Types of Search Techniques

1. Blind or Uniformed Search
2. Heuristic or Informed Search

Blind or Uniformed Search

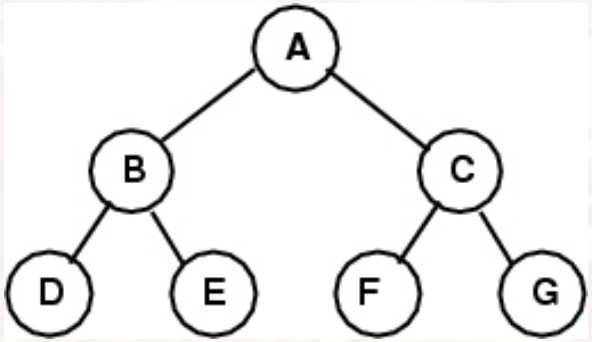
- Blind or Uniformed Search means searching without information.
- Do not have additional information about states beyond problem definition.
- Total search space is looked for solution.
- No information is used to determine preference of one child over other.
- Examples: Breadth First Search(BFS), Depth First Search(DFS)

Heuristic or Informed Search

- Heuristic or Informed Search means searching with information.
- Some information about problem space (heuristic) is used to compute preference among the children for exploration and expansion.
- Examples: Best First Search, Hill Climbing, Constraint Satisfaction etc.
- Heuristic function: It maps each state to a numerical value which depicts goodness of a node.
- H(n)=value where , H() is a heuristic function and ‘n’ is the current state.

Depth-first Search

In depth-first search, we start with the root node and completely explore the descendants of a node before exploring its siblings (and siblings are explored in a left-to-right fashion).

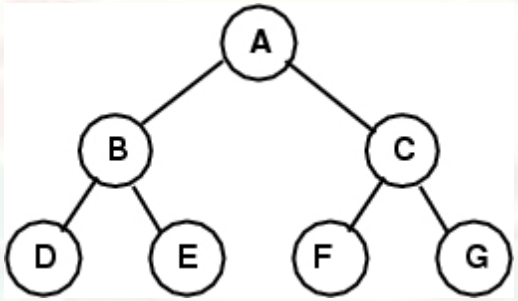


Depth-first traversal: A -> B -> D -> E -> C -> F -> G

Breadth-first Search

The problem with depth-first search is that we may get lost in an infinite branch, while there could be another short branch leading to a solution.

Such problems can be overcome by using breadth-first search, where we explore (right-hand) siblings before children.



Breadth-first traversal: A -> B -> C -> D -> E -> F -> G

Advantages of DFS:-

- Memory requirements in DFS are less compared to BFS as only nodes on the current path are stored.
- DFS may find a solution without examining much of the search space of all.

Disadvantages of BFS:-

- This search can go on deeper and deeper into the search space and thus can get lost. This is referred to as blind alley.

Advantages of BFS:-

- BFS is a systematic search strategy, all nodes at level n are considered before going to n+1th level.
- If any solution exists then BFS guarantees to find it.
- If there are many solutions , BFS will always find the shortest path solution.
- Never gets trapped exploring a blind alley.

Disadvantages of BFS:-

- All nodes are to be generated at any level. So even unwanted nodes are to be remembered. Memory wastage.
- Time and space complexity is exponential type.

Informed/ Heuristic Search Techniques are:-

- **Generate and Test**
- **Hill Climbing**
- **Best- first Search**
- **Problem Reduction**
- **Constraint Satisfaction**
- **Backtracking**

Generate and Test

Generate-and-Test search algorithm is a very simple algorithm that guarantees to find a solution if done systematically and there exists a solution.

ALGORITHM:-

1. Generate a possible solution.
2. Test to see if this is the expected solution.
3. If the solution has been found quit else go to step 1.

Potential solutions that need to be generated vary depending on the kinds of problems. For some problems the possible solutions may be particular points in the problem space and for some problems, paths from the start state.

