CS CAPSTONE PROGRESS REPORT

MAY 16, 2017

BOEING OPTIMIZATION PROGRESS REPORT

PREPARED FOR

BOEING

MARISSA BEDFORD _____ _____
Signature *Date*

PREPARED BY

GROUP 45

THE BOEING RESEARCH PROJECT TEAM

MICHAEL CHAN _____ _____
Signature *Date*

BRANDON TO _____ _____
Signature *Date*

MIKE TRUONG _____ _____
Signature *Date*

Abstract

A focused workforce is able to accomplish multiple tasks within the time allocated. The direction of this project is to define a plan that determines the most efficient way for salaried Boeing employees to get to work on time to minimize the disruption to both their production and personal lives. This project will be experimental using the technique of a case study to examine Boeing employees over a time duration of two work days. Our plan is to use data collection techniques and analytics to develop an algorithm that can be used as demands change. The data collection involves evaluating instances that can be improved within employee work schedules with the use of surveys, questionnaires, observation, and records. Combining faculty documents with the data obtained from the employees of Boeing Everett the findings will determine a method to improve efficiency. At length the results will boost productivity for Boeing Everett.

CONTENTS

1 Purpose and Goals

The Boeing Everett Factory struggles with inefficient arrivals and departure times, causing a direct correlation to the employees ability to balance production efficiency and time spent with their families. With a large population of approximately 30,000 employees, it is common to have many instances of dense amounts of workers who would attempt to arrive/depart from the Everett Facility at the same time, bringing about unnecessary traffic congestion. Boeing has identified the main cause of this being from unorganized shift times of their salaried employees. It is imperative that large factories like Boeing remain efficient to not only ensure that the optimal amount of progress is made at the work environment but also, ensuring that their employees are content and commute safely to their homes.

Our client has noted that there is no designated budget at this time in consideration for maintaining an application. In result, the purpose of this project is to research and design an algorithm to increase optimize the shift start and stop times of the salaried employees and management team. The qualities that we are working to accomplish in our algorithm are efficiency, sustainability, and versatility. To maintain efficiency in all areas of Boeing's production, our algorithm must ensure that changing the start and stop times of shifts will not hinder the productivity of employees. Sustainability of this algorithm is crucial to ensure that this is a long term solution rather than creating issues else where within the production. Lastly, versatility of the algorithm is key as we must consider the possibility of new objectives that the company may want to accomplish.

2 Problems and Solutions (Brandon and Mike)

Our current problem is to continue working on the algorithm for Boeing. It did not match the specifics that our client wanted. We presented them with a linear program and also an analysis in graphical form. Both of these showed data that result in identifying the problem, however it did not solve the main issue that Boeing wanted. The issue was to come up with a recommendation that will alter shift times for employees; this alteration should increase work efficiency and also clear the parking lot traffic issue. Our solution is to take the week to analyze data more deeply.

Regarding some of the problems within writing the white paper analysis, Boeing had requested that the final product would be able to determine/analyze the optimal percentage of salaried employees to move into the second shift in order to decrease congestion within the first shift. This analysis, however, required the usage of parking data as well as discrete information about the program groups of both salaried and non-salaried workers. After the tour of the Boeing Everett Factory, we were given a detailed compilation regarding the categorization of employees and were able to map approximately where they worked in the factory. This made it quite simple to draw assumptions of which lots they were most likely going to use. As the deadline drew near, we were able to compile all of the necessary values except for one. During the week of May 8th to the 11th, we had requested data disclosing the amount of stalls within every Boeing parking lot nearby the factory. Unfortunately on the 11th, our client notified us that she and a few others involved in the project were unable to do so since their group did not have authorization and access to that data from the Yards and Parking department. With the absence of this important information, we were only able to complete the write up in the form of a mock-up that contained steps to determine the percentage requested and the constraints used in the mock-up.

One issue with using the provided data in the excel sheet is that there are too many cells for the free Excel Solver to run with. If there are over 200 possible changing variables, Solver will refuse to run unless we buy the upgraded

version. For any linear programs using data pulled from Boeing, the data will have to be simplified for the purpose of this project.

3 Current Progress (Brandon and Mike)

We are working on specifying our algorithm for Boeing. Boeing would like us to focus more on giving a recommendation to actually altering the start times of their salary employees. They stated that what we have shown is helpful but would like us to go farther and alter the shift times ourselves to see if it can corresponded to lower population for the parking lot issue. This is equivalent as improving efficiency at Boeing. We were given a week to do this.

