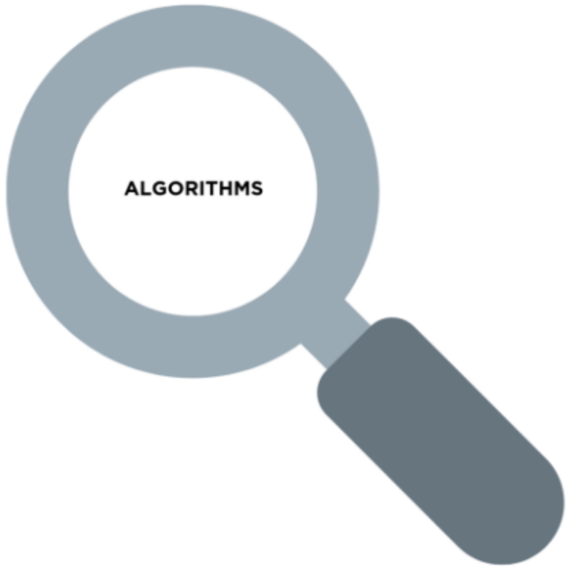
AI SEARCH ALGORITHMS

Agenda

- *Heuristic Search - Introduction*
- *Hill Climbing*
- *Best First Search*
- *A* Algortihm*
- *Questions*



Heuristic Search - Introduction

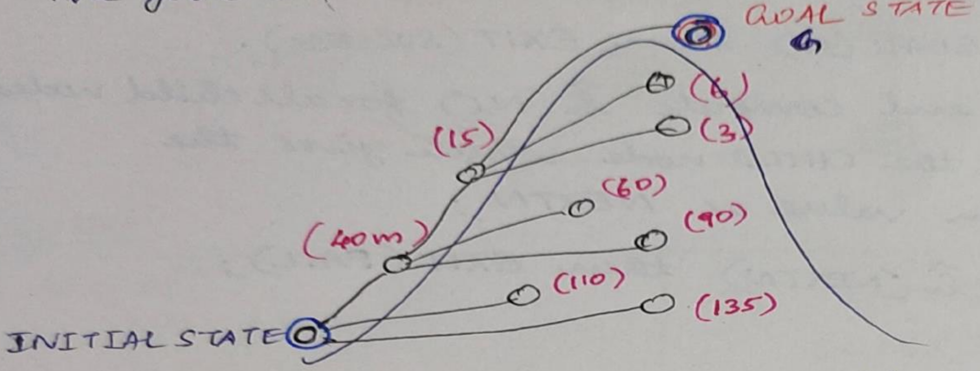
- Blind search techniques - searches blindly through the state-space - require too much time or memory - if the search space is big.
- **Examples:** Depth-First search and Breadth-First search
- Neither uses any knowledge about the specific domain in question to search through the state-space in a more directed manner.
- *How to reduce the running time?*
  - Use problem-specific knowledge to pick which states are better candidates
- **Solution - Informed (or Heuristic) methods -** search is carried out by using additional information to determine the next step towards finding the solution.
- Informed search methods are more efficient, low in cost and high in performance as compared to the uninformed search methods.
- **Examples** of such algorithms - **Hill Climbing and Best First Search**

Hill Climbing - Introduction

- ***Hill Climbing is a heuristic search used for mathematical optimisation problems in the field of Artificial Intelligence.***
- A local search algorithm - moves continuously upward (increasing) until the best solution is attained (when the peak is reached).
- Has a node that comprises two parts: **state** and **value**.
- Begins with a non-optimal state (the hill's base) and upgrades this state until a certain precondition is met **(heuristic function)**.
- The process of continuous improvement of the current state of iteration can be termed as climbing – hence termed as a ***hill-climbing algorithm.***
- **Objective** - to attain an optimal state that is an upgrade of the existing state.
- When the current state is improved, the algorithm will perform further incremental changes to the improved state.
- This process will continue until a peak solution is achieved, where, further improvements are impossible.

Hill Climbing

* Hill climbing – used for optimization.
* The SSM is like a hill, where the goal is perched at the top of the hill.
* The goal is searched for, from the foothills.



* HC algorithm starts generating child nodes
  - Child node 1 – 40 m to reach G
  - Child node 2 – 110 m "

  ...
* From one child node, other child nodes are generated.
* Two conditions:
  1) Select the minimal child – which helps to reach G at minimal cost.
  2) Also, check that the child node is taking up the hill
     ie) Cost of child < Cost of parent.