Hill Climbing Search

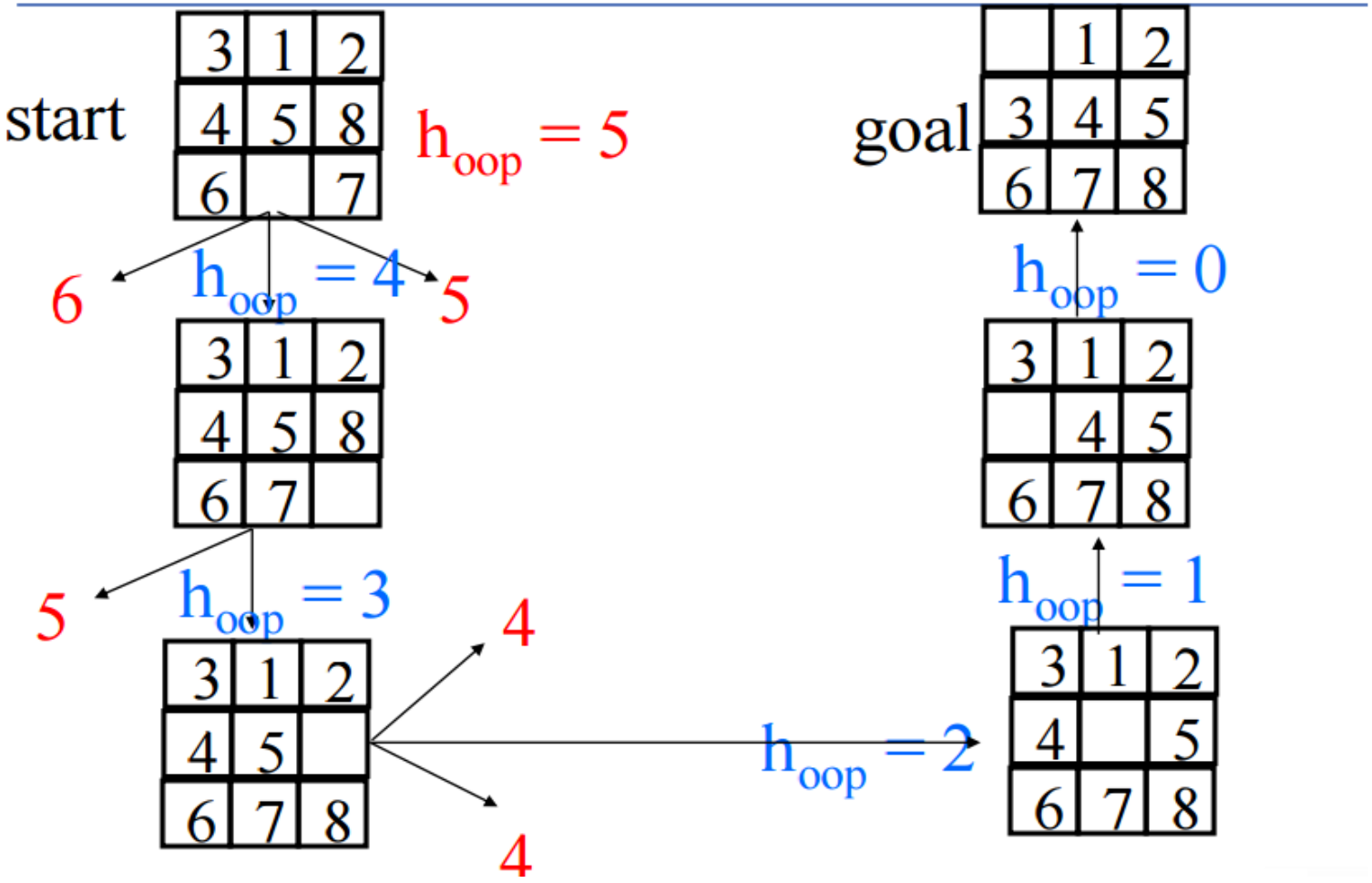
Hill Climbing = Generate and Test + Direction to move

- Hill Climbing is a variant of Generate and Test in which feedback from the test procedure is used to help the generator decide which direction to move in the search space.
- In Generate and Test procedure, the test function responds with only a Yes or No, solution is found or not.
- But in Hill Climbing the test function is augmented with a Heuristic function to estimate how close a given state is to a goal state.

Simple Hill Climbing Algorithm

1. Evaluate the initial state.
2. Loop until a solution is found or there are no new operators left to be applied:
    a) Select and apply a new operator.
    b) Evaluate the new state:
        (i) If it is a goal state then return it and quit.
        (ii) If better than current state then make it the current state.
        (iii) If not better than the current state then continue in the loop.