Chancing the excel sheet to examine only first shift, we are looking to alter 24,918 individuals. Our client also told us that the ratio of salary to hour workers is 8:1. This leads to 3,315 salary workers compared to 21,603 hourly workers. 3,315 salary workers go through the shift of 5:48 AM until 2:18 PM. This gives us 10 hours and 6 minutes to work with. With the staggering method of intervals of 20-30 minutes we could shift a percentage of 3,315 over the duration of the first three hours of 1st shift. For some this will affect their departure time. We want at least 829 salary per hour for the first four hours. Pushing these hours to .5 increase would push to number to 553 salary per hour for a duration of 6 hours; this ranges from 6:00 AM to Noon. What this allows is sufficient work flow which happens in the morning for the first shift. Boeing will than need to examine individuals who need to be there at our recommend time interval to improve their work efficiency. This means moving 16.78% of the salaried employees from 6:00 AM to Noon with intervals of half an hour.

4 What's Left (Mike)

The next meeting with Boeing will be May 15. At this meeting we will be suggesting the final recommendation for Boeing. This meeting should include a specific discrete value of shift times that will be altered; the change should be helpful to Boeing in increasing their work efficiency. What we plan to do is look more in detail the excel that Boeing provided us. Specifically looking at how we can move a percentage of individuals who work first shift to later shifts in the day. The bulk of the issue at Boeing ranges the first five hours of the early shift. Moving a percentage of these workers will help stagger the parking lot issue. Expo is May 19th, we will use this time as a team to prepare. This includes practicing our pitch for the event and also prepare how we would like to display our research project to everyone.

5 Logical Analysis (Brandon)

The aim of this write up was to determine a broken down close estimate of the available parking stalls available for the salaried employees after first accurately assessing enough spaces to be filled by the non-salaried. This was a critical constraint that must be assessed since the non-salaried workers have already been standardized within their shifts. Once the analysis has been done regarding the allocation of non-salaried workers, the left over spaces within each lot totaled together are the spaces that can be distributed to the salaried workers. Obtaining the necessary values and following the steps will provide a percentage that will suggest the amount salaried employees that surpass a specific lot's occupancy and should be allocated to the second shift.

1) We will first obtain a scope of the amount of parking stalls within each lot.
2) Find a correlation between non-salaried employees and the programs that they working within.

3) Given the maximum amount of parking stalls in a lot and the general location and population for each project, we can now proceed on to assigning them onto a lot. However, this is easier said than done. Assumptions must be drawn since parking lots are always changing due to a multitude of factors during a given day.
4) Ensure that the lot can fill this given amount of people. If the distribution of non-salaried employees fill up over 90 percent of one lot for example, this is not feasible.
5) Subtract the maximum occupancy of the lot by the number of estimated non-salaried workers are stationed to park within that lot. The result of this would be the amount of unoccupied spaces remaining within the given lot.
6) Refer to the spreadsheet of the amount of salaried workers and determine program associated with location.
7) Allocate to the given lots closest to their location.
8) If the location supports the amount of workers, then problem is resolved. If the given amount of workers for one lot exceeds the occupancy, determine the sum of the overall salaried employees that have exceed the given space.
9) We should now have a count that is the sum of how many salaried workers exceed the total amount of spaces in the lots. Determine percentage of salaried workers that should be recommended to be moved to second shift.

There were several constraints that we limited our analysis to. The first constraint was that we were going to solidly focus on the first shift times. Being the most desirable time for employees and containing a significant bulk of the factory's work force, we want to reduce congestion from this time by allocating a percentage of salaried workers to the second shift. The second constraint was that in this analysis, we were going to focus on the factory itself because it contains the most non-salaried employees. As mentioned before, this workforce already has an existing standardized shift time which means that they have to be on site at that time. Another constraint had to be made later at this point because we were unable to acquire the parking stalls information. There was an initial attempt to design a rough estimate but since square footage of the lots weren't disclosed either, we left it as a dummy variable. Since our clients emphasized that their two East lots (E5 and E6) nearby the factory were the troubled lots, this was what we focused on. After analyzing the factory map, we drew the assumption that the 777 and 787 programs were most likely to utilize these lots due to proximity.

After compiling the values of both programs, we were able to discover that both programs together contained about 3,100 people with about 91.2 percent of them being non-salaried. We concluded that the lots must be able to accommodate that much with the remaining 8.9 percent considered as the maximum to be allocated to second shift. The mock-up equation breaks it down as follows: We would use the number of salaried employees as the numerator and the number of stalls left as the denominator. Dividing this number would show us whether or not there is a percentage of an excess amount of salaried employees during a given shift. We would then take the excess plus a buffer size (possibly 10 percent of salaried workers) to allocate to the second shift. It is not required to calculate a buffer but it is recommended in order to make sure there is a less of a chance to run into congestion.

