DS9 Report

Approach (1)

The goal of this linear programming model is to distribute the maximum units of energy throughout the DS9 network. In order to create the model, two variables were created. The constraint xa represents the units of energy flowing forward between two arcs. In DS9, energy can flow forward and backwards between nodes. To simplify this, each arc between two nodes is only represented once (notated as moving from the node with smaller numerical name). For example, the arc between node 1 and node 2 is included but the arc from node 2 to node 1 is not; the latter arc would simply be the negative of the former. The variable, yn, is the total energy units received by node n. This variable is maximized in the objective function to determine the optimal solution where all the constraints are met.

To maximize the total demand by all nodes, we begin by meeting two initial constraints. The first main constraint is setting the upper bound for all nodes. It is assumed that the each node can only accept enough energy to satisfy the amount demanded. Secondly, the upper and lower limits of each arc is set. Interestingly, the arcs between nodes have no constraint dictating which direction they must flow - to model this, we set lower and upper bounds of equal magnitude for each node. The upper bound represents how much demand it moving forward and the negative lower bound represents the maximum units of energy moving backwards on the arc.

The last constraint set ensured that for any given node, the demand satisfied, yn, must be the sum of incoming arcs and negative sum of outgoing arcs. Since it is assumed that the generator can produce an unlimited amount of power, the arc, x0,1, connects a fake source to the first node to somewhat trivially start supplying energy to the network.

Implementation Results (1)

The maximum amount of energy distributed throughout the network is 99 units. It was distributed among the nodes as listed in the table on the following page.

A few important observations from the results should be discussed. First, the first node receives the total amount of energy demanded since it was trivally connected to the energy source. Secondly, 99 units of energy flowed through the arc, x0,1, which indicates that the model is accurate. Finally, due to the objective function, the model does not distribute energy fairly throughout the nodes. Instead, some nodes receive their total demand requested while many nodes receive none. This issue will be discussed in the next portion of the model.

Linear Program Solution (1)

| Node | Demand Received (yn) |
| --- | --- |
| 1 | 6 |
| 2 | 8 |
| 3 | 2 |
| 4 | 4 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |
| 8 | 4 |
| 9 | 3 |
| 10 | 6 |
| 11 | 0 |
| 12 | 4 |
| 13 | 0 |
| 14 | 6 |
| 15 | 0 |
| 16 | 6 |
| 17 | 6 |
| 18 | 3 |
| 19 | 0 |
| 20 | 0 |
| 21 | 8 |
| 22 | 2 |
| 23 | 0 |
| 24 | 6 |
| 25 | 1 |
| 26 | 8 |
| 27 | 8 |
| 28 | 0 |
| 29 | 7 |
| 30 | 1 |

Mathematical Model (1)

*Variables*

*yn = node n’s fulfilled demand,* ∀*n = 1,…,N*

*xa = energy travelled forward on arc from node n1 to n2* ∀*a* ∈ *A*

*Parameters*

*A = set of arcs*

*N = number of nodes in network*

*un = upper bound for arc n,* ∀*n = 1,…,N*

*Dn = max demand for node n,* ∀*n = 1,…,N*

Fairness Metric (2a)

For our fairness metric we decided to implement a constraint that would require each group to have at least a specific percentage of power demanded. We implemented this by creating a value ‘a’ that we could manipulate based on a power percentage that was achievable given the arc capacities and demand requirements. Before we could test our metric though, we had to create the groups by appending a dictionary that would assign each node with its group. From here we can calculate each group’s maximum capacity and its demand. Using the demand values from each group, we could set the constraint by formulating that ‘a’ must be less than or equal to each group’s percentage of demand met. This fairness method was successful in that our total demand satisfied didn’t change from our fairness method application.

Shortcomings (2b)

Our metric stresses that groups get at least a certain percentage of power demanded, but there is no upper constraint except for maximum capacity. This means that with an ‘a’ value of 60%, every group will get at least 60%, but one could get 60% and another could get 90%. Another concern is smaller groups that demand less power. While these groups will get the same percentage of power, 40% less power in a small group could be much more detrimental than that of a larger group. One final concern, is that our metric doesn’t concern the individual nodes of each group. There are no implications of this being a problem in this scenario, but if each of those nodes are cities and two nodes have 100% and the other has 20%, while the group still has 60%, one city is scavenging for power.

Approach (2c)

The original LP model was changed to formulate a model that maximizes the fairness metric described earlier. The variable, a, represents the percentage of demand that each group must receive. The goal of this model is to maximize fairness, so the objective is maximize “a”. This maximizes the percentage of demand that is satisfied for each group. In order to complete this model, two constraints are added. First, the fairness metric must within the range 0 and 1. Secondly, for each group, the percentage of the demand satisfied must be greater than or equal to “a”.

