To solve part 1.1 of the MP we used a StateNode class as our primary data structure. The StateNode object contained the path that the agent took up to that point. It was able to do this by having a character representing the agent’s current location, a progress vector showing how far the agent is in the assembly of each of the widgets, and pointer to the parent StateNode.

The algorithm works by creating a StateNode with no current position, and a progress vector filled with zeros. It then expands that node using a “get\_transitions()” method to get a list of child nodes and puts all of them into a priority queue. The priority queue ranks the nodes by the heuristic function, plus the length of the path it has already traveled. It determines valid transitions as all nodes created by transitions that would advance the progress on at least one widget. When those new nodes were created, if the transition caused the advancement of the development on a widget, the element in the progress vector describing that widget was incremented.

It then pops a state node from that heap and repeats the process until a node is found that has all five widgets completed. Or in other words, all five elements in the progress vector were equal to five. Since our heuristic was consistent and admissible, the described algorithm is an A\* search.

For the case where all the factory locations were equidistant, our heuristic function is the maximum of two smaller functions. The first, being the maximum number of remaining stops to complete a given widget. This means that even if four of the widgets were completed and the last widget was un-started, the cost would be five. The second function computes how many unique characters still need to be visited. By taking the maximum of these two, we got a strong and consistent heuristic.

In order to adapt it to the non-unit distance case, we multiplied that cost times the minimum distance between two nodes. This gave us the path “BAEDCADBCDE” as our shortest unit-distance path and “BEDAEDCBCAED,” which had a cost of 7043, as our shortest non-unit distance path. The unit cost version had to expand 1613 nodes while the non-unit cost expanded 6175 nodes to find the solution.