Hill climbing example (*minimizing* h)



Steepest- Ascent Hill Climbing

- Considering all the moves from the current state and selecting the best one as the next state might prove to be a useful variation on simple hill climbing.
  This method is called steepest-ascent hill climbing.

- Instead of moving to the first state that is better than the current state, move to the best possible state that is one move away.

Steepest-Ascent Hill Climbing Algorithm

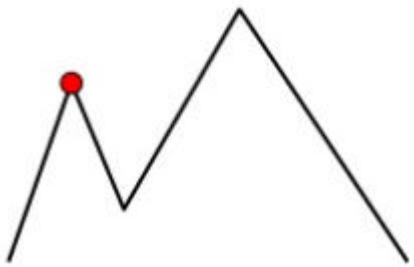
1. Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.
2. Loop until a solution is found or until a complete iteration produces no change to current state:
    (a) Let SUCC be a state such that any possible successor of the current state will be better than SUCC.
    (b) For each operator that applies to the current state do:
        i. Apply the operator and generate a new state.
        ii. Evaluate the new state. If it is a goal state, then return it and quit. If not, compare it to SUCC. If it is better, then set SUCC to this state. If it is not better, leave SUCC alone.
    (c) If the SUCC is better than current state, then set current state to SUCC.

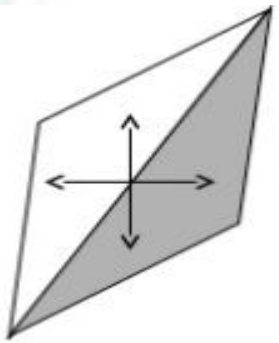
Hill Climbing: Disadvantages

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.



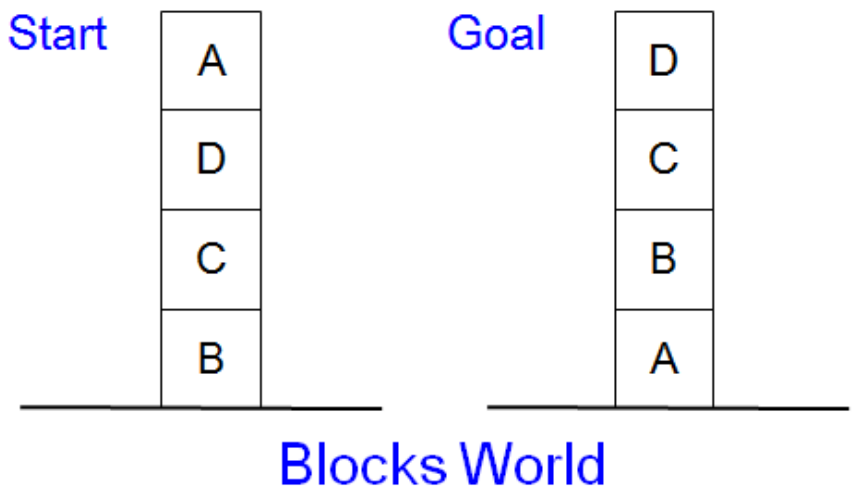


Some methods to deal with these problems

- 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)

Hill Climbing: Disadvantages

- Hill climbing is a local method:
  Decides what to do next by looking only at the "immediate" consequences of its choices.
- Global information might be encoded in heuristic functions.
- Consider a Block World Problem



Local heuristic:

+1 for each block that is resting on the thing it is supposed to be resting on.

-1 for each block that is resting on a wrong thing.

Initial State Score: 0

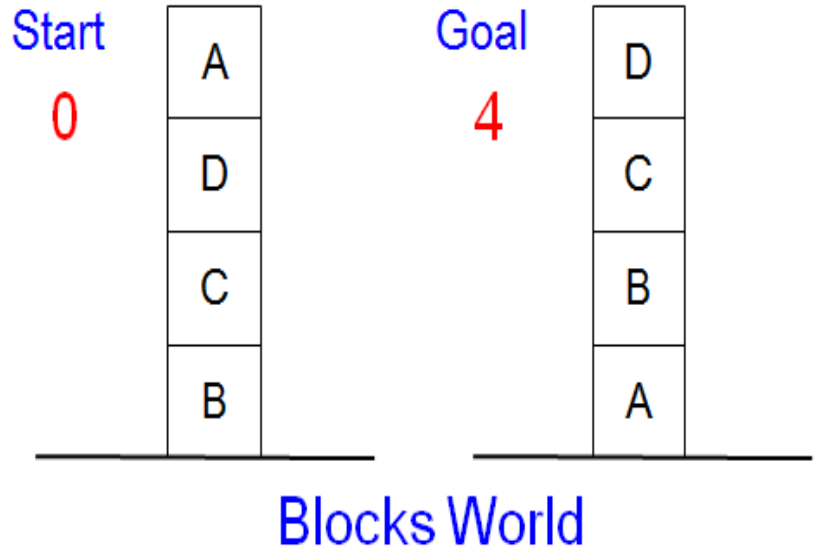
+1(for C & D) and -1(for B & A)