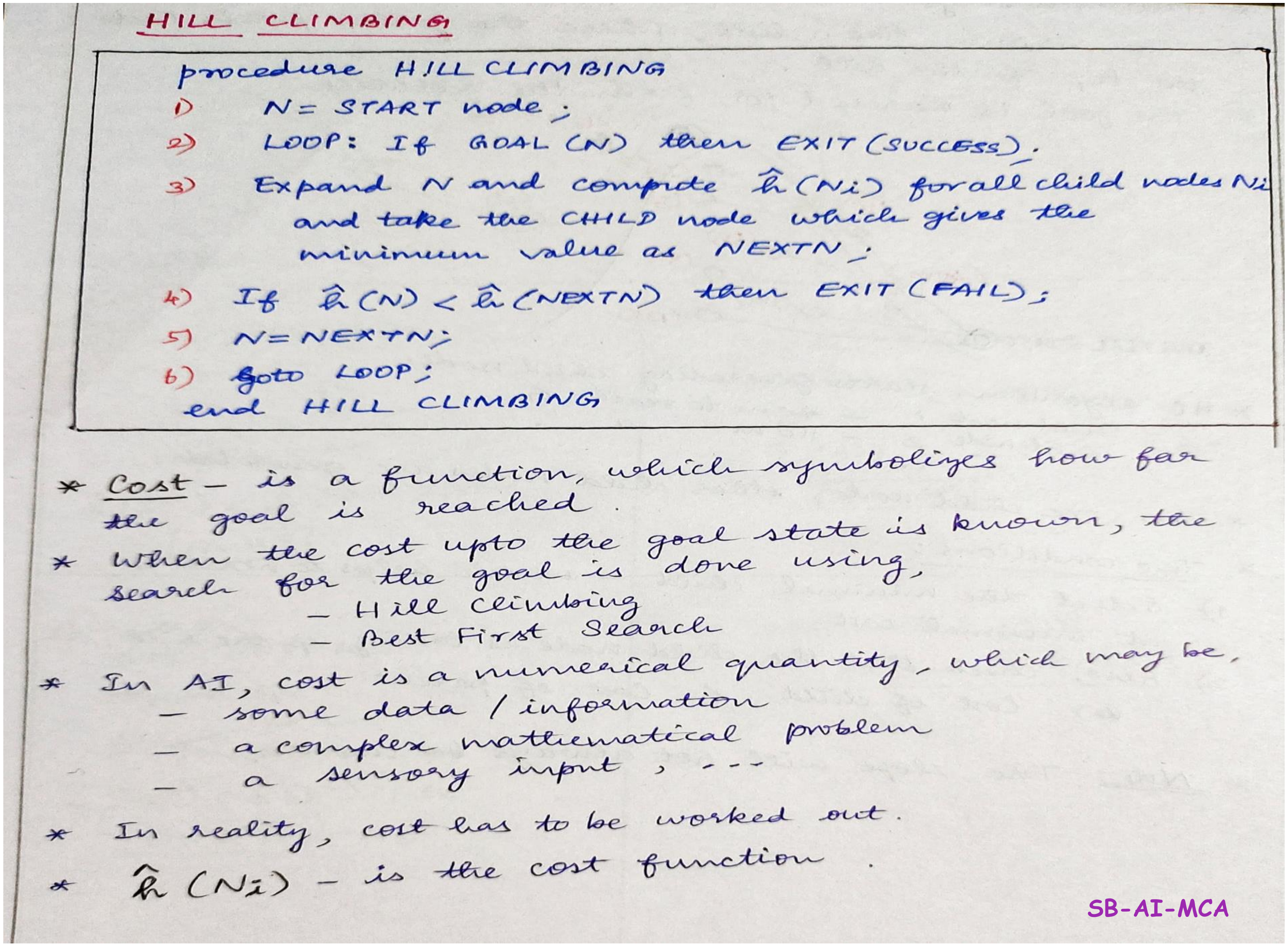
* Note: The slope will not always be smooth.

SB-AI-MCA

Hill Climbing – Features

1. **Greedy approach:** Moves in a direction in which the cost function is optimized - enables the algorithm to establish local maxima or minima.
2. **No Backtracking:** Only works on the current state and succeeding states (future) - does not look at the previous states.
3. **Generate and Test variant:** Generate-and-test technique provides feedback - to decide the direction of move in the search space. (whether up or down the hill).
4. **Incremental change:** The algorithm improves the current solution by incremental changes.

