines are operational is required, the total number of machines required is 36

Source Code

build_schedule.cpp

```
// Builds schedule using largest-flights at edge of hour algorithm.
// Takes in Identity.txt file from stdin, using top for number and
// size of flights.
// Use flag -uniform to create uniform distribution of number of planes
// through time or -bunch to bunch planes consecutively at the beginning
// and end of time.

#include <iostream>
#include <set>
#include <string>
using namespace std;

void print_usage()
{ cout << "Usage: build_schedule < -uniform | -bunch >" << endl; }

multiset<double> load_flights()
{
    multiset<double> flights;
    string trash;
    cin >> trash; // "IDENTITIES"
    cin >> trash >> trash >> trash >> trash >> trash >> trash; // 2nd line
    double dtrash;
    cin >> dtrash;
    while (0 != dtrash)
    {
        double avg_people;
        int num_flights;
        cin >> avg_people >> num_flights >> dtrash;
        cin >> dtrash; // seats
        for(int i=0; i<num_flights; i++)
            flights.insert(avg_people);
    }
    return flights;
}

int main(int argc, char **argv)
{
    const unsigned int schedule_length = 60;
    vector<double> schedule(schedule_length, 0);

    // Parse command line options
    bool uniform_plane_dist = false;
    if(argc != 2)
    { print_usage(); return 1; }
    if(!strcmp("-uniform", argv[1]))
        uniform_plane_dist = true;
    else if(!strcmp("-bunch", argv[1]))
        uniform_plane_dist = false;
    else
    { print_usage(); return 1; }

    multiset<double> flights = load_flights();

    double mins_per_plane;
    if(uniform_plane_dist)
        mins_per_plane = (double)schedule_length / (double)flights.size();
    else
        mins_per_plane = 1;

    // Layout plane takeoffs
    double schedule_posn = 0;
    for(multiset<double>::reverse_iterator it=flights.rbegin();
        flights.rend() != it;
        it++)
    {
        schedule[(int)schedule_posn] = *it;
        if(schedule_posn < schedule.size()/2)

```

```
        schedule_posn += mins_per_plane;
        schedule_posn = schedule.size() - schedule_posn;
    }

    for(unsigned int i=0; i<schedule.size(); i++)
        cout << i << " " << schedule[i] << endl;

    return 0;
}
```

load_flight_schedule.h

```
#ifndef LOAD_FLIGHT_SCHEDULE_H
#define LOAD_FLIGHT_SCHEDULE_H

#include <vector>
#include <string>
using namespace std;

const double NoFlight = -1;
// schedule must be size needed, eg 60
void load_excel_flight_schedule(string filename, vector<double> &schedule);
void load_flight_schedule(      string filename, vector<double> &schedule);

#endif
```

load_flight_schedule.cpp

```

#include "load_flight_schedule.h"
#include <iostream>
#include <fstream>
#include <map>
using namespace std;

void load_excel_flight_schedule(string filename, vector<double> &schedule)
{
    ifstream fin(filename.c_str());
    if(!fin) { cout << "unable to open flight schedule file" << endl; exit(1); }

    // Associate letters with avg # of people
    map<string, double> name2people;
    string name;
    fin >> name;
    while ("Time" != name)
    {
        double avg_people;
        fin >> avg_people;
        name2people[name] = avg_people;
        fin >> name;
    }

    string trash;
    fin >> trash; // "Flight"
    double minute;
    while (fin >> minute)
    {
        string flight_name;
        double flight_avg_people;
        fin >> flight_name;
        if(name2people.find(flight_name) == name2people.end())
        {
            flight_avg_people = NoFlight;
        }
        else
        {
            flight_avg_people = name2people[flight_name];
            fin >> trash;
        }
        if((int)minute >= schedule.size())
        {
            cerr << "Attempt to store outside of range of schedule size averted,"
                 << " but bailing out. goodbye. (" << (int)minute << ")" << endl;
            exit(1);
        }
        schedule[(int)minute] = flight_avg_people;
        //schedule.push_back(flight_avg_people);
    }
    fin.close();
}

void load_flight_schedule(string filename, vector<double> &schedule)
{
    const double FileNoFlight = 0, InternalNoFlight = NoFlight;

    ifstream fin(filename.c_str());
    if(!fin) cout << "unable to open flight schedule file" << endl;

    int minute;
    double avg_people;
    while (fin >> minute && fin >> avg_people)
    {
        if (FileNoFlight == avg_people)
            avg_people = InternalNoFlight;
        schedule[minute] = avg_people;
        //schedule.push_back(avg_people);
    }
    fin.close();
}