Total Score: 2-2=0

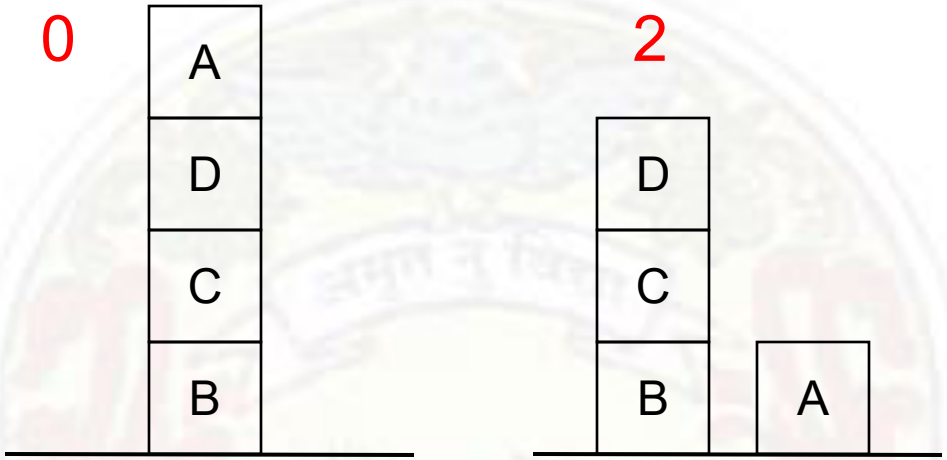
Goal State Score: 4

+1(for A,B,C & D)