6 WRITE-UP DRAFT

The Data that Boeing provided us was in an excel sheet that was enumerated with the amount of employees that worked at the facility at Everett. This excel was also broken down to subcategories called shifts. These shifts are broken down by different time stamps of when employees would come into work. Using this sheet we came up with different recommendations that Boeing could in act in their facility to improve work efficiency. One being to stagger shift times for the first shift. Boeings' first shift includes 24,918 individuals. First shift occurs from 5:48 AM until 2:18 PM. The issue that Boeing needed us to address was that parking at the facility; the issue caused frustration and was hindering work efficiency. Michael Callaghan, one of our clients told us that the ratio of salaried employees versus hourly workers was an estimate of 1:8 ratio. Using cross multiplication we were able to find out that of the 24,918 individuals who work at Boeings' first shift that 3,315 were salaried. With an estimate of 21,603 hourly workers. The issue that needs to be addressed is the high alert of parking lot spots being filled from the 6:00 AM period to 10:00 AM. If there is a huge influx is 3,315 salary workers coming in at the same time as hourly it would make sense why red flags are appearing. Cutting the first shift salary employees by 16.78% allows 553 salary workers to entire per hour compared to the usual 3,315. In intervals of half hour starting at 5:48 AM the next handful of 553 would come into work; this would continue all the way until noon. Effectively moving the 1st shift salary workers across the day without harming work efficiency but increasing accessibility for the Boeing parking lot.

6.1 Linear Programming

A linear program takes decision variables, which represent the decisions to be made, to represent an objective function which can be minimized or maximized based on a set of constraints. The function looks like so:

(Minimize/Maximize) Coefficient * Variable + Coefficient * Variable + + Coefficient * Variable

If you want to maximize the cost of 3 items, x1, x2, x3, and they sell for $1, $2, $4 respectively, it would look like:

$$Maximize\, 1x1 + 2x2 + 4x3$$

Constraints limit the amount of each variables. They can be inequalities ($\leq, \geq, =$) and Excel solver allows constraints such as forcing binary or integer numerals. A constraint that says "the budget is 200" will look like:

$$1x1 + 2x2 + 4x3 \leq 200$$

Another common constraint is sign restrictions, which just say whether or not a number can be negative or positive:

$$x1 \geq 0$$

Put all of this together, and you have a linear program, as seen below.

$$\begin{gathered} Maximize\, 1x1 + 2x2 + 4x3 \\ SubjectTo: \\ 1x1 + 2x2 + 4x3 \leq 200 \\ x1 \geq 0 \end{gathered} \tag{1}$$

So how do you reach an optimal solution using linear programming? If we look at a program graphed out visually, it will make more sense. Here is an example program:

$$
\begin{aligned}
&Maximize\ x + y \\
&Subject\ To: \\
&x + 2y \ \geq 2 \\
&X \leq 3 \\
&Y \leq 4 \\
&X \geq 0 \\
&Y \geq 0
\end{aligned}
\tag{2}
$$

The LP is graphed out below. The shaded blue region is the area bound by the constraints, and is called the feasible region, because any point within the area is a possibly solution, because it satisfies all constraints. The other areas are part of the in-feasible region, because they violate the constraints. Therefore the optimal solution must be within the blue area. Any point within the area contains a objective value. The point (2,1) has the objective value of 2+1=3. This is not the optimal value, because we are trying to optimize the solution, and there are more objective values possible we have not tested. To do this, we have to find the largest objective value possible, and in this case, it is (3,4), which gives the objective value of 7.

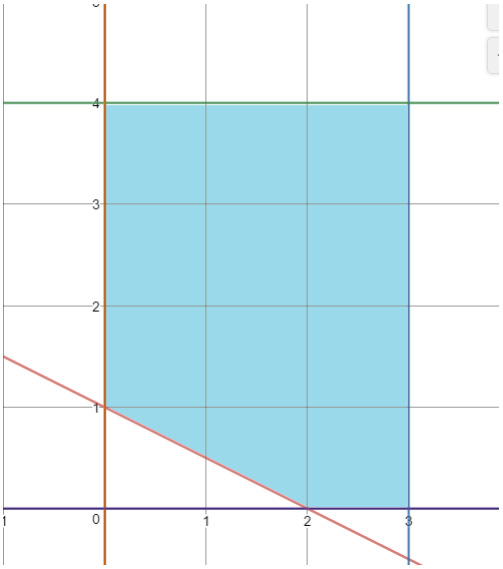

Fig. 1: The example program graphed out.