```

}

dist_merge.cpp

```
// Build a distribution of people coming in from the distributions for
// certain sized planes and the schedule using these kinds of planes.

#include "load_flight_schedule.h"
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
#include <map>
using namespace std;

void load_flight_dists(string filename,
                      map<double, vector<double> > &flight_dists)
{
    /* Tab-delimited file formatted as:
    *
    * IDENTITIES:
    *
    * Seats\tAvg people\tFlights\tCorr coeff
    * s\ta\tf\tc
    * ...
    *
    * Dists
    * 34 seats
    * Minute\tPeople
    * m\tp
    * ...
    * total\t<total>
    *
    * <repeat with 46 seats and so on>
    */
    const bool debug = false;

    ifstream fin(filename.c_str());
    if(!fin) {
        cout << "Unable to open flight dists file" << endl;
    }

    string buffer;
    // Read IDENTITIES
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;
    fin >> buffer;

    // Make vector of seats for kinds of planes
    map<int, double> flight_seats;
    int seats;
    fin >> seats; // read first "Seats entry"
    while(seats != 0)
    {
        double avg_seats;
        fin >> avg_seats;
        flight_seats[seats] = avg_seats;
        fin >> buffer; // "Flights"
        fin >> buffer; // "Corr coeff"

        fin >> seats; // read next "Seats" entry
    }

    while(fin >> seats)
    {
        int minute=0;
        double num_people=0;
        vector<double> dist; // distribution of people for this flight
    }
}
```

```

        if(debug)
            cout << "Reading dist for flight with seating for: "
                  << seats << ", avg: " << flight_seats[seats] << endl;

        fin >> buffer >> buffer >> buffer; // "seats", "Minute", and "People"
        do
        {
            fin >> minute;
            fin >> num_people;
            dist.push_back(num_people);
            if(debug)
                cout << minute << ", " << num_people << endl;
        } while (minute < 90);
        if (dist.size() != 91)
            cout << "Warning: # records mismatch for flight with "
                  << num_people << "avg num people:"
                  << "read: " << dist.size() << " expected: 91" << endl;
        fin >> buffer >> buffer; // "total" and the total
        flight_dists[flight_seats[seats]] = dist;
    }

    fin.close();
}

void add_flight_dist(const vector<double> &flight_dist,
                    int start_time,
                    vector<double> &dist)
{
    const bool debug = false;
    if (flight_dist.size() + start_time > dist.size())
        cerr << "flight_dist (" << flight_dist.size()+start_time
              << ") continues past global dist (" << dist.size() << ")" << endl;
    for(unsigned int i=0; i<flight_dist.size(); i++)
    {
        if (debug)
            cout << dist[i+start_time]
                  << " (" << i+start_time << ") -> "
                  << dist[i+start_time] + flight_dist[i] << endl;
        dist[i+start_time] += flight_dist[i];
    }
}

vector<double>
build_distribution(map<double, vector<double> > &flight_dists,
                  const vector<double> &flight_schedule)
{
    const bool debug = false;
    const double NoFlight = -1;
    const int flight_dist_offset = 120; // dist start 120mins before takeoff
    // Assumes no flight dists starting earlier than 120mins before schedule.
    vector<double> dist(flight_dist_offset+flight_schedule.size(), 0);
    for(unsigned int i=0; i<flight_schedule.size(); i++)
    {
        if (NoFlight != flight_schedule[i])
        {
            if (debug)
                cout << "Adding in flight (" << i << ") "
                      << flight_schedule[i] << endl;
            const double key = flight_schedule[i];
            map<double, vector<double> >::iterator flight_dist
                = flight_dists.find(key);
            if (flight_dists.end() != flight_dist)
                add_flight_dist((*flight_dist).second, i, dist);
            else
                cerr << "Unable to find flight dist for flight with avg people: "
                      << key << endl;
        }
    }
    return dist;
}

