Q. Why does BFS/DFS need optimization?

Solving Problems by Searching

Well-Defined Problem

A problem is well-defined if 5 components are clearly stated

① Initial State

② Set of Possible Actions

③ Transition Model Description of what each action does or Successor Function

④ Goal Test Determines whether a given state is a goal state

⑤ Path Cost

Example: 8 Puzzle

Example Initial State ⇨

7	2	4
5		6
8	3	1

Fig 1

	1	2
3	4	5
6	7	8

Fig 2

⇦ Goal State

① Initial State: Any state can be designated as initial state

② Set of Possible Actions: {Left, Right, Up, Down}

③ Transition Model: Left - switches blank tile and the tile to the left
Right - . . . .
Up - . . . .
Down - . . . .

④ Goal Test: Does state match Fig-2

⑤ Path Cost: Each step costs 1

Q. Example Scenario/Problem given. Agent & Environment type given. Goal given. Determine actions taken by agent

→ Convert given problem to Well-defined problem (Define 5 components)

Search Strategy Evaluation

① Completeness Whether the search strategy will find a solution when there is one

② Optimality Whether the search strategy finds the highest quality solution when there are multiple solutions

③ Time Complexity Amount of time required to find a solution

④ Space Complexity Amount of memory required to perform the search

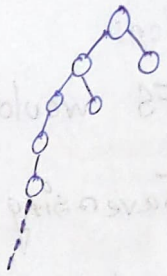
Q. Scenario Given. Which of BFS, DFS, IDS would be best? What problems will a certain search strategy have in the scenario? (BFS, DFS or IDS)

Branching Factor

If Graph/Tree has high number of branches (High Branching Factor), BFS faces issue.

Depth Factor

If Graph/Tree has high depth on a lot of child nodes (High Depth Factor), DFS faces issue



Note: All graph/tree search problems are uninformed search problem. (BFS, DFS, IDS).

Q. Complexity comparison of search strategies given. Scenario given. Analyze which search strategy is optimal

Uninformed Search

○ Breadth First Search	○ Depth First Search	○ Bidirectional Search
> Uniform Cost Search	> Depth-Limited Search	
	> Iterative Deepening Search	

b = Branching Factor

d = Depth of Shallowest Solution

m = Maximum Depth of Search Tree

l = Depth Limit

Uninformed Search

Search strategies in which

- there is no information about number of steps
- path cost from current state to goal is unknown

① Breadth First Search or BFS

Parent (or Root) node is expanded first, then child (or Successor) nodes are expanded.

Analysis

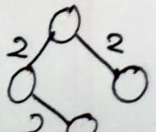
ⓐ Complete? Yes, if b is finite

ⓑ Time Complexity $O(b^{d+1})$

ⓒ Space Complexity $O(b^{d+1})$

where,
b = Branching factor
d = Depth of Shallowest Solution

ⓓ Optimal? Yes, if path costs are identical



Q. Condition given - Except own branch and root, no repeated node state allowed during backtracking in search. Will the search be complete? (Can goal be reached)

Problems of BFS

> Space and Time complexity are enormous if branching factor b is large

> Optimal only if path costs are identical

Solves ↗

1.1 Uniform Cost Search [Optimization of BFS]

Parent (or Root) node is expanded first, then lowest cost child node is expanded

Analysis

ⓐ Complete? Yes if b is finite AND all step costs exceed some positive constant ϵ [all path costs $\geq \epsilon$]

ⓑ Time Complexity $O(b^{1+[C^*/\epsilon]})$

ⓒ Space Complexity $O(b^{1+[C^*/\epsilon]})$

where C^* is cost of optimal solution & every path cost is $\geq \epsilon$, a positive constant

ϵ = Minimum Positive Path Cost

ⓓ Optimal!? Yes

② Depth First Search

Always expands node at the greatest depth first

Analysis

ⓐ Complete? No. A Path may be infinite/looping

ⓑ Time Complexity $O(b^m)$

ⓒ Space Complexity $O(bm)$

ⓓ Optimal? No. Does not guarantee best solution

where,
b = Branching factor
m = Maximum depth of search tree

Note: Space Complexity of DFS is much less compared to BFS (Store only a single path from root to leaf node and remaining unexpanded sibling nodes for each node on the path)

Problems of DFS
> Not Complete if a path is infinite/looping > Not Optimal

②.1 Depth Limited Search [Optimization of DFS]

DFS with a depth limit (Maximum depth upto which nodes are expanded)

Solves

Analysis

ⓐ Complete? No if $l < d$

ⓑ Time Complexity $O(b^l)$

ⓒ Space Complexity $O(bl)$

ⓓ Optimal? No if $l > d$

where,
l = Depth Limit
d = Depth of Shallowest Solution
b = Branching factor

(2.2) Iterative Deepening Search [Optimization of DFS]

Iteratively conducts DFS with depth limits = 0, 1, 2...

Combines benefits of BFS & DFS

Analysis

(a) Complete? : Yes if b is finite

(b) Time Complexity $O(b^d)$

(c) Space Complexity $O(bd)$

(d) Optimal? Yes if path costs are identical

where,
b = Branching Factor
d = Depth of Shallowest Solution

Solves All Problems of DFS

(2.3) Backtracking Search

Similar to DFS, but expands only one successor/child node at a time rather than all successors

Space Complexity $O(m)$ [whereas DFS is $O(bm)$]

Completeness, Optimality & Time Complexity same as DFS

Note: Iterative Deepening Search is the preferred uninformed search method when search space is large and depth of solution is unknown

(2) Iterative Lengthening Search

Iteratively conducts DFS with cost limit increments

Analysis

(a) Complete? Yes if b is finite AND path costs $\geq \epsilon$

(b) Time Complexity $O\left(\frac{C}{\epsilon} \cdot b^{\frac{C}{\epsilon}}\right)$

(c) Space Complexity $O\left(b^{\frac{C}{\epsilon}}\right)$

(d) Optimal? Yes if all path costs $\geq \epsilon$

where,
b = Branching Factor
C = Cost of cheapest solution
ϵ = Minimum Positive path cost

Note: Less Space Complexity than Uniform-Cost Search, but more Time Complexity

(3) Bidirectional Search

Runs two simultaneous searches (One forward from initial state, one backward from goal state)

Analysis

(a) Complete? Yes if b is finite AND both searches use BFS

(b) Time Complexity $O(b^{d/2})$

(c) Space Complexity $O(b^{d/2})$

(d) Optimal? Yes if path costs are identical AND both searches use BFS

Avoiding Repeated States

> There is a possibility of expanding states that have already been expanded on another path.

> May cause the path to be infinite

Solution:

> Define a set containing every expanded node (so far)

> If the current node matches a node on the set, discard it (do not expand it)

See [Slide - 177] for Summary of Search Strategy Evaluation

Q. Search Strategy & Scenario given. Show an analysis of Time and Space Complexity

Example: Romania [Slide-109]

Initial State: Timisoara Goal State: Fagaras

Problem-1: Reach Goal State (not with minimum distance just reach goal)

Solution:

> DFS would work better

> Traversing nodes depthwise would allow agent to reach goal faster.

Problem-2: Reach Goal State with minimum distance.

Solution:

> Uniform-Cost Search (or Iterative Lengthening Search) would work better, as optimality is ensured in these strategies