I have chosen BFS to traverse the graph to find if a path exists or not between a source city and destination city because of the following reasons:

- DFS starts the traversal from the root node and explores the search as far as possible from the root node i.e. depth wise where as BFS starts traversal from the root node and then explore the search in the level by level manner i.e. as close as possible from the root node.
- Breadth First Search can be done with the help of queue i.e. FIFO implementation but Depth First Search can be done with the help of Stack i.e. LIFO implementations.In the Assignment it was indicated to use java.util.Queue class.So, I narrowed down my option to BFS.
- BFS uses more memory and is slower than DFS but in our example we do not have many cities between which we are traversing.So, I decided to go for BFS.
- BFS is useful in finding shortest path.BFS can be used to find the shortest distance between some starting node and the remaining nodes of the graph but DFS is not so useful in finding shortest path.
- DFS is used to perform a traversal of a general graph and the idea of DFS is to make a path as long as possible, and then go back (backtrack) to add branches also as long as possible.We might ending up traversing very deep into the graph in case of DFS when an alternative and a better trversal path exists instead.So, I have choosen BFS for efficient traversing .