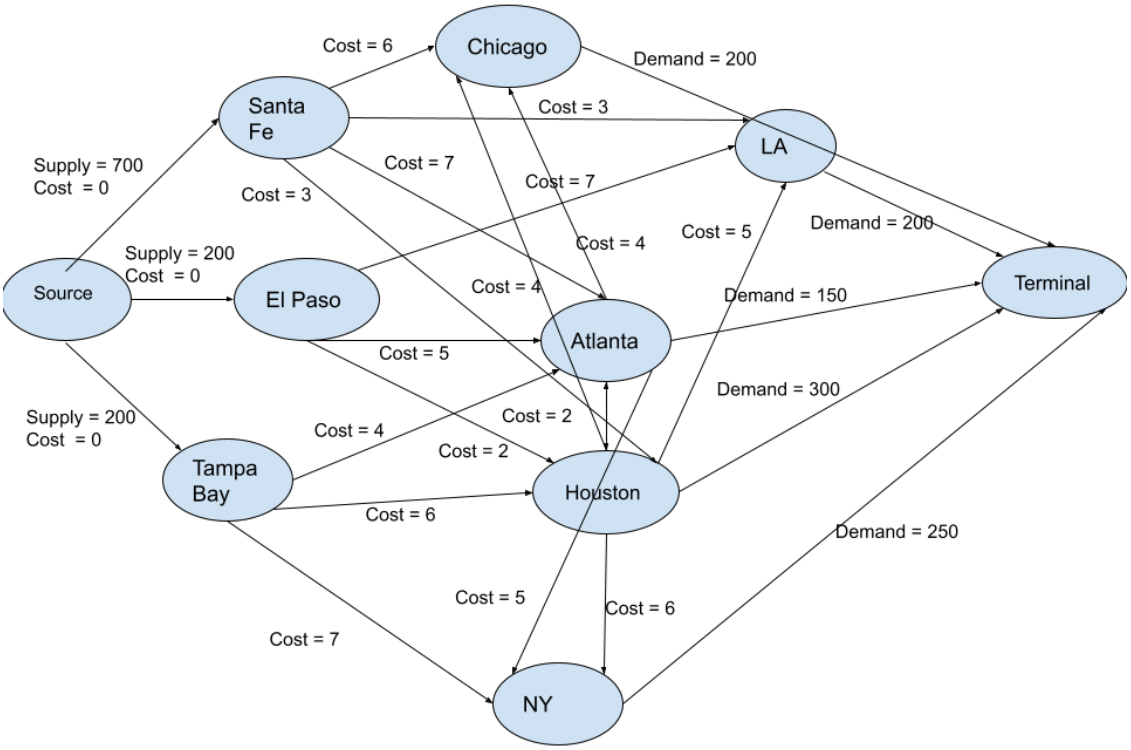
Midterm 1

Scott Fullenbaum
MATH 87 Math Modeling

February 22, 2022

1 Network Model

Here is my network flow model, containing all possible nodes. The initial node and the terminal node don't matter in terms of calculations, but they represent the total supply and demand. Each edge can have a max flow of 200 ducks traveling through it, as laid out in the problem. The edges directly connected to the terminal technically have no max flow, though they serve no direct purpose when it comes to making calculations:



2 Linear Program

The goal here is to minimize shipping costs while meeting demand. For all future parts, p_1 will refer to Chicago, p_2 will refer to LA, p_3 will refer to Atlanta, p_4 is for Houston, and p_5 is for NY.

To determine the objective function, we can look at how much it costs to send a duck from any destination to another destination.
For Chicago, it costs 6 to send from Santa Fe, and 4 to send from Houston and Atlanta. We can then sum these up and get $6 * p_1 + 4 * p_1 + 4 * p_2 = 14p_1$. Repeating this for each other destination, the objective function we arrive at is $14p_1 + 15p_2 + 18p_3 + 13p_4 + 18p_5$.

There are two classes of constraints, those relating to supply and those relating to demand. For the supply constraints, the amount each factory can send to any number of cities is limited. Sante Fe can send up ducks to Chicago, LA, Houston, and Atlanta, and has only 700 it can send, so the constraint here is $p_1 + p_2 + p_3 + p_4 \leq 700$. Repeating this process for the other two supply centers, we get the other two supply constraints to be $p_2 + p_3 + p_4 \leq 200$ for El Paso and $p_3 + p_4 + p_5 \leq 200$ for Tampa Bay.
There are two other supply constraints to consider, Atlanta and Houston can also serve as suppliers, sending ducks to others. Any excess ducks should go to other cities, so for Houston, this additional constraint becomes $p_1 + p_2 + p_4 + p_5 \geq 300$ and for Atlanta, it is $p_1 + p_2 + p_3 \geq 150$

For demand constraints, the goal is to meet demand, meaning we want the number of ducks sent to each city to be at least the demand. These constraints are $p_1 \geq 200$, $p_2 \geq 200$, $p_3 \geq 150$, $p_4 \geq 300$, and $p_5 \geq 250$. We multiply these and any constraints accompanied with $\geq$ by negative 1 so that all constraints share the same sign.