Hill Climbing – Algorithm



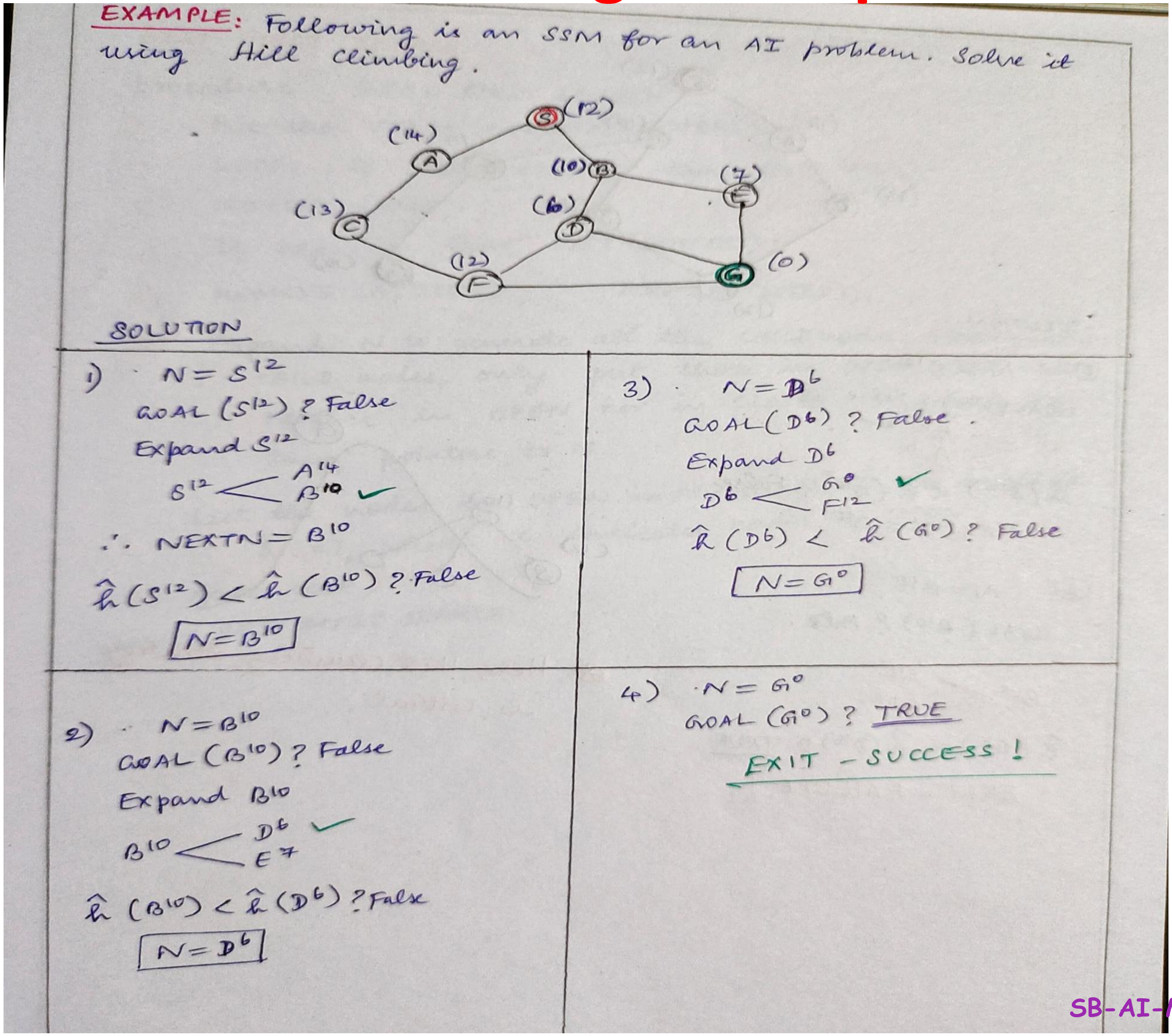
HILL CLIMBING

```
procedure HILL CLIMBING
1)    N = START node;
2)    LOOP: If GOAL (N) then EXIT (SUCCESS).
3)    Expand N and compute $\hat{h}(N_i)$ for all child nodes $N_i$
        and take the CHILD node which gives the
        minimum value as NEXTN;
4)  If $\hat{h}(N) < \hat{h}(NEXTN)$ then EXIT (FAIL);
5)  N = NEXTN;
6)  goto LOOP;
end HILL CLIMBING
```

