Nick DeGroote
Optimal Control: Homework Assignment No. 3
8 June 2021

Arthrodax Company has been approached by Ranger Sound with a rush order offer to purchase 100 units of a customized version of Arthrodax's SoundScreamer audio mixer at $5,000 per unit, and Arthrodax needs to decide how to respond. The electronic modifications of the standard SoundScreamer needed for this customized version are straightforward, but there will be a fixed cost of $100,000 to design the modifications and set up for assembly of the customized Sound- Screamers, regardless of the number of units produced. It will cost $2,000 per unit to manufacture the circuit boards for the units. Since Arthrodax has some short term spare manufacturing capacity, the Ranger offer is potentially attractive. However, the circuit boards for the customized units will not fit into the standard SoundScreamer case, and Arthrodax must decide what to do about acquiring cases for the customized units as it decides whether to accept Ranger's purchase offer. An appropriate case can be purchased at $500 per case, but Arthrodax could instead purchase an injection molder to make the cases. It will cost $20,000 to purchase the molder, and there is a 0.6 probability that it will be possible to successfully make the cases using the molder. If the molder does not work, then the purchase price for the molder will be totally lost and Arthrodax must still purchase the cases at $500 per case. If the molder works, then it will cost $60 per case to make the cases using the molder. Regardless of which case is used, the cost of assembling the SoundScreamer circuit boards into the case is $20 per unit. Unfortunately, there is no way to test the molder without purchasing it. Assume that there is no other use for the molder except to make the cases for the Ranger order.

(i) Draw a decision tree for Arthrodax's decision about whether to accept the Ranger offer and how to acquire the cases for the customized SoundScreamers.

We will first go through a short description of logic included in the decision tree. The first decision is whether or not to accept the offer. If the offer is accepted, then there will be a flat $100,000 development cost for all scenarios. There will also be another $200,000 cost incurred after summing the total unit cost for all 100 units at $2,000 a piece. The next decision is whether or not to purchase the injection molder. If the molder is purchased, then there will be a $20,000 equipment cost. This then entails a 40% chance of failure and a 60% chance of success. Different assembly costs are incurred based on the success of the molder, $6,000 if successful and $50,000 if not. Regardless of if the molder is successful, there is a final $2,000 assembly cost over all units. If the injection molder was not purchased, then the equipment cost is $0 and the cost for all cases is $50,000. The illustration of the tree for all outcomes is shown in Figure 1.

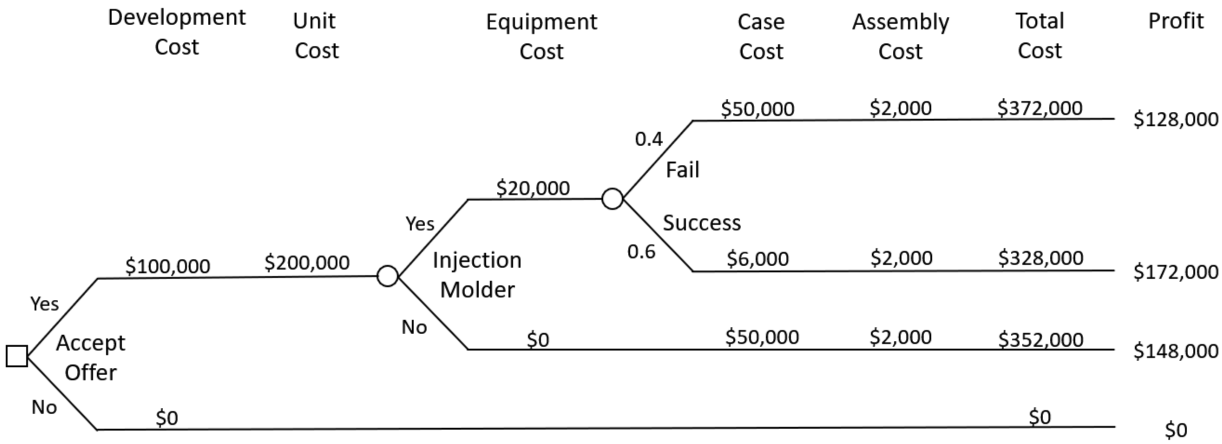


Figure 1: The decision tree illustration for Problem 1, part (i).