There is one last class of constraints. Since flow is restricted to maximum of 200 ducks on each path, there can never be more than 200 times the number of paths to a single city. For example, there are three paths to Chicago, meaning we can never send more than 600 ducks to Chicago. This creates the last set of constraints to be $p_1 \leq 600$, $p_2 \leq 600$, $p_3 \leq 800$, $p_4 \leq 800$, $p_5 \leq 600$.
As a side note, I did find that including the last set of constraints only created a marginal difference in the resulting solutions.
Here is a complete list of the constraints, as used in the code

$$\min f(p_1, p_2, p_3, p_4, p_5) = 14p_1 + 15p_2 + 18p_3 + 13p_4 + 18p_5$$
$$\text{constraint 1: } p_1 + p_2 + p_3 + p_4 \leq 700$$
$$\text{constraint 2: } p_2 + p_3 + p_4 \leq 200$$
$$\text{constraint 3: } p_3 + p_4 + p_5 \leq 200$$
$$\text{constraint 4: } -p_1 - p_2 - p_4 - p_5 \leq -300$$
$$\text{constraint 5: } -p_1 - p_2 - p_3 \leq -150$$
$$\text{constraint 6: } -p_1 \leq -200$$
$$\text{constraint 7: } -p_2 \leq -200$$
$$\text{constraint 8: } -p_3 \leq -150$$
$$\text{constraint 9: } -p_4 \leq -300$$
$$\text{constraint 10: } -p_5 \leq -250$$
$$\text{constraint 11: } p_1 \leq 600$$
$$\text{constraint 12: } p_2 \leq 600$$
$$\text{constraint 13: } p_3 \leq 800$$
$$\text{constraint 14: } p_4 \leq 800$$
$$\text{constraint 15: } p_5 \leq 600$$

3 Code

Here is how I coded the constraints in Python. I used this general setup for solving the linear program in any of the cases, including under striking circumstances, I just adjusted constraints

accordingly.

```
import numpy as np
from scipy.optimize import linprog

c = np.array([14,15,18,13,18])
A = np.array([[1,1,1,1,0], #constraint 1
              [0,1,1,1,0], #constraint 2
              [0,0,1,1,1], #constraint 3
              [-1,-1,0,-1,-1], #constraint 4
              [-1,-1,-1,0,0], #constraint 5
              [-1,0,0,0,0], #constraint 6
              [0,-1,0,0,0], #constraint 7
              [0,0,-1,0,0], #constraint 8
              [0,0,0,-1,0], #constraint 9
              [0,0,0,0,-1], #constraint 10
              [1,0,0,0,0], #constraint 11
              [0,1,0,0,0], #constraint 12
              [0,0,1,0,0], #constraint 13
              [0,0,0,1,0], #constraint 14
              [0,0,0,0,1]]) #constraint 15
b = np.array([700,200,200,-300,-150,-200,-200,-150,-300,-250,600,600,800,800,600])

res=linprog(c,A_ub=A,b_ub=b)

print(res)
```

Since partial ducks are allowed, I'm rounding all duck numbers to three decimal places. I'm also rounding all monetary values to two decimal places, as the cost is total, and the company can't pay thousandths of a cent.

The above code yields the results of $p_1 = 202.477, p_2 = 200.978, p_3 = 150.252, p_4 = 300.653, p_5 = 250.036, and plugging these numbers into the objective function gives a minimum cost of \$16963.01$.

The one issue with the result above is that the solution to the linear program gives a total number of ducks of 1104, but the total supply is 1100. I attempted to fix this by introducing an additional constraint, $p_1 + p_2 + p_3 + p_4 + p_5 \leq 1100$, meaning the total number of ducks sent to all places combined cannot exceed 1100, but the sum of my values was 1103, which directly violates this constraint. At the moment, the only reason I can think this is violated is that my algorithm might not properly account for number of ducks sent from Houston and Atlanta, but that still doesn't properly explain how the total supply can be exceeded when I introduce a constraint stating that.

4 LA and Alternative Situation

4.1 Case 1: All shipping costs to LA double.

In terms of our system, the coefficient for LA, p_2, will double, as that coefficient is the total cost of shipping. Updating the code and coefficients, the code is updated to this and while the optimal

point doesn't change, as only the objective function is changing, you get a higher cost of $1996.76.

4.2 Case 2: Max flow to LA cut in half.

If supply is cut in half, then the max demand through any path that leads to LA has a max flow of 100, rather than 200. This means we have to half the max flow for LA, while leaving all other constraints the same, as none of them are related to the max flow going into LA. For my constraints, this falls under constraint 11. Halving this and calculating gives a new total cost of 16953.19, which is very close to the initial result. This makes sense as demand for LA is 200, but max flow to LA is 300, meaning under reduced capacity, the city can still reach demand.