```

```
void print_dist(const vector<double> &dist)
{
    //      cout << "Distribution, one line per minute:" << endl;
    for(unsigned int i=0; i < dist.size(); i++)
        cout << dist[i] << endl;
    //      cout << "End Distribution" << endl;
}

int main(int argc, char **argv)
{
    map<double, vector<double> > flight_dists;
    vector<double> flight_schedule(60, -1); // init with -1 for every min
    vector<double> dist;
    bool excel_schedule = true;

    if(argc < 3) {
        cout << "Usage: dist_merge plane_dists schedule [true], true for build_schedule
input" << endl;
        return(-1);
    }
    if(4 == argc)
        excel_schedule = false;
    else if (2 == argc)
        excel_schedule = true;

    const string schedule_filename = argv[2];
    const string plane_dists_filename = argv[1];

    load_flight_dists(plane_dists_filename, flight_dists);
    if(excel_schedule)
        load_excel_flight_schedule(schedule_filename, flight_schedule);
    else
        load_flight_schedule(schedule_filename, flight_schedule);
    dist = build_distribution(flight_dists, flight_schedule);
    print_dist(dist);
}
```

create_dists_incrementally.sh

```
export TARGET_DIR="dist_ends_incremental/"
rm -rf $TARGET_DIR
mkdir $TARGET_DIR
OFFSET=10;
for ((i=1; i<=60; i++));
do
    head -n $i schedule > schedule_part \
        && ./dist_merge plane_dists schedule_part true >
"$TARGET_DIR/dist_ends_`expr $OFFSET + $i`";
done
rm schedule_part
```

create_dists_incrementally_excel.sh

```
export TARGET_DIR="dist_ends_incremental/"
export HEADER_LINES_IN_SCHEDULE=11
rm -rf $TARGET_DIR
mkdir $TARGET_DIR
OFFSET=10;
for ((i=1; i<=60; i++));
do
    head -n `expr $i + $HEADER_LINES_IN_SCHEDULE` schedule > schedule_part
\
    && ./dist_merge plane_dists schedule_part >
"$TARGET_DIR/dist_ends_`expr $OFFSET + $i`;
done
rm schedule_part
```

merge_dist_ends_incremental.cpp

```
// Combine the distributions of many files into a matrix of all, with
// rows being minutes and columns the dists.
// List of files given on the command line.

#include <iostream>
#include <fstream>
#include <vector>
using namespace std;

int main(int argc, char** argv)
{
    vector< vector<double> > dists;
    const int num_dists = argc-1;

    if(argc < 2)
    {
        cout << "Usage: merge_dist_ends_incremental FILE1 FILE2 ..." << endl;
        return 1;
    }

    // Read in all distributions
    for(int i=1; i<num_dists+1; i++)
    {
        ifstream fin(argv[i]);
        if(!fin) {
            cerr << "Couldn't open \"" << argv[i] << "\"." << endl;
            continue;
        }
        vector<double> dist;
        double people_rate;
        while(fin >> people_rate)
            dist.push_back(people_rate);

        fin.close();
        dists.push_back(dist);
    }

    // Print out in matrix form
    // Assumes all dists the same size.
    for(int min=0; min<dists[0].size(); min++)
    {
        cout << min << "\t";
        for(vector< vector<double> >::iterator it=dists.begin();
            dists.end() != it;
            it++)
        {
            cout << (*it)[min] << "\t";
        }
        cout << endl;
    }

    return 0;
}
```

merge_catchup_times.cpp

```
// Loads the results of create_catchup_times and outputs the worst
// catchup times for each processing rate n the form"rate\ttime".

#include <iostream>
#include <fstream>
#include <vector>
using namespace std;

vector< pair<double, int> > build_final_catchup_times(string catchup_times_filename)
{
    vector< pair<double, int> > final_catchup_times;

    ifstream fin(catchup_times_filename.c_str());
    if(!fin) {
        cerr << "Unable to open catchup times outputfile \""
              << catchup_times_filename << endl;
        exit(1);
    }

    string trash;
    while(fin >> trash && fin >> trash)
    {
        double cur_processing_rate;
        fin >> cur_processing_rate;

        string minute;
        int catchup_time, worst_catchup_time = -1;
        fin >> minute;
        while("END" != minute)
        {
            fin >> catchup_time;
            if (worst_catchup_time == -1 && catchup_time != -1)
                worst_catchup_time = catchup_time;
            else if (catchup_time < worst_catchup_time)
                worst_catchup_time = catchup_time;

            fin >> minute;
        }
        pair<double, int> rate_catchup_time;
        rate_catchup_time.first = cur_processing_rate;
        rate_catchup_time.second = worst_catchup_time;
        final_catchup_times.push_back(rate_catchup_time);

        fin >> trash >> trash;
    }

    fin.close();
    return final_catchup_times;
}

void print_final_catchup_times(vector< pair<double, int> > final_catchup_times)
{
    for(vector< pair<double, int> >::iterator it = final_catchup_times.begin();
        final_catchup_times.end() != it;
        it++)
    {
        cout << (*it).first << "\t" << (*it).second << endl;
    }
}

int main(int argc, char **argv)
{
    if(argc != 2)
    {
        cout << "Usage: merge_catchup_times create_catchup_times_outputfile"
              << endl;
        return 1;
    }
    const string catchup_times_filename = argv[1];
```

```
vector< pair<double, int> > final_catchup_times
    = build_final_catchup_times(catchup_times_filename);
print_final_catchup_times(final_catchup_times);

return 0;
}
```


