**Part 1: Smart Manufacturing**

### 1.1 Planning Using A\* Search

For the five different types of widgets:

* Widget 1 = Components AEDCA
* Widget 2 = Components BEACD
* Widget 3 = Components BABCE
* Widget 4 = Components DADBD
* Widget 5 = Components BECBD

The factory sequence with the smallest number of stops is: (Three results using different starting points)

1. ['D', 'B', 'A', 'E', 'D', 'C', 'A', 'B', 'C', 'D', 'E']
2. ['B', 'A', 'E', 'D', 'C', 'A', 'D', 'B', 'C', 'D', 'E']
3. ['A', 'B', 'E', 'D', 'C', 'A', 'D', 'B', 'C', 'D', 'E']

But the sequence starting with D have the least expanding which is 121

The factory sequence with the smallest number of miles traveled is:

['D', 'E', 'B', 'E', 'C', 'A', 'E', 'D', 'E', 'B', 'E', 'C', 'A', 'E', 'D']

The distance traveled is 5594

State representation:

In every state I stored

name

The name of current location (factory) of the truck

Exp. 'A', 'B', 'C', 'D' **or** 'E'

remainelements

All the elements still need installed (not installed yet) in every Widget

Exp. [[ 'E', 'D', 'C', 'A']

['E', 'A', 'C', 'D']

['A', 'B', 'C', 'E'] After going through 'D', 'A', 'B'

['D', 'B', 'D']

['E', 'C', 'B', 'D']]

remaintype

All the type of elements still need installed (not installed yet) in all Widgets

Exp. ['B', 'C', 'D', 'E' ] After going through 'D', 'B', 'A', 'E', 'D', 'C', 'A'

remainNum

The number of each type of elements need installed (not installed yet) in all Widgets

Exp. [3, 3, 4, 4, 4] After going through 'D', 'A', 'B'

path

The Path the truck has gone through to be current state.

Exp. ['D', 'B', 'A', 'E', 'D', 'C', 'A'] After going through factory 'D', 'B', 'A', 'E', 'D', 'C', 'A'

G

G: **Step**: The number of steps in the path the truck has already go through

**Miles**: The number of miles in the path the truck has already go through

H

H: **Step**: The number of types (kinds) of elements left (need installed and not installed yet)

**Miles**: The minimum summary of a set of minimum (weight) spanning trees. These trees are ones using from elements need installed to all the elements

F

F: G + H

**Actions:**

* **Going from one factory to another factory**
* **Change the name of the state**
* **Add the G by distance (if Step, add 1; if mile, add the mile distance from the last location)**
* **Predict the heuristic value**
* **Remove the first element in every widget if it’s the same as the element of current factory**
* **Update the number of each type of every element**
* **Update the kinds of elements left**
* **Add the current factory to the path**
* **Go to the next state in the priority queue**

**Heuristic:**

* Steps:

The number of types (kinds) of elements left (need installed and not installed yet)

* Miles:

First Attempt:

1. We get the types (kinds) of elements left and generate a minimum (weight) spanning trees.
2. Get the summary of edges’ length in minimum (weight) spanning trees and set it as a least value.
3. Get all possible combination of types (kinds) of elements which have finished installation.
4. For each time, add the possible combination to the elements which have not finished installation and generate a minimum (weight) spanning trees.
5. Get the minimum value of summary of edges’ length in all the minimum (weight) spanning trees. This is the heuristic value.

Second Attempt:

1. We get the types (kinds) of elements left and generate a minimum (weight) spanning trees.
2. Get the summary of edges’ length in minimum (weight) spanning trees and set it as a least value.

**Implement:**

In the main function we get all the elements which are the first one of all the widgets and make them the possible start factory. Then we run a function named **step** considering all possible start factories. The function took the argument of the Widgets list and a start factory.

In the Step function, we generate a state object which represent the truck’s starting at the “start factory”, and calculate all the attribute in the state object including the name of current location (factory) of the truck, all the elements still need installed (not installed yet) in every Widget, all the types of elements which has not finished all the installation in all Widgets, number of each type of elements need installed (not installed yet) in all Widgets, the path, G, H, and the F. Then we add it to the priority queue which is sorted by the F value. Then we just implement the A\* search to let it go through all the possible “Next State”. When the current state don’t have a next state, this is the final state. We can get the final result including the path it goes and the distance traveled.

### 1.3 Extra Credit

1.

nodes expanded

Step:

3: 14

4: 37

5: 258

6: 1859

7: 2510

8: 82351

Miles:

3:118

4:882

5:18397

6:

7:

8:

2.

Step:

3: 542

4: 1707

5: 3316

6: 106931

7:

8:

Miles:

3: 32961

4: 25040

5: 422281

6:

7:

8:

The number of nodes expanded in uniform cost search is much larger than the A\*.