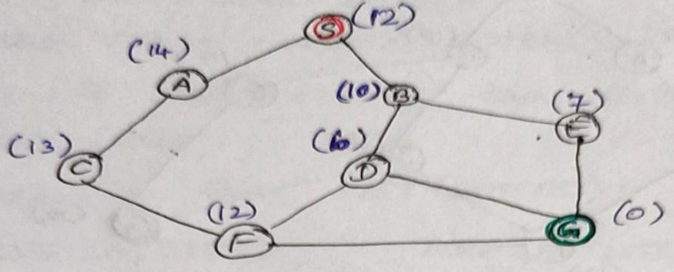
* Cost – is a function, which symbolizes how far the goal is reached.
* When the cost upto the goal state is known, the search for the goal is done using,
  - Hill climbing
  - Best First Search
* In AI, cost is a numerical quantity, which may be,
  - some data / information
  - a complex mathematical problem
  - a sensory input, . . .
* In reality, cost has to be worked out.
* $\hat{h}(N_i)$ – is the cost function.

SB-AI-MCA

Hill Climbing - Example



EXAMPLE: Following is an SSM for an AI problem. Solve it using Hill climbing.



SOLUTION

1) $N = S^{12}$

GOAL (S^{12}) ? False

Expand S^{12}

$S^{12} < \begin{matrix} A^{14} \\ B^{10} \checkmark \end{matrix}$

$\therefore$ NEXTN $= B^{10}$

$\hat{h}(S^{12}) < \hat{h}(B^{10})$? False

$\boxed{N = B^{10}}$

2) $N = B^{10}$

GOAL (B^{10}) ? False

Expand B^{10}

$B^{10} < \begin{matrix} D^{6} \checkmark \\ E^{7} \end{matrix}$

$\hat{h}(B^{10}) < \hat{h}(D^{6})$? False

$\boxed{N = D^{6}}$

3) $N = D^{6}$

GOAL (D^{6}) ? False.

Expand D^{6}

$D^{6} < \begin{matrix} G^{0} \checkmark \\ F^{12} \end{matrix}$

$\hat{h}(D^{6}) < \hat{h}(G^{0})$? False

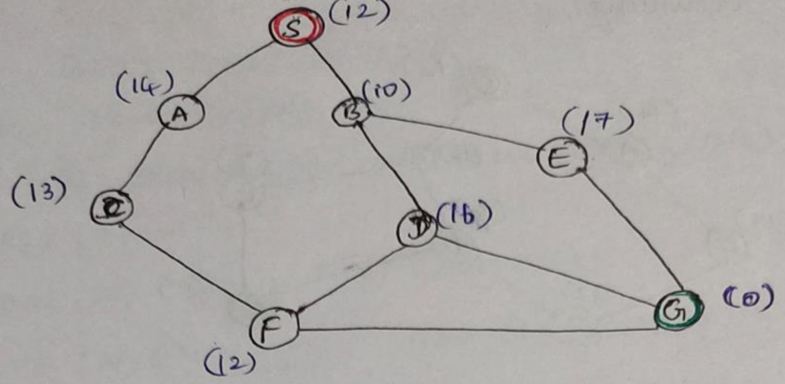
$\boxed{N = G^{0}}$

4) $N = G^{0}$

GOAL (G^{0}) ? TRUE

EXIT - SUCCESS !

Hill Climbing - Example



<u>SOLUTION</u>

① $N = S^{12}$

GOAL(S^{12})? False

$$\begin{tikzcd} & B^{10} \checkmark \\ S^{12} \arrow[ur, dash] \arrow[dr, dash] & \\ & A^{14} \end{tikzcd}$$

$\hat{h}(S^{12}) < \hat{h}(B^{10})$? False

NEXTN = B^{10}.