Mathematical Model (2c)

*Variables*

*yn = node n’s fulfilled demand,* ∀*n = 1,…,N*

*xa = energy travelled forward on arc from node n1 to n2* ∀*a* ∈ *A*

*a = minimum percentage of demand that each group on DS9 must receive*

*Parameters*

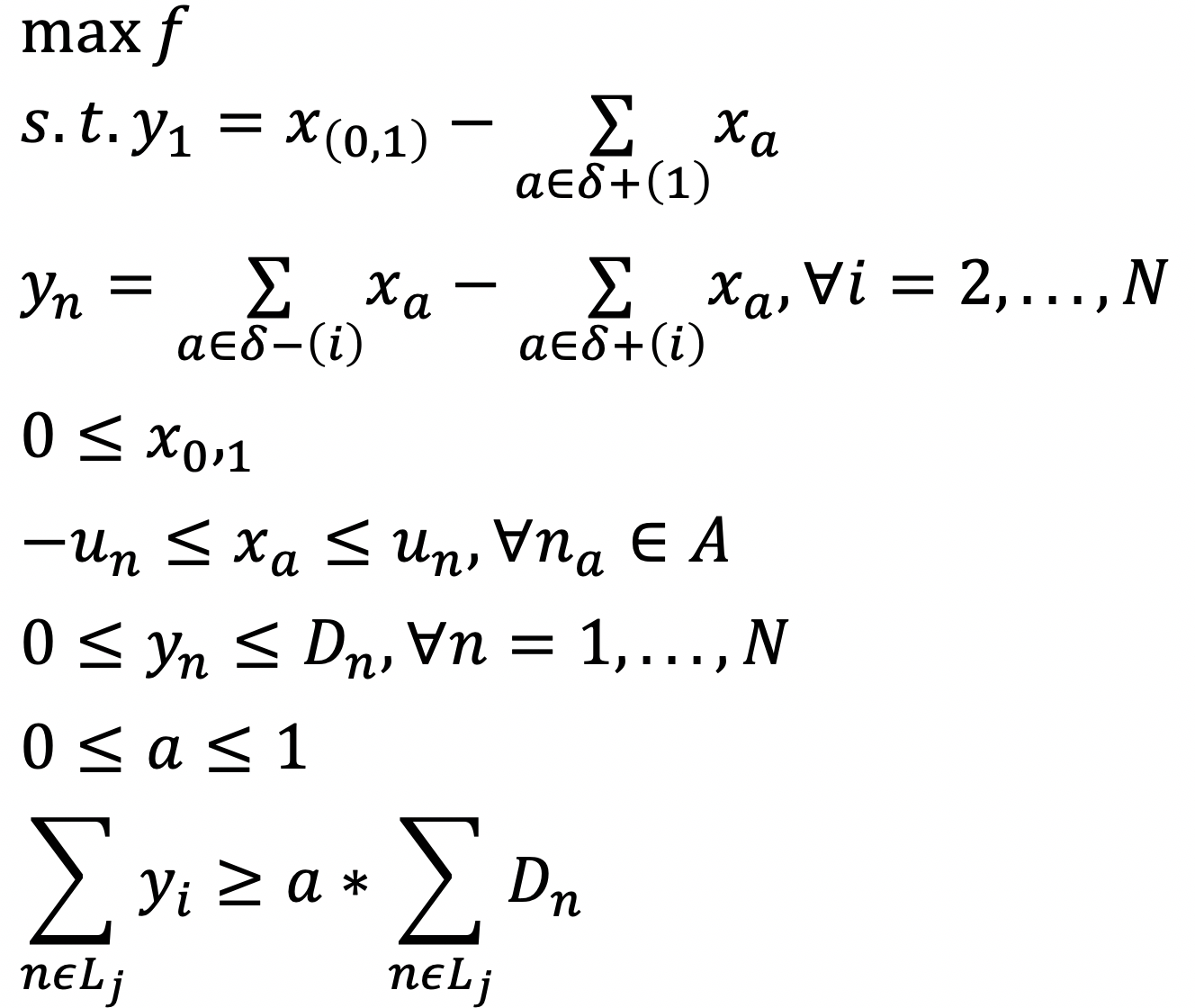
*A = set of arcs*

*N = number of nodes in network*

*un = upper bound for arc n,* ∀*n = 1,…,N*

*Dn = max demand for node n,* ∀*n = 1,…,N*

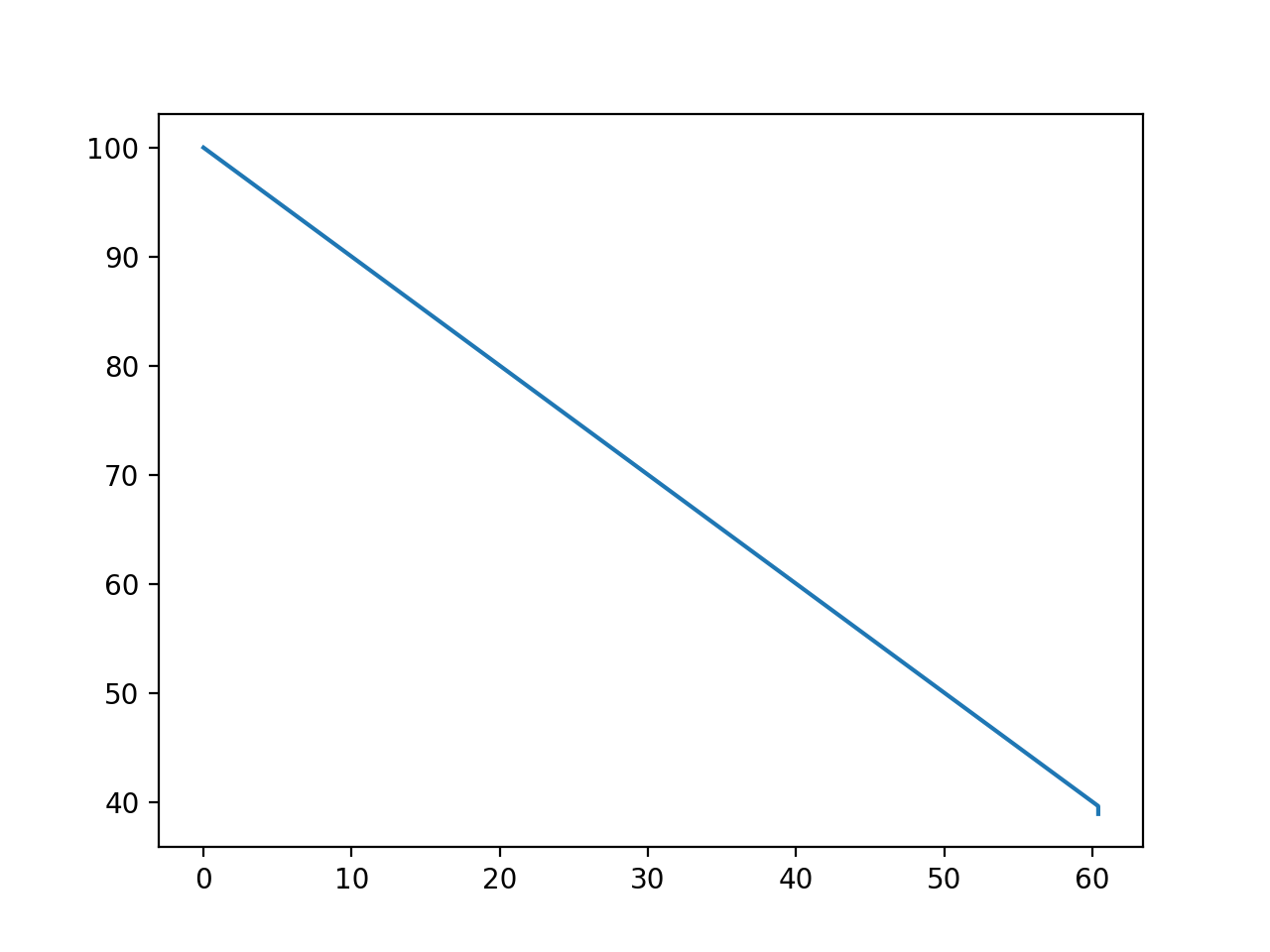
*Lj = set of all nodes in group j ,* ∀ *j = 1, 2, 3, 4, 5, 6, 7*

**

Implementation Results (3a)

The result of the optimization model yielded a fairness metric of 0.603. This indicates that enough power was distributed to each group to satisfy 60.3% of their demand or more. The total demand satisfied was constant at 99 units of energy. This fairness metric distributed the power more evenly among the groups, unlike the solution to the first problem.

Trade-off Curve (3b)