find_scan_behind_times.cpp

```
// Reads in output from merge_dist_ends_incremental and the processing
// rate and finds the amount of time before takeoff that scanning is
// no longer behind.

// Output: "<plane #>\t<catches up mins before takeoff, -1 if never behind>"

#include "load_flight_schedule.h"
#include <math.h>
#include <iostream>
#include <fstream>
#include <algorithm>
using namespace std;

vector< vector<double> > load_dists(const string &people_dists_filename)
{
    // Assumes minute column and 60 time columns
    const int num_flight_cols = 60;
    vector< vector<double> > dists(60);
    int min;
    double person_rate;
    ifstream fin(people_dists_filename.c_str());
    if(!fin)
    { cerr << "Unable to open people dists matrix file." << endl; exit(1); }

    while (fin >> min)
    {
        for(int flight_col=0; flight_col<num_flight_cols; flight_col++)
        {
            fin >> person_rate;
            dists[flight_col].push_back(person_rate);
        }
    }

    fin.close();
    return dists;
}

// Finds last time caught back up, returns -1 if never got behind.
int calc_time_no_longer_behind_single(vector<double> &dist,
                                     const double process_rate)
{
    double prev_behind_rate=0, cur_behind_rate=0;
    int last_caught_back_up=0;
    for(int min=0; min<dist.size(); min++)
    {
        cur_behind_rate = max(0.0, prev_behind_rate + dist[min] - process_rate);
        if(prev_behind_rate > 0 && cur_behind_rate == 0)
        {
            last_caught_back_up = min;
        }
        prev_behind_rate = cur_behind_rate;
    }

    if(cur_behind_rate > 0) // still behind even though we've past end of time
        last_caught_back_up=dist.size()+(int) ceil(cur_behind_rate/process_rate);
    else if(last_caught_back_up == 0)
        last_caught_back_up = -1;
    // if(last_caught_back_up < -1)
    //     cerr << "eh?" << endl;
    return last_caught_back_up;
}

vector<int> calc_time_no_longer_behind(vector< vector<double> > &dists,
                                     const double process_rate)
{
    vector<int> caught_up_times;
    for(int i=0; i<dists.size(); i++)
    {
```

```

        caught_up_times.push_back(calc_time_no_longer_behind_single(dists[i],
                                                                    process_rate));
    }
    return caught_up_times;
}

vector<int> calc_catch_up_times_before_flights
    (vector<int> &time_no_longer_behind,
     vector<double> &flight_schedule)
{
    const int time_offset = 120;
    vector<int> catch_up_before_flight;
    if(time_no_longer_behind.size() != flight_schedule.size())
    { cerr << "Mismatch in vector sizes." << endl; exit(1); }

    for(int takeoff_time=0; takeoff_time<flight_schedule.size(); takeoff_time++)
    {
        if(flight_schedule[takeoff_time] == NoFlight)
            continue;
        int time_diff = (time_offset + takeoff_time
                        - time_no_longer_behind[takeoff_time]);

        if(time_no_longer_behind[takeoff_time] == -1)
            time_diff = -1; // to show didn't fall behind

        catch_up_before_flight.push_back(time_diff);
    }

    return catch_up_before_flight;
}

void print_results(const vector<int> &catch_up_before_flight)
{
    for(int i=0; i<catch_up_before_flight.size(); i++)
        cout << i << "\t" << catch_up_before_flight[i] << endl;
}

int main(int argc, char **argv)
{
    if(argc != 5)
    {
        cout << "Usage: find_flight_behind_times people_processing_rate < -excel | -build
> flight_schedule people_dists_matrix_file" << endl;
        return -1;
    }
    const double process_rate = strtod(argv[1], NULL);
    bool excel_schedule;
    if(!strcmp("-excel", argv[2]))
        excel_schedule = true;
    else if(!strcmp("-build", argv[2]))
        excel_schedule = false;
    else
    { cout << "unknown option: \"" << argv[3] << "\"" << endl; return -1; }
    const string flight_schedule_filename = argv[3];
    const string people_dists_filename = argv[4];

    vector< vector<double> > dists = load_dists(people_dists_filename);
    vector<int> time_no_longer_behind
        = calc_time_no_longer_behind(dists, process_rate);

    const int flight_schedule_length = 60;
    vector<double> flight_schedule(flight_schedule_length);
    if(excel_schedule)
        load_excel_flight_schedule(flight_schedule_filename, flight_schedule);
    else
        load_flight_schedule(flight_schedule_filename, flight_schedule);

    vector<int> catch_up_before_flight
        = calc_catch_up_times_before_flights(time_no_longer_behind,

