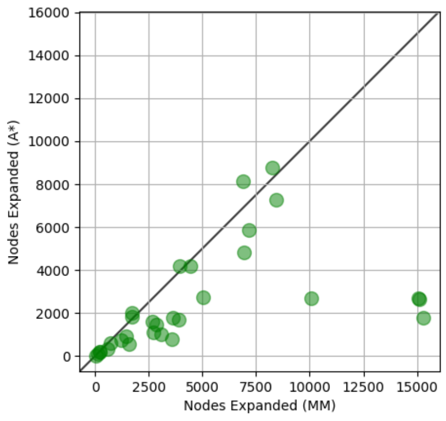

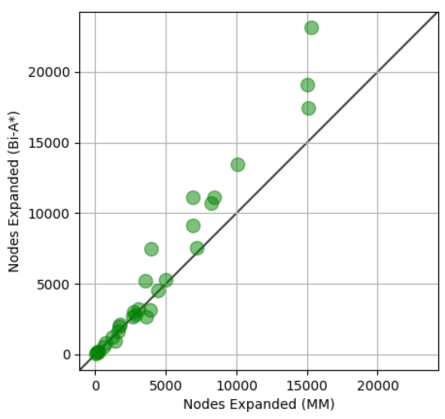

1. (0.5 Mark) In general, does the use of a heuristic function increase or decrease the number of nodes the algorithms need to expand to find a solution?

The use of a heuristic function decreases the number of nodes the algorithms needs to expand.

2. (1.5 Mark) What do you expect to observe in terms of the running time of the algorithms that use a heuristic function in comparison
3. to the algorithms that do not use a heuristic function? Select one of the following options and justify your answer. Hint: You should consider the differences in terms of computational cost of generating a set of children and adding them to the OPEN list for uninformed and informed algorithms.

I agree with D. A* will have a better runtime than Dijkstra's, however it may not be proportional. Compared to Dijkstra, A* has the heuristic function which will 'guide' its search and thus will expand significantly less nodes. Dijkstra's searches in a circle whereas A* searches in almost a straight line. However this runtime is not a proportional increase, due to the fact that A* may have to 'reheapify' more and the runtime can increase a bit. However if we were to use a method where reheapifying is not necessary (using more memory) then a proportional increase in runtime is viable, making B also potentially correct.

3. (0.5 Mark) Does MM tend to perform more or fewer expansions than Bi-A*?

MM has fewer expansions than Bi-A*

4. (0.5 Mark) Did the heuristic-guided bidirectional algorithms deliver their promise of substantially reducing the number of expansions one needs to perform to solve a problem?

Using a heuristic guided-bidirectional algorithm has significantly less expansions than an uninformed search algorithm

5. (1.0 Mark) Speculate on an explanation for the distribution of points in the scatter plot that compares A* and MM. Since you are being asked to speculate, you will receive full marks in this question as long as your answer does not contain conceptual errors.

A* has a better run time in almost every scenario. MM struggles when there is no solution. In the mapped paths, we can see that if there is no solution the starting point searches the inside, while the goal searches the outside. Resulting in more nodes than necessary expanded. Another scenario where A* proves to be better is in the case where we have a wall. A* would just go around one side of the wall whereas MM will "eat" the wall and can go around on both sides. Looking at the graph we can see that in most cases A* expands less nodes than MM. However there are some very specific and niche scenarios where MM will beat out A*