首先来看广度优先遍历BFS（Breadth First Search），其主要思想是从起始点开始，将其邻近的所有顶点都加到一个队列（FIFO）中去，然后标记下这些顶点离起始顶点的距离为1.最后将起始顶点标记为已访问，今后就不会再访问。然后再从队列中取出最先进队的顶点A，也取出其周边邻近节点，加入队列末尾，将这些顶点的距离相对A再加1，最后离开这个顶点A。依次下去，直到队列为空为止。

BFS: From the beginning node, put all neighbor nodes into FIFO queue, then mark these neighbor nodes’ distance between the beginning node is 1, then mark the beginning node as visited, which won’t be visited again any more. After, pop the first in node from queue, repeat the above procures, until the whole queue is empty.

DFS（Depth First Search）深度优先搜索是从起始顶点开始，递归访问其所有邻近节点，比如A节点是其第一个邻近节点，而B节点又是A的一个邻近节点，则DFS访问A节点后再访问B节点，如果B节点有未访问的邻近节点的话将继续访问其邻近节点，否则继续访问A的未访问邻近节点，当所有从A节点出去的路径都访问完之后，继续递归访问除A以外未被访问的邻近节点。

DFS: From the beginning node, recursively visit all neighbor nodes, then visit all neighbor nodes’ neighbor. For example, if node A is the first neighbor of beginning node, and node B is a neighbor of A, then visit B after A. If there are unvisited neighbor of B, then visit these neighbors of B, otherwise continue visit unvisited neighbors of A. After all neighbors of A are visited, then start from other unvisited node except A.

Difference: If the tree is very deep and solutions are rare, DFS might take an extremely long time, but BFS could be faster. If the tree is very wide, a BFS might need too much memory, so it might be completely impractical. If solutions are frequent but located deep in the tree, BFS could be impractical. If the search tree is very deep you will need to restrict the search depth for DFS, anyway (for example with iterative deepening).