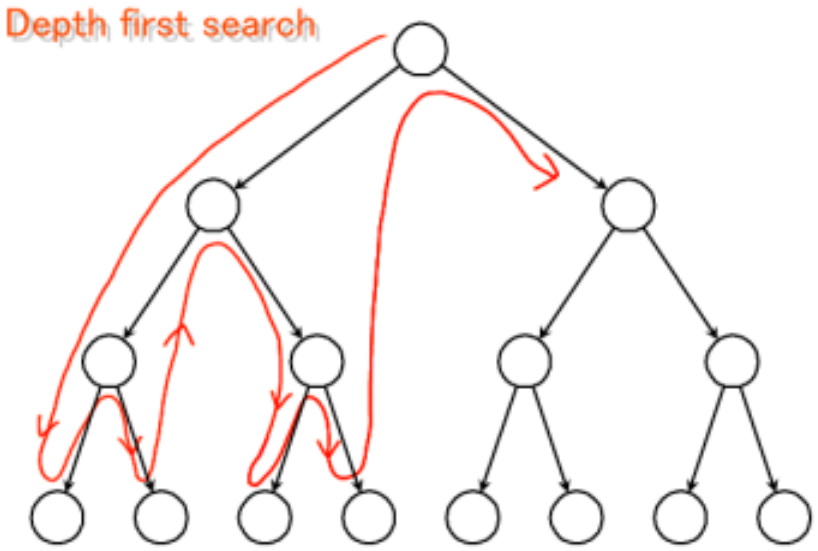
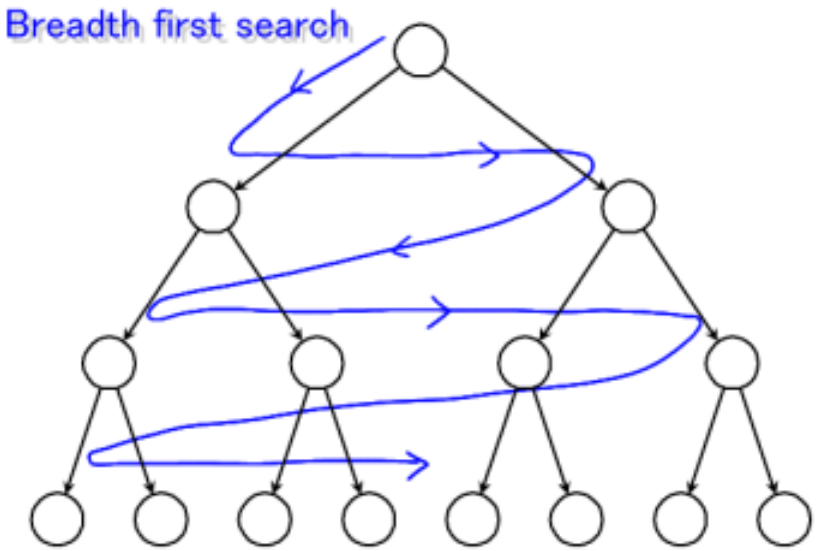
# *Graph Traversals*

# Graph Traversals




- Both take time: O(V+E)

# Use of a stack

- It is very common to use a stack to keep track of:
  - nodes to be visited next, or
  - nodes that we have already visited.
- Typically, use of a stack leads to a *depth-first* visit order.
- Depth-first visit order is “aggressive” in the sense that it examines complete paths.

# Topological Sort as DFS

- Do a DFS of graph G
- as each vertex v is “finished” (all of it’s children processed), insert it onto the front of a linked list
- return the linked list of vertices

- *why is this correct?*

# Use of a queue

- It is very common to use a queue to keep track of:
  - nodes to be visited next, or
  - nodes that we have already visited.
- Typically, use of a queue leads to a *breadth-first* visit order.
- Breadth-first visit order is “cautious” in the sense that it examines every path of length i before going on to paths of length i+1.

# Graph *Searching ???*

- Graph as state space (node = state, edge = action)
- For example, game trees, mazes, ...
- BFS and DFS each search the state space for a best move. If the search is exhaustive they will find the same solution, but if there is a time limit and the search space is large...
- DFS explores a few possible moves, looking at the effects far in the future
- BFS explores many solutions but only sees effects in the near future (often finds shorter solutions)