(ii) Using expected net profit as the decision criterion, determine the preferred course of action for Arthrodax.

The course of action for Anthrodax can be easily determined using expected value. Expected value is determined by the weighted profit value based on the likelihood of an outcome. In this scenario, we have only one decision which has a probabilistic outcome. This is for the chance that the injection molder is successful at producing cases for the units. The expected values for each possible outcome is shown below.

1. Accept Offer, Buy Injection Molder

$$EV = 128000 * 0.4 + 172000 * 0.6 = 154400 \tag{1}$$

2. Accept Offer, Do not buy Injectio Molder

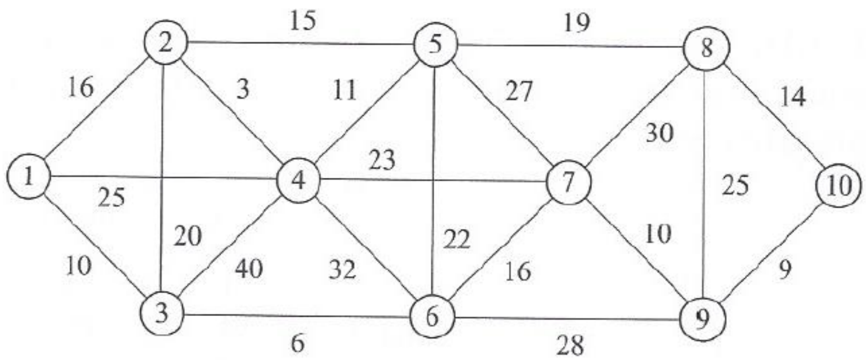
$$EV = 148000 * 1.0 = 148000 \tag{2}$$

3. Do not Accept Offer

$$EV = 0 \tag{3}$$

From these simple equations, we can see that Anthrodax should accept the offer from Ranger Sound, then buy the injection molder. On average, buying the injection molder should produce more profit than choosing not to do so.

2. For the network shown below, develop a MATLAB algorithm, based on Dijkstra's Algorithm, to find the shortest path from node 1 to node 10. Draw a block diagram explaining your algorithmic approach and compare your result with that obtained using a nearest neighbor heuristic approach (also to be realized using MATLAB). What are your observations and reflect upon the lessons learned from a computational as well as from a performance perspective. Attach your MATLAB m file(s) with comments to explain your code embedded.



A Python script was written to find the shortest path between two nodes in a graph using Dijkstra's Algorithm and is attached with this assignment. A block diagram of the implementation is shown on the next page in Figure 2. Using Dijkstra's Algorithm, we find the following optimal path.

$$1 \rightarrow 3 \rightarrow 6 \rightarrow 7 \rightarrow 9 \rightarrow 10 : 51 \quad (4)$$

This means the shortest route visits nodes 1, 3, 6, 7, 9, and 10 with a total distance of 51.