x1	x2	x3	x4	x5	x6	x7				
0	0	0	0	0	0	0				
Constraints										
1			1	1	1	1	0	>=	5000	Mon
1	1			1	1	1	0	>=	4000	Tues
1	1	1			1	1	0	>=	3000	Wed
1	1	1	1			1	0	>=	2000	Thu
1	1	1	1	1			0	>=	4000	Fri
	1	1	1	1	1		0	>=	5000	Sat
		1	1	1	1	1	0	>=	7000	Sun
						total work	0			
//lets pretend we need half salaried										
Numbers required:										
Workers needed:			Max spots							
5000			10000							
4000			8000							
3000			6000							
2000			4000							
4000			8000							
5000			10000							
7000			14000							

Fig. 2: The program before running.

Now we have some intuition on how linear programs work. We want to list the corner points within the feasible region and find the best objective value out of all corners, but what if there were lots of corners, and how do we find them? Finding them is a matter of finding where our constraints intersect, for each constraint, you get another dimension and another constraint that intersects. 2 dimensions means 2 constraints intersect, 3 means 3 constraints intersect, and so on. Finding where these constraints intersects means solving a system of equations using Linear reduction/Gaussian elimination. Computers can do this for us easily, so we don't have to worry about that. If there are a lot of corners and the graph is now a polyhedron with hundreds of corners, we need a method to quickly and efficiently find the optimal corners. The Simplex Method is a efficient method for solving linear programs, and is the one I used in Excel's Solver. In fact, most commercially available linear solvers will use the Simplex Method. To keep things simple, this method traverses corner to corner until it finds the optimal solution. Knowing about this method is important in the understanding of linear programming.

6.1.1 Day scheduler

This is a method using linear programming to final an optimal solution for finding the number of employees to have come in at specific days based on their work schedules. The Constraints cells shows the potential work schedules that a set of employees may have. This is an example matrix that shows 7 different work schedules where employees choose to work 5 days in a row with 2 off, with this happening on each possible day of the week. There can potentially be more or less schedules to adapt to the flexibility of salaried work weeks. The x1, x2, .. variables are the number of workers who start on the day they represent. X1 is Monday, x2 is Tuesday, Sunday is the last day at x7. The numbers to the right of the ¿= signs are constraints on the numbers required. There is no way to know exactly how many people are needed for daily duties to be accomplished, so this is just a theoretical number. The number to the left of the ¿= sign is the number of workers for each day that are working. Total workers is the number of starting workers each day summed. Constraints:

All values must be integers;

Workers within each day must be greater than the required amount;

Workers cant be above the max spots; One feasible solution:

x1	x2		x4	x5	x6	x7				
0	0	2000	0	4000	1000	0				
Constraints										
1			1	1	1	1	5000	>=	5000	Mon
1	1			1	1	1	5000	>=	4000	Tues
1	1	1			1	1	3000	>=	3000	Wed
1	1	1	1			1	2000	>=	2000	Thu
1	1	1	1	1			6000	>=	4000	Fri
	1	1	1	1	1		7000	>=	5000	Sat
		1	1	1	1	1	7000	>=	7000	Sun
						total work	7000			

Fig. 3: The program after running.

We can create even more complex programs with more realistic constraints and schedules. The schedule can also be even more detailed, allowing us to optimize within each day by hour as well. There are also constraints that need some more looking at, such as potential max spots. In reality, it would be total spots minus hourly workers, and salaried workers would fill the remaining.

6.1.2 Lot Preferences

	Lot 1	Lot 2	Lot 3	Lot 4	Lot 5	Lot 6	Lot 7	Lot 8	Lot 9	Lot 10
Employee 1	3	2	1	4	6	5	8	9	10	7
Employee 2	5	3	2	6	1	7	9	8	4	10
Employee 3	10	8	1	9	7	4	3	6	2	5
Employee 4	7	3	2	9	5	4	8	6	1	10
Employee 5	1	3	6	8	5	2	9	10	7	4
Employee 6	4	9	1	5	6	8	2	7	10	3
Employee 7	2	1	10	9	5	3	6	8	4	7
Employee 8	6	5	1	3	2	4	7	8	9	10
Employee 9	8	9	10	5	4	3	2	1	6	7
Employee 10	9	10	3	2	5	4	1	7	8	6
Employee 11	7	3	5	2	9	8	1	10	4	6
Employee 12	6	5	1	9	10	2	3	4	7	8
Employee 13	6	8	10	9	1	2	3	4	5	7
Employee 14	6	3	5	9	1	2	10	4	8	7