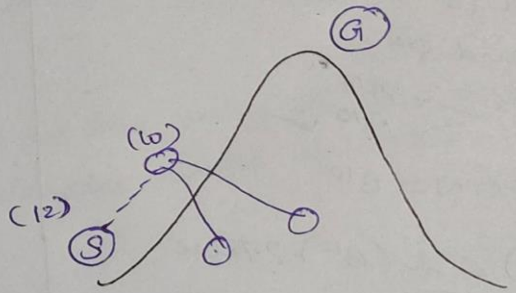
② $N = B^{10}$

GOAL(B^{10})? False.

$$\begin{tikzcd} & D^{16} \\ B^{10} \arrow[ur, dash] \arrow[dr, dash] & \\ & E^{17} \end{tikzcd}$$

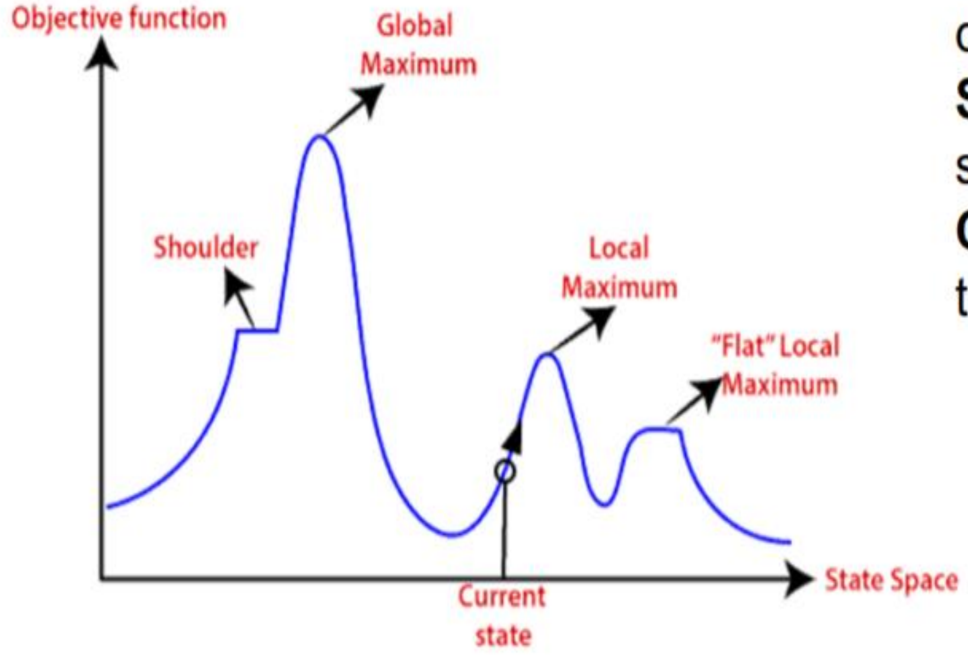
$\hat{h}(B^{10}) < \hat{h}(D^{16})$? <u>TRUE</u>

EXIT – FAILURE!



⊛ Here, Hill climbing strategy is critical.

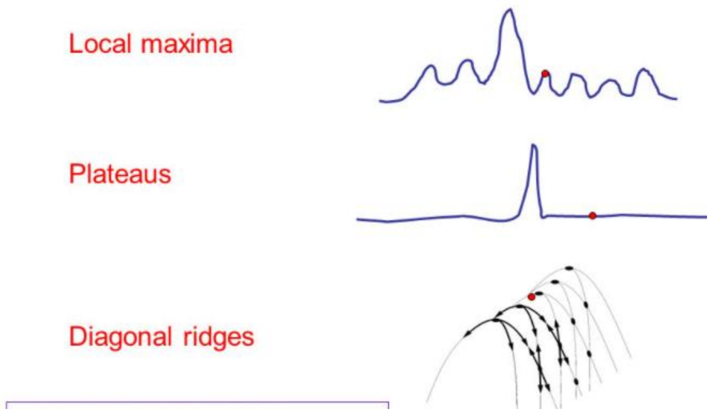
Hill Climbing - Landscape



Global Maximum: It is the highest point on the hill, which is the goal state.
Local Maximum: It is the peak higher than all other peaks but lower than the global maximum.
Flat local maximum: It is the flat area over the hill where it has no uphill or downhill. It is a saturated point of the hill.
Shoulder: It is also a flat area where the summit is possible.
Current state: It is the current position of the person.

Hill Climbing - Drawbacks

- **Local maximum:** A state that is better than all of its neighbors, but not better than some other states far away.
- **Plateau:** A flat area of the search space in which all neighboring states have the same value.
- **Ridge:** The orientation of the high region, compared to the set of available moves, makes it impossible to climb up. However, two moves executed serially may increase the height.



