Road Navigation

To solve the road navigation problem, one of the most efficient algorithms to use is Dijkstra's algorithm, which is a greedy algorithm that calculates the shortest path between a starting node and all other nodes in the graph. Its worst-case time complexity is O(E*log(V)), where E is the number of edges and V is the number of vertices in the graph.

To implement Dijkstra's algorithm, we can use a priority queue to keep track of the unvisited nodes, ordered by their tentative distance from the starting node. We start by setting the distance of the starting node to zero and all other nodes to infinity. Then, we select the node with the smallest tentative distance from the priority queue and relax all its neighbors (i.e., update their tentative distances if they can be improved by going through the current node). We repeat this process until we visit all nodes or reach the ending node.

To keep track of the shortest path, we can maintain a parent array that stores the previous node on the shortest path to each node. Once we reach the ending node, we can backtrack from it to the starting node using the parent array and construct the path in reverse order.

The worst-case space complexity of this implementation is O(V), as we need to store the tentative distances and parent pointers for all nodes in the graph. Additionally, the priority queue can hold up to V nodes at a time. Therefore, the overall worst-case space complexity of the solution is O(V).