Assignments	Lot 1	Lot 2	Lot 3	Lot 4	Lot 5	Lot 6	Lot 7	Lot 8	Lot 9	Lot 10	Total	Preferenc
Employee 1	0	0	0	0	0	0	0	0	0	0	0	0
Employee 2	0	0	0	0	0	0	0	0	0	0	0	0
Employee 3	0	0	0	0	0	0	0	0	0	0	0	0
Employee 4	0	0	0	0	0	0	0	0	0	0	0	0
Employee 5	0	0	0	0	0	0	0	0	0	0	0	0
Employee 6	0	0	0	0	0	0	0	0	0	0	0	0
Employee 7	0	0	0	0	0	0	0	0	0	0	0	0
Employee 8	0	0	0	0	0	0	0	0	0	0	0	0
Employee 9	0	0	0	0	0	0	0	0	0	0	0	0
Employee 10	0	0	0	0	0	0	0	0	0	0	0	0
Employee 11	0	0	0	0	0	0	0	0	0	0	0	0
Employee 12	0	0	0	0	0	0	0	0	0	0	0	0
Employee 13	0	0	0	0	0	0	0	0	0	0	0	0
Employee 14	0	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0		0
Required	1	1	1	1	2	1	2	2	2	1		

Fig. 4: The program before running.

The top table represents 14 employees with lot preferences ranging from 1-10, with 1 being most preferred. The second table contains the changing variables that indicate where each employee will park. The total row on the right will equal the total workers that park within each lot, which should be one for each lot since workers can't park at two lots at once. The total column is total lots each worker parked in. The goal is to minimize each preference so each worker gets the highest happiness while meeting the constraints on where each worker must park, remember that the lower the preference, the better. This program can also be written in reverse, with 10 being max preference and the objective value

is maximized instead of minimized. The required row at the bottom indicates this by saying the minimum amount of workers parking at each lot. The black bordered box in the bottom right is the objective value that gets minimized.

Here are the constraints:

There can only be one lot an employee parks in;

Each cell can only be binary, ensures that total is 1;

The total in each lot must equal the required;

Here is a optimal solution:

	Lot 1	Lot 2	Lot 3	Lot 4	Lot 5	Lot 6	Lot 7	Lot 8	Lot 9	Lot 10
Employee 1	3	2	1	4	6	5	8	9	10	7
Employee 2	5	3	2	6	1	7	9	8	4	10
Employee 3	10	8	1	9	7	4	3	6	2	5
Employee 4	7	3	2	9	5	4	8	6	1	10
Employee 5	1	3	6	8	5	2	9	10	7	4
Employee 6	4	9	1	5	6	8	2	7	10	3
Employee 7	2	1	10	9	5	3	6	8	4	7
Employee 8	6	5	1	3	2	4	7	8	9	10
Employee 9	8	9	10	5	4	3	2	1	6	7
Employee 10	9	10	3	2	5	4	1	7	8	6
Employee 11	7	3	5	2	9	8	1	10	4	6
Employee 12	6	5	1	9	10	2	3	4	7	8
Employee 13	6	8	10	9	1	2	3	4	5	7
Employee 14	6	3	5	9	1	2	10	4	8	7

Assignments	Lot 1	Lot 2	Lot 3	Lot 4	Lot 5	Lot 6	Lot 7	Lot 8	Lot 9	Lot 10	Total	Preference
Employee 1	0	0	1	0	0	0	0	0	0	0	1	1
Employee 2	0	0	0	0	1	0	0	0	0	0	1	1
Employee 3	0	0	0	0	0	0	0	0	1	0	1	2
Employee 4	0	0	0	0	0	0	0	0	1	0	1	1
Employee 5	1	0	0	0	0	0	0	0	0	0	1	1
Employee 6	0	0	0	0	0	0	0	0	0	1	1	3
Employee 7	0	1	0	0	0	0	0	0	0	0	1	1
Employee 8	0	0	0	1	0	0	0	0	0	0	1	3
Employee 9	0	0	0	0	0	0	0	1	0	0	1	1
Employee 10	0	0	0	0	0	0	1	0	0	0	1	1
Employee 11	0	0	0	0	0	0	1	0	0	0	1	1
Employee 12	0	0	0	0	0	0	0	1	0	0	1	4
Employee 13	0	0	0	0	1	0	0	0	0	0	1	1
Employee 14	0	0	0	0	0	1	0	0	0	0	1	2
Total	1	1	1	1	2	1	2	2	2	1		23
Required	1	1	1	1	2	1	2	2	2	1		

Fig. 5: The program after running.