✱Ch. 3 : Problem solving and Search:-

✱ Problem solving agents:-

- Intelligent agents are supposed to act in such a way that the environment goes through a sequence of states to maximize performance measure.

- For an agent to solve a problem, it needs to adopt a goal. There are a few steps for the agent to solve a problem:

(1) Goal formulation: it organizes finite steps to formulate a target/goals which requires some action to achieve

(2) Problem formulation: it decides what action should be taken to achieve the formulated goal.

✯ Problem types:-

- Single state problem: the solution is a sequence. Agents know exactly which state it will be in.

- Conformant problem: the solution (if any) is a sequence. Agents may have no idea where it is.

- Contingency problem: the solution is a contingent plan. The agent percepts provide new information about the current state

- Exploration problem: Unkown state space. This kind of problem is a real world problem. The agent experiments and gradually discovers what its actions do, and what sort of states exist.

* Single-State problem formulation:

- A problem is defined by 4 items:

① Initial state: The state where the agent starts in to reach a certain goal

② Successor function: returns the set of reachable states

③ Goal test: The test simply checks if the agent reached the goal.

④ Path cost: A function that assigns a cost to a path. The cost of the is the sum of the costs of the individual actions along the path.

- Solution: a sequence of actions leading from the initial state to the goal state.

★ Selecting a state space:-

- State space must be abstracted for problem solving.

↑ removing details

- Abstract state ⇒ set of real states
- Abstract action ⇒ complex combination of real actions
- Abstract solution ⇒ set of real paths that are solutions in the real world.
- Each abstract action should be easier than the original problem.

☆ Tree Search algorithms :-

- The basic idea is an exploration of state space by generating successors of explored states.

- A state is a physical configuration

- A node is a data structure constituting part of a search tree.

☆ Search strategies :

- A strategy is defined by picking the order of node expansion.

- How do we evaluate a strategy ?

  - Completeness, time complexity space complexity, optimality.

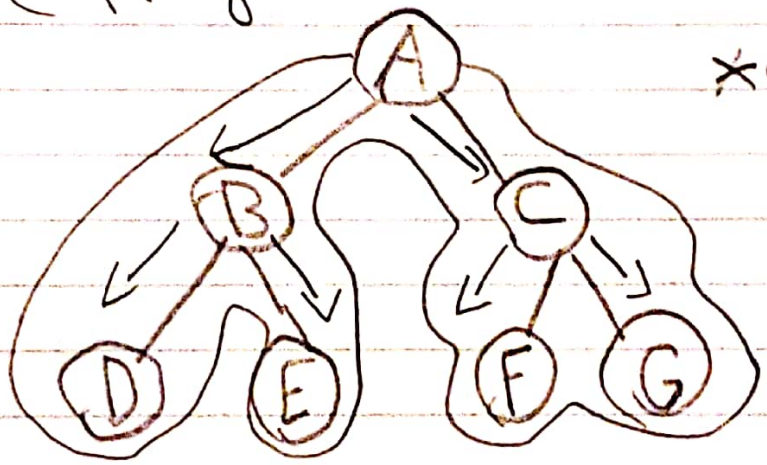
* Uninformed Search strategies:-

① BFS:

- Expand the shallowest unexpanded node.

*The fringe here is a FIFO queue

check slide 45



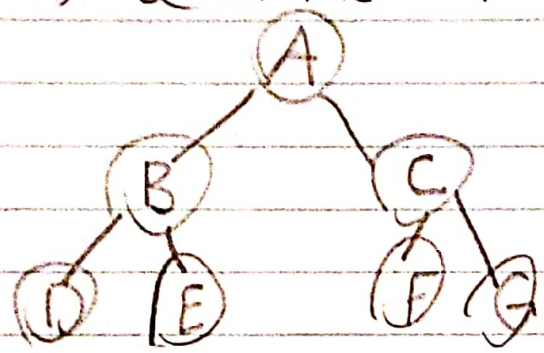
- The idea here is visiting and exploring all nodes and their neighbors/children

- Start at root A, From A you explore its children and visit them, so now we are at B and C. Now we need to explore their children. The order doesn't matter but we typically go with the alphabetical order. Now we explore B and go visit D and E, then since D and E have no children we go back and explore C, we move to F and G and we return the path

② DFS :-

- Expand deepest unexpanded node

* Fringe is a LIFO queue



- The idea here is exploring a node then explore and visit one of its children then explore that child in the same way

eg. We start at A, then we will move to B, and from B we will explore its children the same way we explored A. Now we go to D, D has nothing, so we go to E, E has nothing so we go to C, and explore it just like B, so we go to F, then we go to G.

