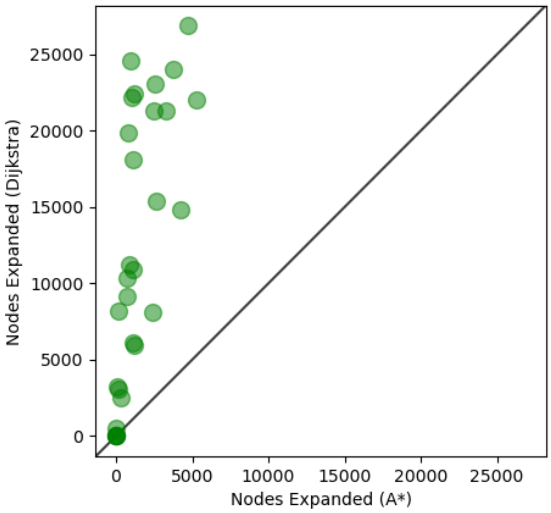
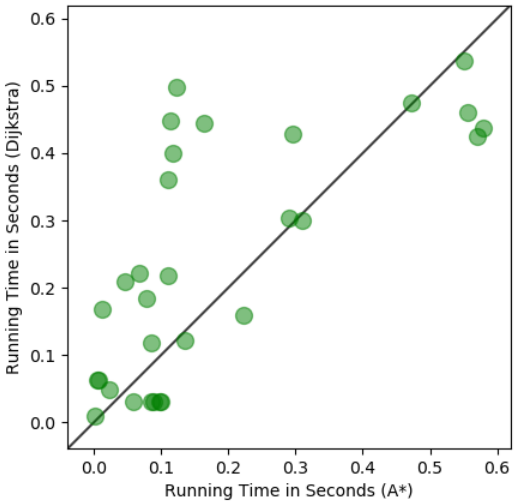
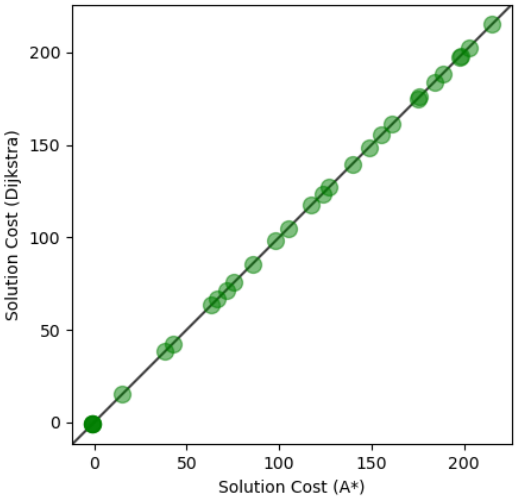
Assignment1