Total Score: 4

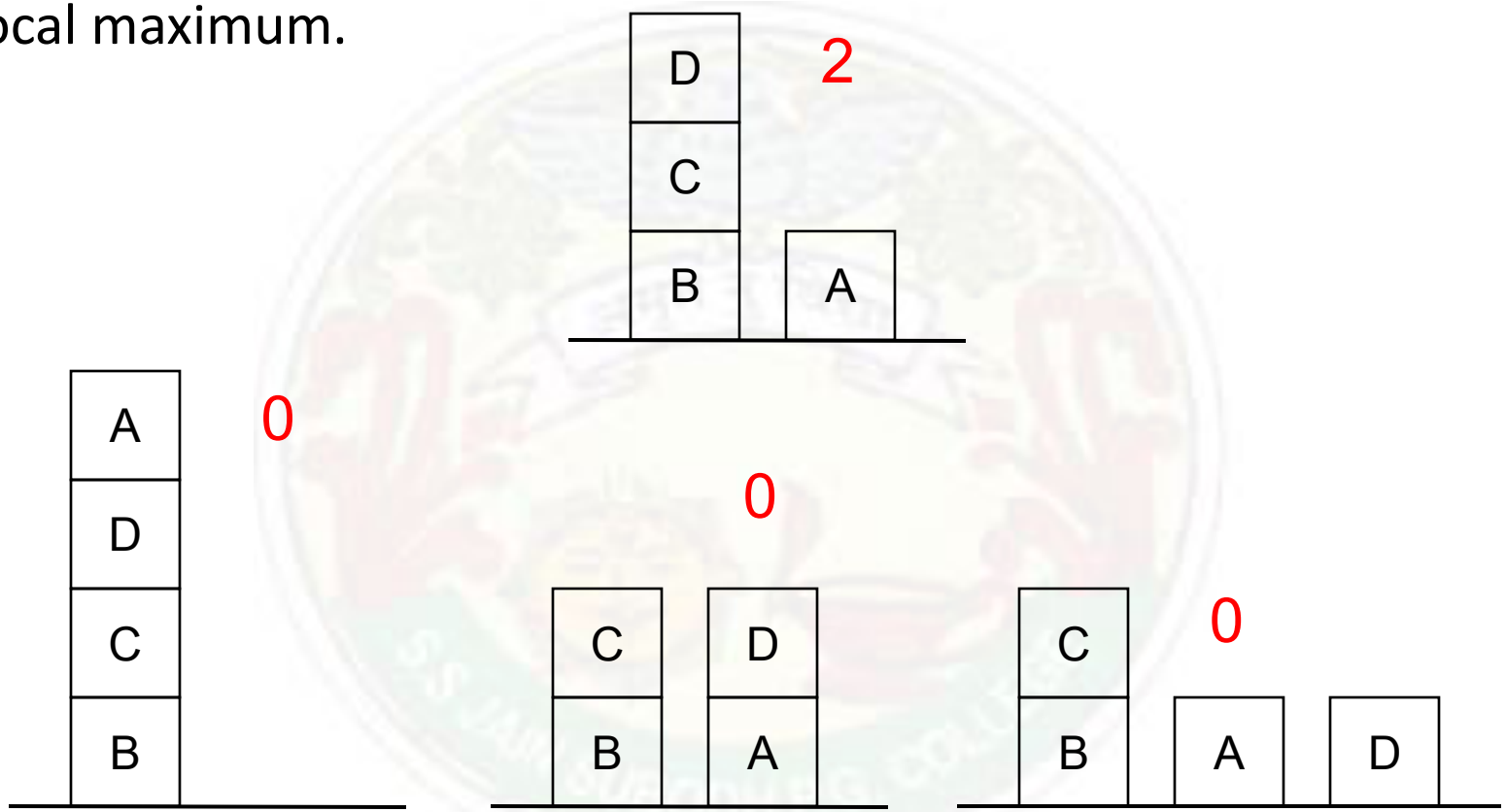


There is only one move from the Initial State i.e. move block A to the table.

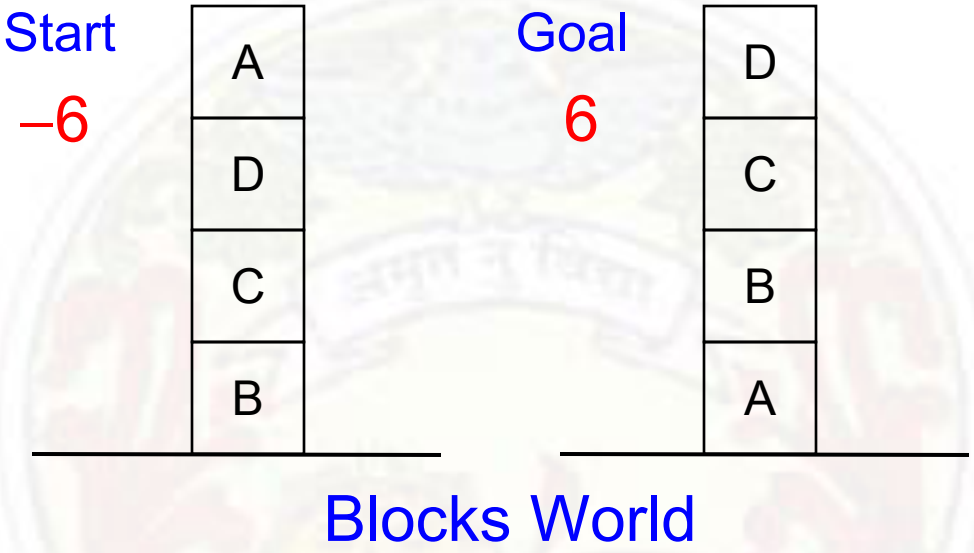


Now the New State has score 2 ,+1 for A,C & D while -1 for B Total Score(3-1=2).

From the new state there are 3 possible states. All these states have scores 0. Hill Climbing halt because all these states have lower score then the current state. The process has reached a local maximum.



We could blame Hill Climbing itself for the failure to look far enough ahead to find a solution. But we could also blame the heuristic function and try to modify it.