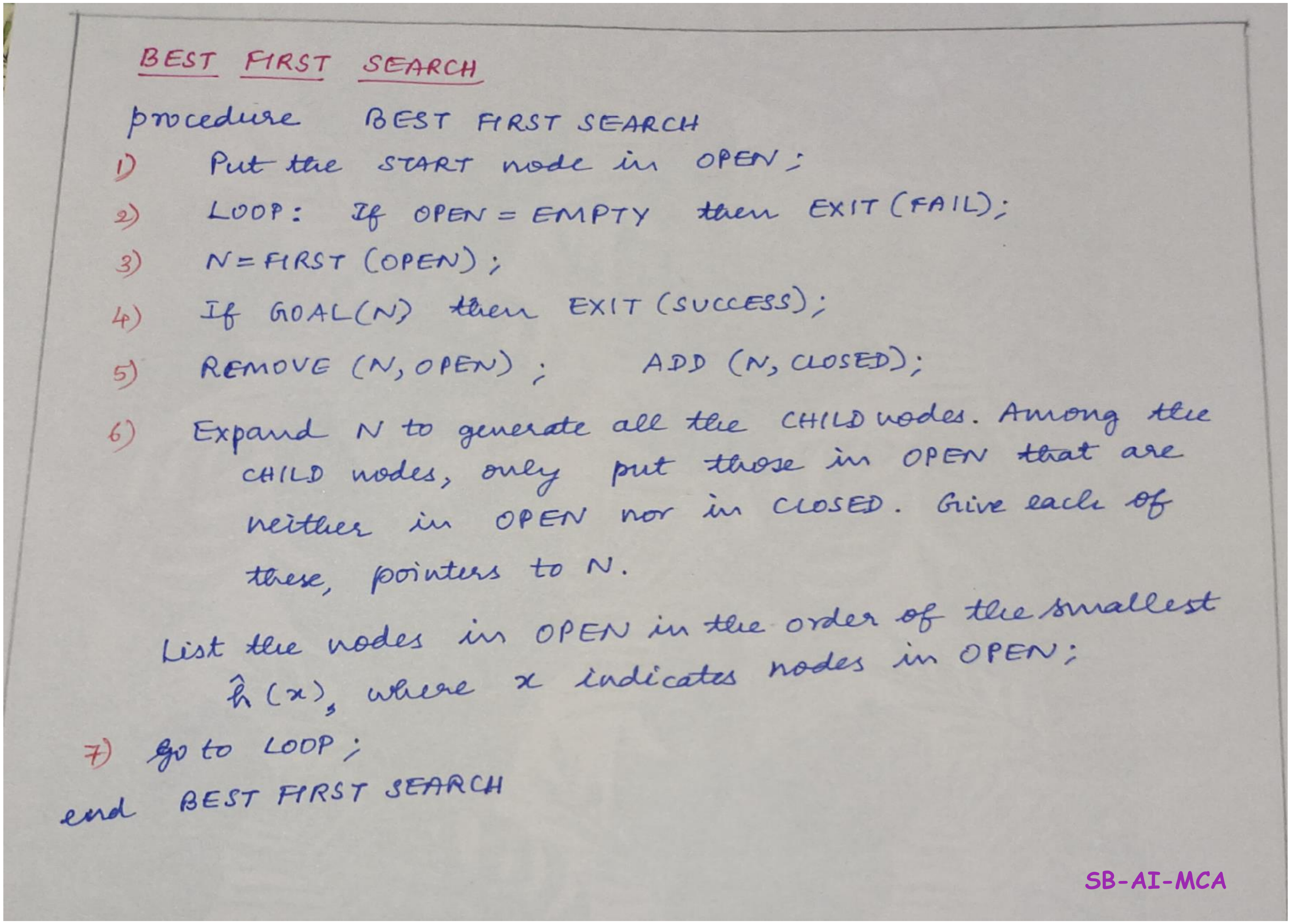
Hill Climbing – Overcoming the drawbacks

- **Backtrack** to some earlier node and try going in a different direction.(good way to deal with Local Maxima)
- Make a **big jump** to try to get in a new section.(good way to deal with Plateaus)
- **Moving** in several directions at once. (good way to deal with Ridges)

Best First Search

- Is an instance of graph search algorithm.
- Idea - a node is selected for expansion based o evaluation function f (n) - to decide which among the various available nodes is the most promising (or 'BEST') before traversing to that node.
- Uses the concept of a **priority queue** and **heuristic search**
- Uses two lists for tracking the traversal:
  - **'OPEN'** list – to keep track of current 'immediate' nodes available for traversal
  - **'CLOSED'** list - to keep track of the nodes already traversed.
- Serves as combination of DFS and BFS algorithms.
- Often referred to as **greedy algorithm –** since it quickly attacks the most desirable path as soon as its heuristic weight becomes the most desirable.