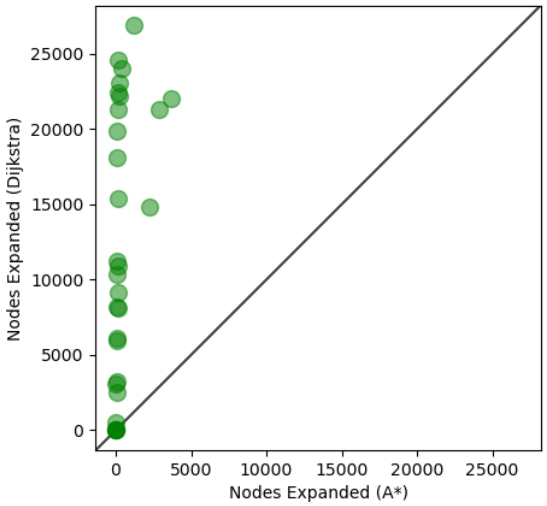


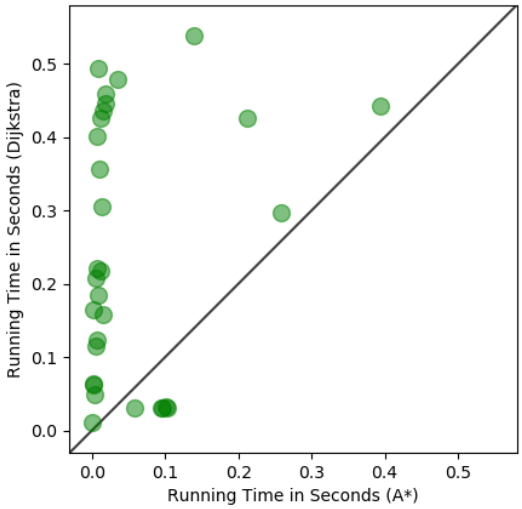


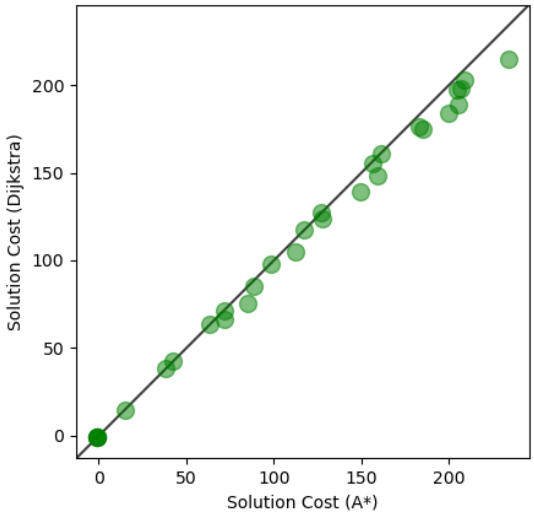
Explain the Results:

From the node expanded plot and running time plot we can find correlation in between. As we can tell in the node expanded plot, the Dijkstra's algorithm had experienced more nodes than the A*. As nodes are more press close to Dijkstra's axis, meaning at this node A* expanded less nodes. Vice versa. From the runtime plot we can get Dijkstra's algorithm is slower than A*, because for more than half of the nodes are above the cutting line. The cutting line meaning the equal points for both algorithm across the plot. On the top triangle, Dijkstra's algorithm takes longer to run. On the bottom triangle, A* takes longer to run. Since more than half of the nodes above the cutting line, it means more than half of the nodes, Dijkstra's algorithm takes longer to calculate. Combine the discovery of two plots we can conclude that more node expanded will take longer to calculate. Which mean the runtime is proportional to the node expanded. These two plots are only similar, because in runtime plot, we can see there are few nodes takes longer time for A* to calculate. But node expanded plot showing all the node, Dijkstra's algorithm where always expanded more nodes. So, the correlation we summarized before should be more the whole node expanded plot and runtime plot. Once we zoom into single node, the proportional rule we summarized isn't always right. Next, we will look at the solution plot. This plot contains the result of what each node's path cost because the result of the shortest path cost is same for both algorithms, all our nodes are on the cutting line. This telling us the shortest path cost for both algorithms are the same.

Multiplicative Factor







These three plots we have above is done by the Dijkstra's algorithm and A* with heuristic function two times bigger (multiply h value by two before return). Because the heuristic function changed for the A*, our three new plots are also slightly different. In the nodes expanded plot we can see the new one's nodes are closer together and almost formed a line. This is because A* expanded less nodes than the previous. So showing in the plot with nodes close to Dijkstra's algorithm. This is because we doubled the h value, so it influenced the comparison between nodes in each map. Some nodes' f value will be bigger than the nodes supposed to be smaller. This cause nodes expanded less. In the new runtime plot, we can tell almost all the nodes are above the cutting line. This means, A* is faster than the previous get to the goal node. Which is also cause by heuristic function. Since we doubled the h value less node expanded, the calculation will be less than the previous so runtime will be faster. Next, with in the solution plot we can tell the shortest path cost were different for both algorithms. Some nodes are off the cutting line and closer to A* axis. This is because after we doubled the h value the heuristic function is no longer giving the proper cost from current node to goal node. So with wrong number, the whole A* algorithm's result will be slightly off the correct solution. So we will see nodes off the cutting line. As a whole, double the h value do make A* faster because less nodes will expand and make the correlation between nodes expanded plot and runtime plot closer to direct ratio but on the other hand A* will never use 2h to speed up the runtime because it cannot give an correct solution.