1.

In general, the searches that utilise Heuristic functions performed less searches.

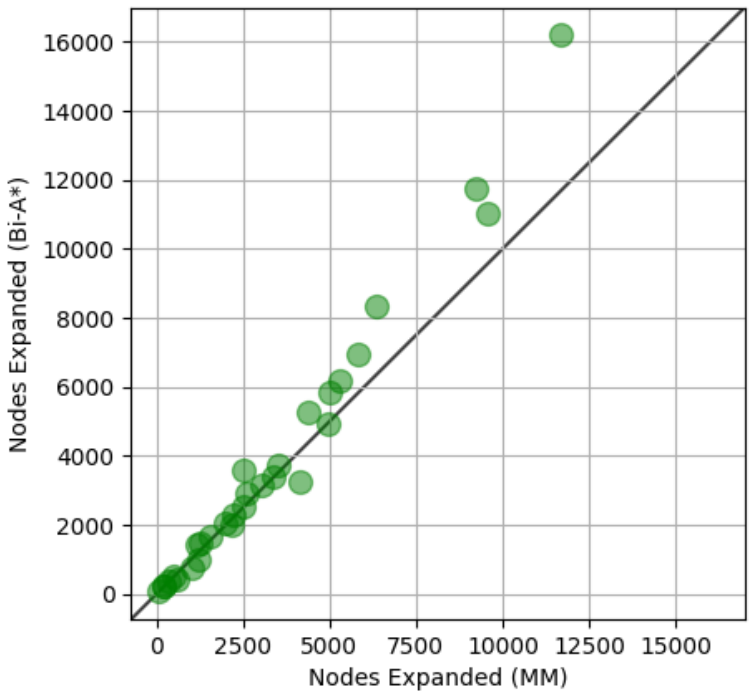
2.

d. The running time will decrease, but not as much as the number of expansions decreases. For example, if A* expands 27% fewer states than Dijkstra's algorithm, then A* will be a bit less than 27% faster than Dijkstra's algorithm.

Algorithms that uses heuristic functions generally perform faster than searches that do not use heuristics, but the difference is not as much as the number of expansions decreases. The factors that play into how efficient an heuristic search algorithm functions depends on the quality of the heuristic. If the heuristic is good, then the search will be very efficient. If the heuristic is not good, the number of expanded nodes will likely be higher than usual, which would offset the difference between heuristic function searches and non-heuristic function searches.

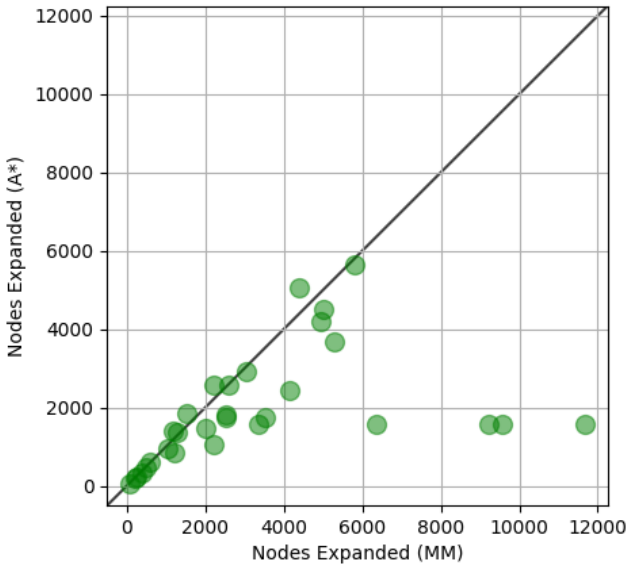
3.

Less, as shown on the plot.



4.

For the test cases within the assignment, it did not, we actually found that the number of expansions to be more if we pit mm search to A8*.



5.

From the plot, we see that for most part A* performed less searches. We can speculate on reason being that since A* it does not have to worry about encountering obstacles and such. The search would go around it as usual. However for a bidirectional search, like mm, if one side of the search encounters an obstacle that drives the search elsewhere, it would take the algorithm more expansion to meet the other search. This is especially true for MM since the algorithm will meet in the middle, one side will potentially have to backtrack and expand more nodes just to meet the other search in the middle.