Best First Search



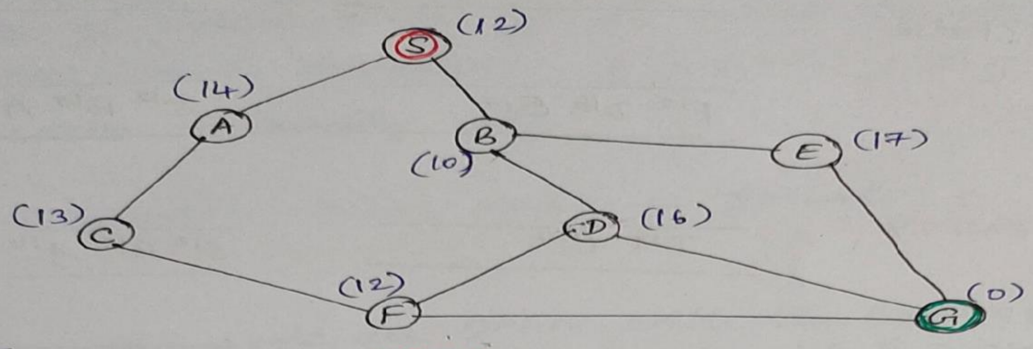
BEST FIRST SEARCH

```
procedure BEST FIRST SEARCH
1) Put the START node in OPEN;
2) LOOP: If OPEN = EMPTY then EXIT (FAIL);
3) N = FIRST (OPEN);
4) If GOAL(N) then EXIT (SUCCESS);
5) REMOVE (N, OPEN);    ADD (N, CLOSED);
6) Expand N to generate all the CHILD nodes. Among the
       CHILD nodes, only put those in OPEN that are
       neither in OPEN nor in CLOSED. Give each of
       these, pointers to N.
   List the nodes in OPEN in the order of the smallest
       $\hat{h}(x)$, where $x$ indicates nodes in OPEN;
7) go to LOOP;
end BEST FIRST SEARCH
```