- This the order of exploration

A, B, D, E, C, F, G

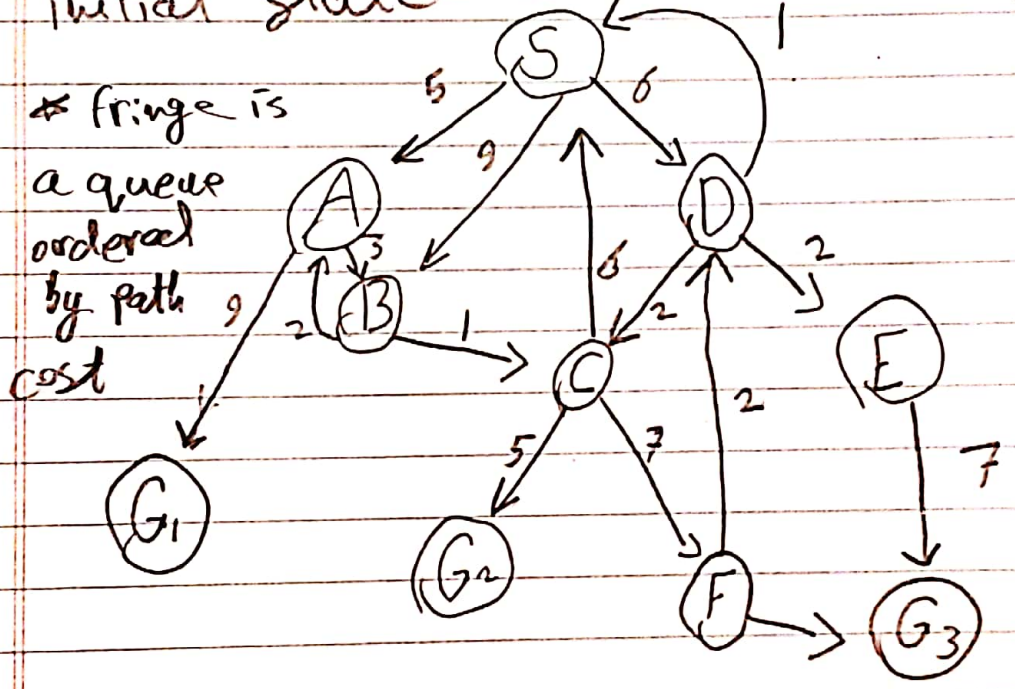
✱ UCS:

Search space

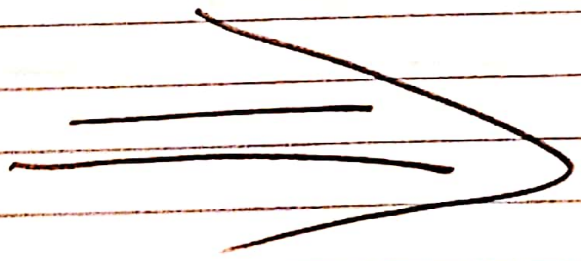
* G1, G2, and G3 are goals "

initial state = S

* fringe is a queue ordered by path cost

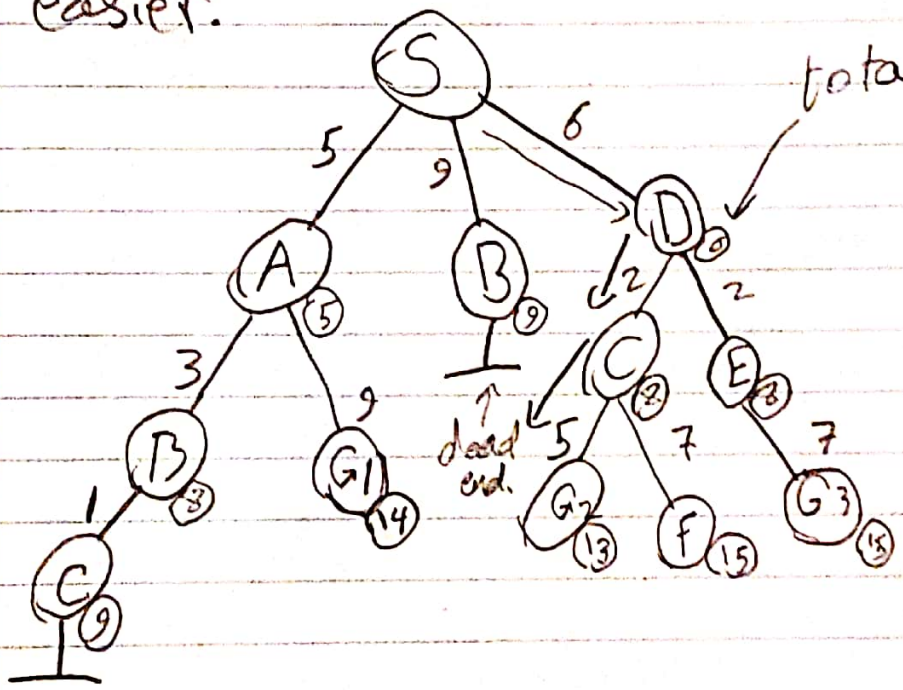


- The idea here is to reach one of the goals with the minimum cost. We visit and explore nodes depending on the cost



*Continued UCS:-

-rewrite the tree to make it easier.



-We basically explored and visited all the nodes once and we calculate the total cost of each path to the goal and take the cheapest. In this case

$S \rightarrow D \rightarrow C \rightarrow G_2$

* Depth limited search

- It's depth first search but with a depth limit.

---

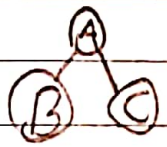
* Iterative deepening search.

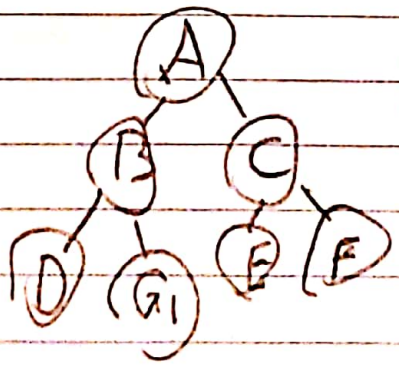
- Just like depth limited search, it's DFS with a depth limit.

We start with depth limit = 0

and then we run DFS. If there are no goals in that limit we increase it by 1.. and so on. eg.

L = 0  no goal.

L = 1  no goal

L = 2 

goal at limit = 2 so we run DFS on that limit and return the path to the goal.

✱ Repeated States :-

- Failure to detect repeated states can turn a linear problem into an exponential one.

✱ Slide's notes :-

- Problem formulation usually requires abstracting away real-world details to define a state space that can feasibly be explored.

- IDS uses only linear space