In conclusion, doubling shipping costs will lead to a much higher price increase.

5 Houston and Strikes

For both cases, we follow the same procedures as done in the previous section.

5.1 Case 1: All shipping costs to Houston double.

If all shipping costs to Houston double, we follow the same procedure as we did in the prior part and in this case, doubling the coefficient, as only ducks coming into the city will have double shipping prices, meaning that coefficient becomes 26 rather than 13 and the new objective function is $f(p_1, p_2, p_3, p_4, p_5) = 14p_1 + 15p_2 + 18p_3 + 26p_4 + 18p_5$
Since only the objective function changes, the solution via number of ducks is the same as it was in part 3, but the new minimum shipping cost is now $20856.32.

5.2 Case 2: Max flow to Houston is cut in half.

If max flow to Houston is cut in half, then the flow constraint from Houston is halved, meaning that constraint is now $p_4 \leq 400$. Using the code from section 3, we get a new minimum shipping cost of 16962.86, which is very close to the original value. The reason this makes sense is that Houston can still receive ducks from all three suppliers along with Houston, meaning that the city is still capable of getting enough ducks through other routes.

6 Profit

The new goal is to maximize a profit objective function. Since nothing about demand or distribution has changed, I think it is fair to assume that all constraints derived in part 2 are still applicable, meaning we need to get a new objective function. For sending a duck from any destination to another, the total profit is the list profit minus production cost, minus shipping along that route. For example, if I send a duck from Santa Fe to Chicago, the profit from that is $1, as it costs $8 to make, $6 to ship, then sells for $15, giving a net profit of 1. Following this process for each other route going from Santa Fe, gives us the total profit being $p_1 + 9p_2 - p_3 + 5p_4$. We can repeat this process for the two other production centers, getting $8p_2 + 3p_4$ for El Paso. For Tampa, we get $-4p_3 - 6p_4 + 8p_5$. For my objective function, I added all three of them together getting a new objective function of $p_1 + 17p_2 - 5p_3 + 2p_4 + 8p_5$.
There are also the two relay nodes, which can serve as traveling points, but since each duck sold

must travel come from any of the main warehouses, I decided to keep my equation as that. An alternative approach would be knowing the profit of each warehouse, maximizing each one individually, then accounting for flow restrictions. Lastly, while the flow constraints must remain, we can change our demand constraints. I think it's fair to assume that any amount of ducks that exceeds demand will not be sold. Therefore, rather the total ducks be $\geq$ demand, we want total ducks $\leq$ demand. This creates the constraints $p_1 \leq 200$, $p_2 \leq 200$, $p_3 \leq 150$, $p_4 \leq 300$, $p_5 \leq 250$.

Here is my updated code for this part:

```
import numpy as np
from scipy.optimize import linprog

c = np.array([1,17,-5,2,8])
A = np.array([[1,1,1,1,0], #constraint 1
              [0,1,1,1,0], #constraint 2
              [0,0,1,1,1], #constraint 3
              [-1,-1,0,-1,-1], #constraint 4
              [-1,-1,-1,0,0], #constraint 5
              [1,0,0,0,0], #constraint 6
              [0,1,0,0,0], #constraint 7
              [0,0,1,0,0], #constraint 8
              [0,0,0,1,0], #constraint 9
              [0,0,0,0,1], #constraint 10
              [1,0,0,0,0], #constraint 11
              [0,1,0,0,0], #constraint 12
              [0,0,1,0,0], #constraint 13
              [0,0,0,1,0], #constraint 14
              [0,0,0,0,1]])  #constraint 15

b = np.array([700,200,200,-300,-150,200,200,150,300,250,600,600,800,800,600])

res=linprog(c,A_ub=A,b_ub=b)

print(res)
```

The optimal solution under this system is at $p_1 = 200$, $p_2 = 0$, $p_3 = 100$, $p_4 = 100$, $p_5 = 0$, and gives a maximum profit of 100. Looking at the data provided, it would best to prioritize on maximizing demand in LA and NY, which, become zero under my constraints. I think these areas are underserved as ducks sell for the maximum price there. Furthermore, when solved, the constraint that $p_3 \leq 150$ was violated, which is some product of the system that is unaccounted for in my results. To maximize future profit, the best strategy would probably be to reallocate a fair amount of production from Santa Fe to El Paso, as you could use the relay station in Houston to send it to NY, netting of \$12 which much higher than any other potential current route.
The two transportation hubs at Houston and Atlanta are also overserved, as they provided the lowest profit, meaning it's harder to break even. As a result, it might be best for the supplier to ignore selling to those cities in general, utilizing them to send ducks to more profitable cities, primarily NY and LA.
Comparing to part 3, profit maximization is best achieved by not selling all ducks, as some demand restrictions can lead to a decrease in profit.