```

```
    flight_schedule);  
    print_results(catch_up_before_flight);  
    return 0;  
}
```

create_catchup_times.cpp

```
// This program allows one to easily run try_catchup_times on a range
// of processing rates. It outputs "NEXT rate: <processing rate>\n",
// then the output from try_catchup_times, then "END rate: <processing rate>\n"

#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <iostream>
#include <sstream>
using namespace std;

string dtostring(const double x)
{
    ostringstream str;
    str << x;
    return str.str();
}

int main(int argc, char **argv)
{
    if(argc != 7)
    {
        cout << "Usage: try_catchup_times process_rate_begin process_rate_end
process_rate_inc < -excel | -build > flight_schedule people_dists_file" << endl;
        return -1;
    }

    const double process_rate_begin = strtod(argv[1], NULL),
        process_rate_end = strtod(argv[2], NULL),
        process_rate_inc = strtod(argv[3], NULL);

    const string excel_schedule = argv[4],
        flight_schedule_filename = argv[5],
        people_dists_filename = argv[6];
    if(excel_schedule != "-excel" && excel_schedule != "-build")
    {
        cerr << "unknown option: \"" << excel_schedule << "\"" << endl;
        return 1;
    }

    const string find_scan_behind_times_execname = "./find_scan_behind_times";

    for(double process_rate_cur = process_rate_begin;
        process_rate_cur <= process_rate_end;
        process_rate_cur += process_rate_inc)
    {
        pid_t pid;
        int status;
        cout << "NEXT rate: " << process_rate_cur << endl;

        pid = fork();
        if(-1 == pid) { perror("fork"); exit(1); }
        if( 0 == pid)
        {
            execlp(find_scan_behind_times_execname.c_str(),
                find_scan_behind_times_execname.c_str(),
                dtostring(process_rate_cur).c_str(),
                excel_schedule.c_str(),
                flight_schedule_filename.c_str(),
                people_dists_filename.c_str(),
                NULL);

            cerr << "Unable to find program \""
                << find_scan_behind_times_execname << "\"" << endl;
            exit(1);
        }
        else
        {
            waitpid(pid, &status, 0);
            if(status != 0)

```

```
        cerr << "WARNING: " << find_scan_behind_times_execname  
              << " ran into problems on processing rate "  
              << process_rate_cur << endl;  
        cout << "END rate: " << process_rate_cur << endl;  
    }  
}  
  
return 0;  
}
```

USAGE

1. Build a schedule, using either excel, to layout by hand,
or `build_schedule`, which does uniform or bunch end-of-hour scheduling.
2. Build dist of people processing rates
For all of schedule, use `dist_merge`.
To take only first, then first two, then first three, ... use
`create_dists_incrementally.sh` (the schedule needs to be named `./schedule`
and the plane dists needs to be named `./plane_dists`)
4. If created incremental dists, run `merge_dist_ends_incremental` on the files
created by `create_dists_incrementally.sh` which creates a matrix
of the incrementally added flights and their incoming person rates.
5. `find_flight_behind_times` gives the time before flight for each flight
that the baggage scanning last caught back up with incoming demand
given a certain processing rate.
`create_catchup_times` and `merge_catchup_times` runs `find_flight_behind_times`
over a range of processing rates and outputs the smallest time before
flight that each processing rate yielded.

Makefile

```
CPP = g++
BINARIES = build_schedule dist_merge merge_dist_ends_incremental \
            find_scan_behind_times create_catchup_times merge_catchup_times

default: ${BINARIES}

load_flight_schedule.o: load_flight_schedule.h load_flight_schedule.cpp
    ${CPP} ${CPPFLAGS} -c load_flight_schedule.cpp

dist_merge: load_flight_schedule.o dist_merge.cpp
    ${CPP} ${CPPFLAGS} -o dist_merge dist_merge.cpp load_flight_schedule.o

find_scan_behind_times: find_scan_behind_times.cpp load_flight_schedule.o
    ${CPP} ${CPPFLAGS} -o find_scan_behind_times find_scan_behind_times.cpp \
    load_flight_schedule.o

clean:
    rm -f ${BINARIES} load_flight_schedule.o
```