Our trade-off curve displays the indirect relationship between our fairness metric ‘a’ on the x-axis and the total demand met as a decimal on the y-axis. The maximum fairness value is .6 and the slope is around 1 so each percentage point of fairness increased results in a 1 percentage point decrease in demand met.

**Satisfied Demand** (% of total)

**Effect of Percent Fairness on Satisfied Group Demand**

**Fairness** (100 x demand satisfied/total group demand)

Glossary

| Mathematical Model (1) | DS9 Python Code |
| --- | --- |
|  |  |
| Variables | |
| *yn* | *yn* |
| *xa* | *xn1xn2* |
|  |  |
| Parameters | |
| *A* | *arcs* |
| *N* | *nodes* |
| *un* | *d[int(i.split(“x”)[1])][int(i.split("x")[2])]), ∀i in A* |
| *Dn* | *demand[int(i[1])], ∀i in N* |

| Mathematical Model (2c) | DS9 Python Code |
| --- | --- |
|  |  |
| Variables | |
| *yn* | *yn* |
| *xa* | *xn1xn2* |
| *a* | *percent* |

|  |  |
| --- | --- |
|  | |
| Parameters | |
| *A* | *arcs* |
| *N* | *nodes* |
| *un* | *d[int(i.split(“x”)[1])][int(i.split("x")[2])]), ∀i in A* |
| *Dn* | *demand[int(i[1])], ∀i in N* |
| *Lj* | *groups[j], ∀j=1,2,3,4,5,6,7* |