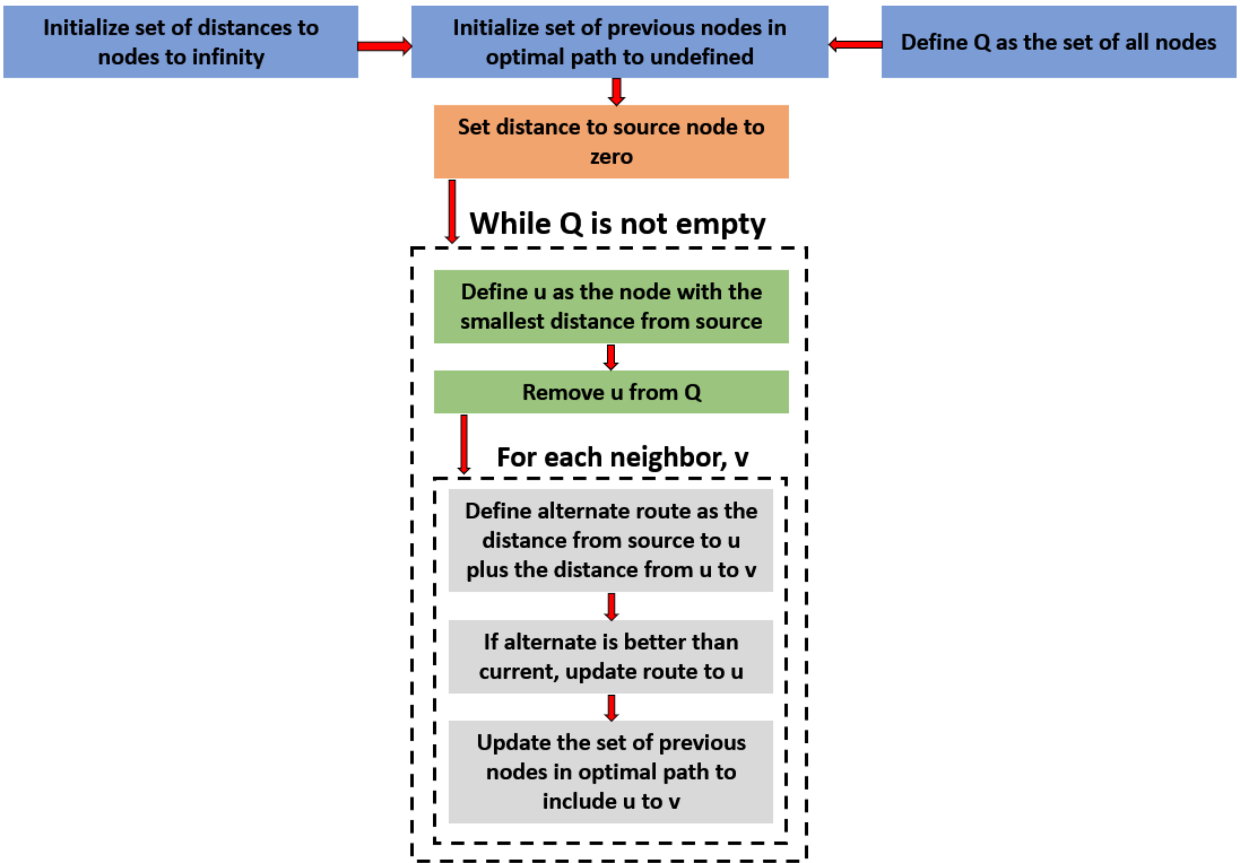


Figure 2: A block diagram of the implementation of Dijkstra's Algorithm used in this assignment

A nearest neighbor approach was also implemented for the purposes of finding a route from node 1 to node 10. Nearest neighbor simply moves from the current node to the next node with the lowest cost. The nearest neighbor algorithm was also implemented in Python and is attached to this assignment. A human can easily trace through this algorithm, finding the following route with nearest neighbor.

$$1 \rightarrow 3 \rightarrow 6 \rightarrow 7 \rightarrow 9 \rightarrow 10 : 51 \tag{5}$$

Somewhat surprisingly, nearest neighbor finds the same solution as Dijkstra's algorithm. Nearest neighbor is a greedy algorithm and can become caught in local minima quite easily. Dijkstra's algorithm on the other hand guarantees an optimal path between two nodes in a graph.

Both algorithms have a good time complexity, where we are defining good here to mean that the problem can be solved in polynomial time or better. They provide a computational method to find a path from one node to another without the need to brute force the solution. We can see that the time complexity for Dijkstra's algorithm is:

$$\mathcal{O}(|E| + |V| \log(|V|)) \tag{6}$$

And the time complexity of nearest neighbor for a defined graph is:

$$\mathcal{O}(|E||V|) \tag{7}$$

The nearest neighbor method runs somewhat faster than Dijkstra's algorithm. It is worth reiterating however, that nearest neighbor does no guarantee an optimal solution. In this situation, it is only a coincidence that the nearest neighbor method also finds the best solution. There are many other situations when nearest neighbor does not find the optimal solution. For example, in a travelling salesman problem it is rare for nearest neighbor to perform the best when compared to several other methods. In conclusion, nearest neighbor may sometimes find a good solution depending on the problem at hand, but the user should be cautions if using it. If optimality is a concern at all then Dijkstra's algorithm should be used instead.