SB-AI-MCA

Best First Search - Example

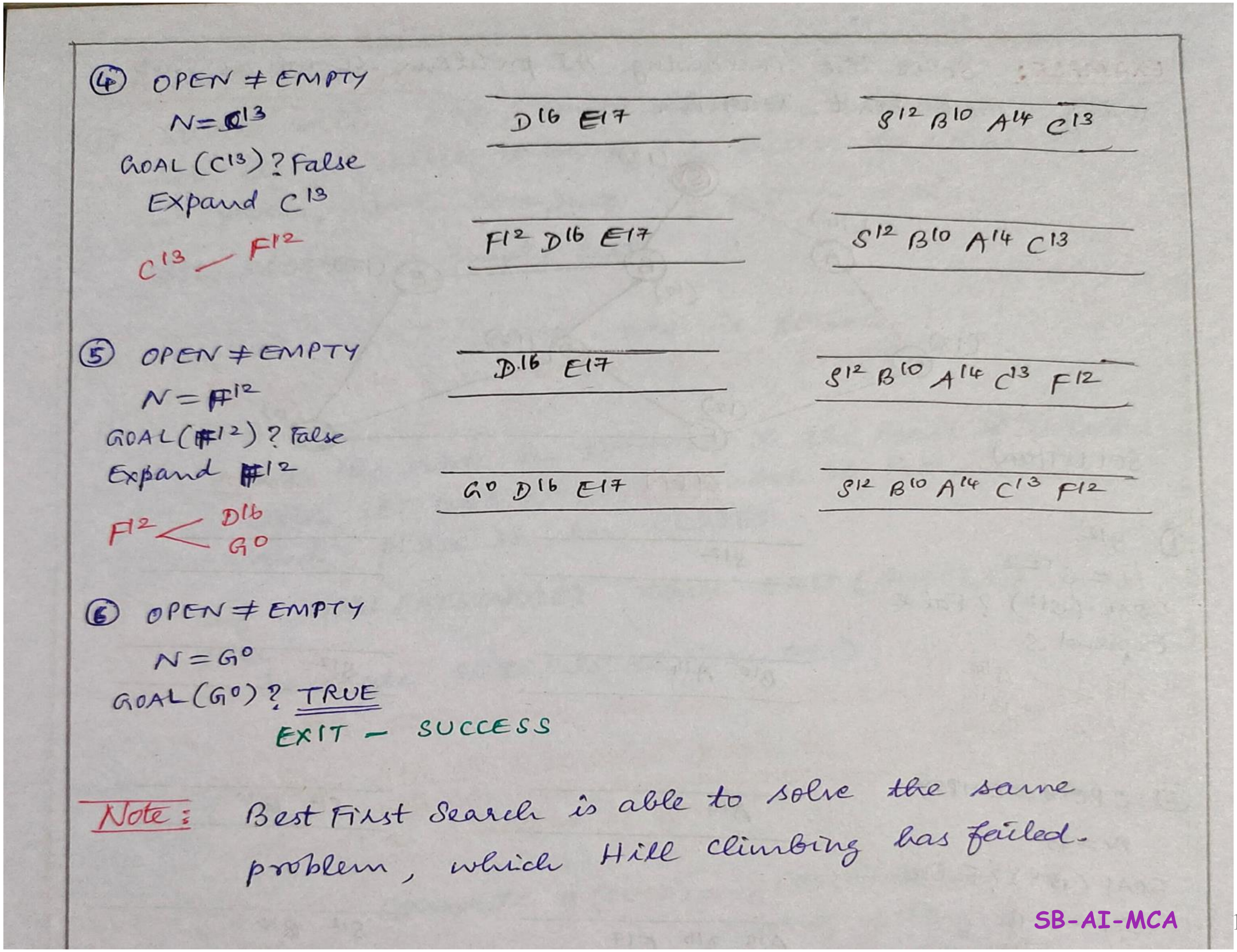
EXAMPLE: Solve the following AI problem shown as SSM, using Best First Search.



SOLUTION

	OPEN	CLOSED
① S^{12}		
$N = S^{12}$	S^{12}	
GOAL (S^{12}) ? False		
Expand S		
$$\begin{tikzcd} & A^{14} \\ S^{12} \arrow[ur, dash] \arrow[dr, dash] & \\ & B^{10} \end{tikzcd}$$	B^{10} A^{14}	S^{12}
② OPEN ≠ EMPTY		
$N = B^{10}$	A^{14}	S^{12} B^{10}
GOAL (B^{10}) ? False		
Expand B		
$$\begin{tikzcd} & D^{16} \\ B^{10} \arrow[ur, dash] \arrow[dr, dash] & \\ & E^{17} \end{tikzcd}$$	A^{14} D^{16} E^{17}	S^{12} B^{10}
③ OPEN ≠ EMPTY		
$N = A^{14}$	D^{16} E^{17}	S^{12} B^{10} A^{14}
GOAL (A^{14}) ? False		
Expand A		
A^{14} —— C^{13}	C^{13} D^{16} E^{17}	S^{12} B^{10} A^{14}

Best First Search - Example



④ OPEN ≠ EMPTY

$N = C^{13}$

GOAL (C^{13}) ? False

Expand C^{13}

C^{13} — F^{12}

D^{16} E^{17}	S^{12} B^{10} A^{14} C^{13}
F^{12} D^{16} E^{17}	S^{12} B^{10} A^{14} C^{13}

⑤ OPEN ≠ EMPTY

$N = F^{12}$

GOAL (F^{12}) ? False

Expand F^{12}

$$F^{12} \begin{cases} D^{16} \\ G^{0} \end{cases}$$

D^{16} E^{17}	S^{12} B^{10} A^{14} C^{13} F^{12}
G^{0} D^{16} E^{17}	S^{12} B^{10} A^{14} C^{13} F^{12}

⑥ OPEN ≠ EMPTY

$N = G^{0}$

GOAL (G^{0}) ? <u>TRUE</u>

EXIT — SUCCESS

Note: Best First Search is able to solve the same problem, which Hill climbing has failed.

Best First Search

Advantages

- Best first search can switch between BFS and DFS by gaining the advantages of both the algorithms.
- This algorithm is more efficient than BFS and DFS algorithms.

Disadvantages

- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS; is not optimal.
- Sometimes, it covers more distance than expected.