DFS(Depth first Search) :-

~ It plays an important role in our project as it helps in exploring and analyzing routes in a Traffic Management System.

~ It helps in detecting possible routes, finding connectivity, and managing congestion by traversing through all intersections (nodes).

-> Core Idea of our Algorithm:

" Starts from a source node (intersection) and explores as far as possible along each branch before backtracking. "

-> Let's understand with an example:

A --- B --- C

| |

D --- E --- F

If we start from node A, DFS will first go deep into connected routes before coming back to check other possible paths.

-> Data Structures Used:-

1. Graphs for representing intersections (nodes) and roads (edges)

2. Stack (can be implemented using recursion or manually) for backtracking

3. Visited Set/List for keeping track of explored intersections

-> Pseudocode:

First we will:-

1. Visit a node.

2. Mark it as visited.

3. Explore each unvisited neighbor.

4. When all neighbors are visited, backtrack to the previous node.

5. Continue until all reachable nodes are visited.

```
DFS(G, v) {
   visited[v] = 1;
   for all w adjacent to v {
      if (!visited[w]) {
         DFS(G, w);
      }
   }
}
```

-> Advantages of using this Algorithm:

1. Detects all possible routes in a city road network.

2. Helps in finding alternate paths during congestion.

3. Useful in traffic signal synchronization.

4. Helps in detecting cycles or closed loops in road networks.