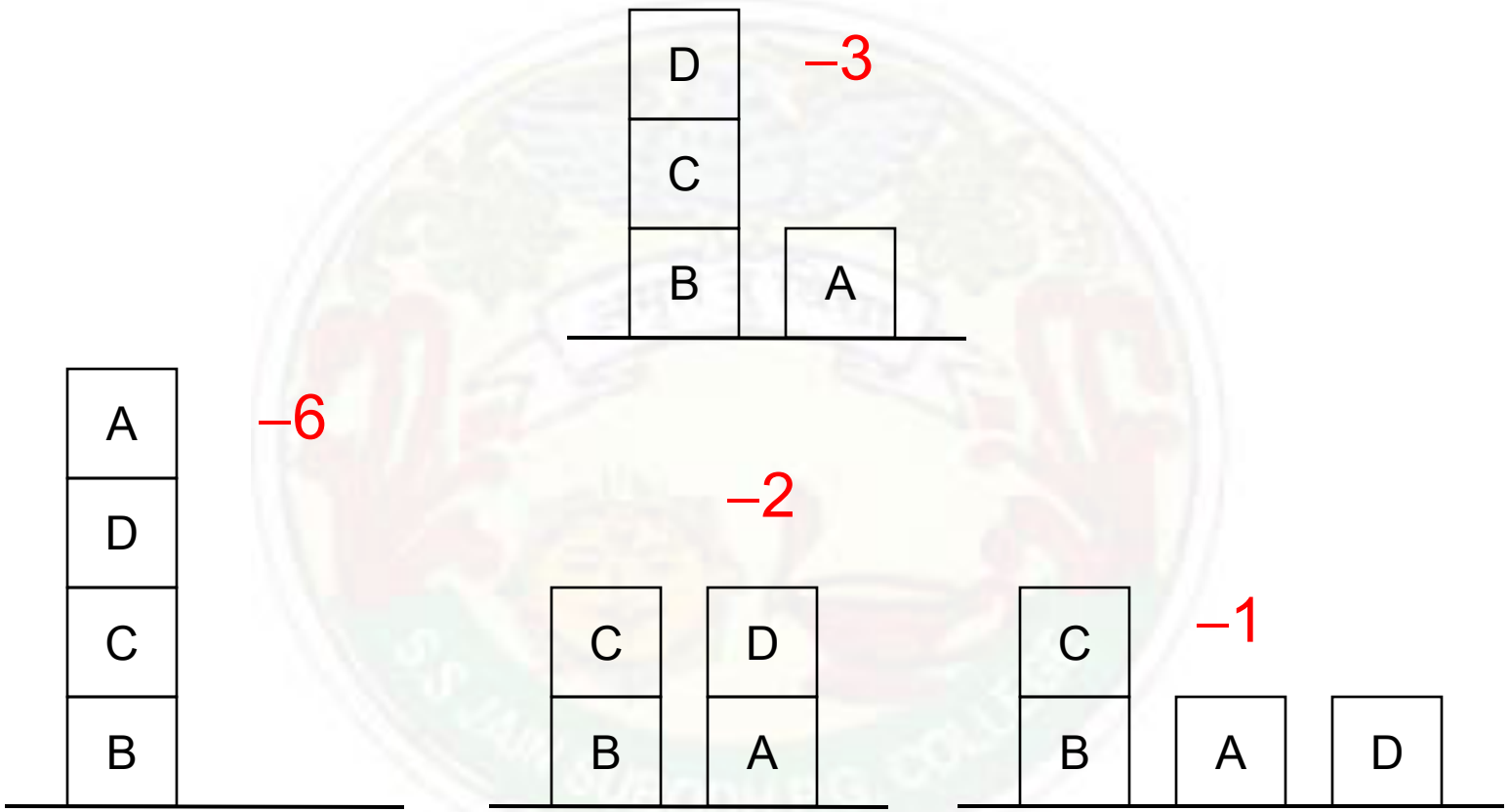
Blocks World

Global heuristic:

For each block that has the correct support structure: +1 to every block in the support structure.

For each block that has a wrong support structure: –1 to every block in the support structure.

Steepest-ascent hill climbing will choose the last move, which is the best one.



Hill Climbing: Conclusion

- Can be very inefficient in a large, rough problem space.
- Global heuristic may have to pay for computational complexity.
- Often useful when combined with other methods that get it started in the right general neighbourhood.

Thank You