Analyzing the Scatter Plot

1. The memory complexity of Dijkstra and Bi-BS results such overall distribution of points in the plot. Dijkstra has a memory complexity of $O(b^{d+1})$ assuming the optimal solution path has depth of d. Compared to Bi-Bs, since it perform searches in both forward and backward search at the same time instead of a single direction search, the depth of each search is $\frac{d}{2}$ instead of d. Thus, in the worst case, each direction search must expanded all nodes at depth $\frac{d}{2}$. Excluding some specifc cases, it results a smaller memory complxity of $O(2 \times b^{\frac{d}{2}})$ compared to $O(b^{d+1})$, which means that the number of nodes generated are smaller than Dijkstra. Thus, the value of horizontal coordinate (nodes expanded in Bi-Bs) is smaller than value of vertical coordinate (nodes expanded in Dijkstra).

2. By observation, when searches do not have a solution path because the start states are completely surrounding by the complete boundary, which block out the any possible path to the goal state, Bi-Bs expanded more nodes than Dijkstra. Thus, the data of these searches corresponding to the points that are clearly **above** the main diagonal. The terminating condition for Bi-Bs when there is no solution path for a search is that both the open forward list and open backward list must be empty or exhausted. For Dijkstra, there is only a single open list. The terminating condition for Dijkstra when there is no solution path for a search is that that single open list is empty or exhausted. For Dijkstra, it terminates earlier because it only need to expand all nodes around the start state before reaching the boundary. However, Bi-Bs need to wait for the backward search to expand all the nodes around the goal state such that the open backward list is empty, despite the open forward list is empty. Therefore, Bi-Bs need to expand more nodes than Dijkstra to reach the terminating condition when there is no solution path from start state to goal state.

3. That those points are clearly above the main diagonal indicates Dijkstras encountered worst (extreme) cases. In the worst case, Dijkstra needs to expand all the nodes within the search depth, and the number of states (nodes) encountered can grow exponentially with the search depth. The memory complexity (number of expansions) can be exponential for Dijkstra as stated in question 1. Also, Bi-Bs utilizes two directional searches orignated from start and goal. It expands the cheapest node from two open lists, and by following this condition, a optimal path will be found once two directional searches meet in a node. If there is a solution path between start and goal, two directional searches can constraint the algorithm from expanding too far and stay within in a general direction to some degree. Despite Bi-Bs does not use heuristic functions, Bi-Bs is "informed" to some of the extend. Compared to Dijkstra, it only expands in a single direction and it's uninformed. In the worst case, Dijkstra might expand unnecessary nodes in some directions and backtrack if a specific direction does not lead to the goal, especially when there are boundary around start and goal.