INT404 ARTIFICIAL INTELLIGENCE

Heuristic search

Lecture 8

Heuristic search

- **Generate-and-test**
- **Hill climbing**

Algorithm : Generate-and-Test

1. Generate a possible solution. It means generate a point in the problem space or generate a path from start state.
2. Test to see if this is actually a solution by comparing the chosen point or the endpoint of the chosen path to the set of acceptable goal states.
3. If a solution has been found, quit. Otherwise, return to step 1.

GENERATE-AND-TEST

- Acceptable for simple problems.
  - Eg : 1. finding key of a 3 digit lock.
    2. 8-puzzle problem
- Inefficient for problems with large space.
- Use DFS as all possible solution generated, before they can be tested.

TYPES OF GENERATE-AND-TEST

- **Generate solution randomly:** British museum algorithm; wandering randomly.

- **Exhaustive** generate-and-test. : consider each case in depth

- **Heuristic** generate-and-test: not consider paths that seem unlikely to lead to a solution.

- **Plan** generate-test:

  – Create a list of candidates.

  – Apply generate-and-test to that list on the basis of constraint-satisfaction.

Ex – DENDERAL, which infers the structure of organic compounds using mass spectrogram and nuclear magnetic resonance (NMR) data.

HILL CLIMBING

- Generate-and-test + **direction to move** (feedback from test procedure).
- Test function + heuristic function = Hill Climbing
- **Heuristic function (objective function)** to estimate how close a given state is to a goal state.
- Hill climbing is often used when a good heuristic function is available for evaluating states but when no other useful knowledge is available.

SIMPLE HILL CLIMBING

- Evaluation function as a way to inject **task-specific knowledge** into the control process.
  - Key difference between Simple Hill climbing and Generate-and-test is the use of evaluation function as a way to inject task specific knowledge into the control process.
  - Better : higher value of heuristic function or Lower value

Algorithm : Simple Hill-Climbing

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 there are no new operators left to be applied in the current state:

(a) Select an operator that has not yet been applied to the current state and apply it to produce a new state.
(b) Evaluate the new state.
(i) If it is a goal state, then return it and quit.

(ii) If it is not a goal state but it is **better** than the current state, then make it the current state.

(iii) If it is not better than the current state, then continue in the loop.

Example

1) Use heuristic function as measure of how far off the number of tiles out of place.
2) Choose rule giving best increase in function.

STEEPEST-ASCENT HILL CLIMBING

- Considers **all the moves** from the current state.
- Selects **the best one** as the next state.
- Also known as **Gradient Search.**

Steepest-Ascent Hill Climbing or Gradient Search

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 **My-State** be a state such that any possible successor of the current state will be better than **My-State**.

(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 **My-State**. If it is better, then set **My-State** to this state. If it is not better, leave **My-State** alone.

(c) If the **My-State** is better than current state, then set current state to **My-State**.

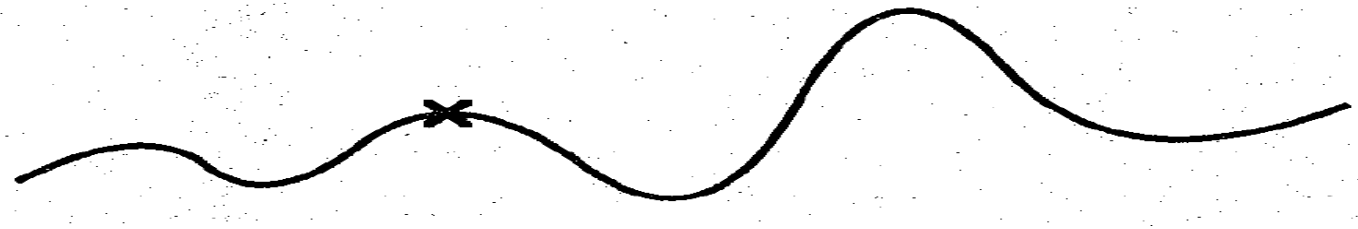
HILL CLIMBING: DISADVANTAGES

- Fail to find a solution
- Either Algorithm may terminate not by finding a goal state but by getting to a state from which no better state can be generated.

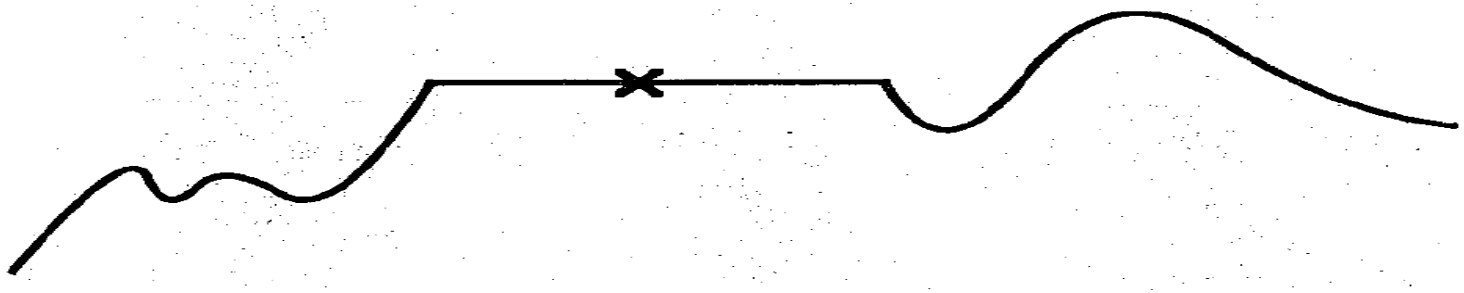
- This happen if program reached
  - **Local maximum:** A state that is better than all of its neighbours, but not better than some other states far away.
  - **Plateau:** A flat area of the search space in which all neighbouring states have the same value.
  - **Ridge:** Special kind of local maximum. The orientation of the high region, compared to the set of available moves, makes it impossible to climb up.

Hill-Climbing Dangers

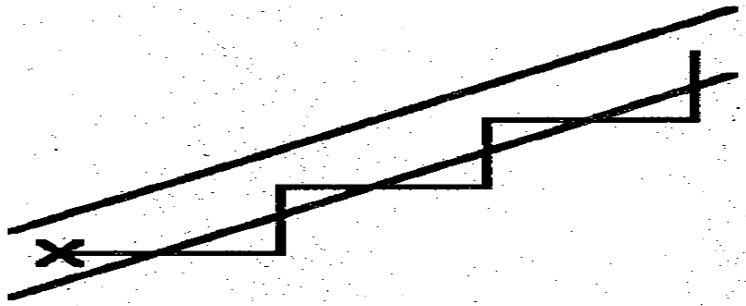
- **Local maximum**



- **Plateau**



- **Ridge**



HILL CLIMBING: DISADVANTAGES

Ways Out

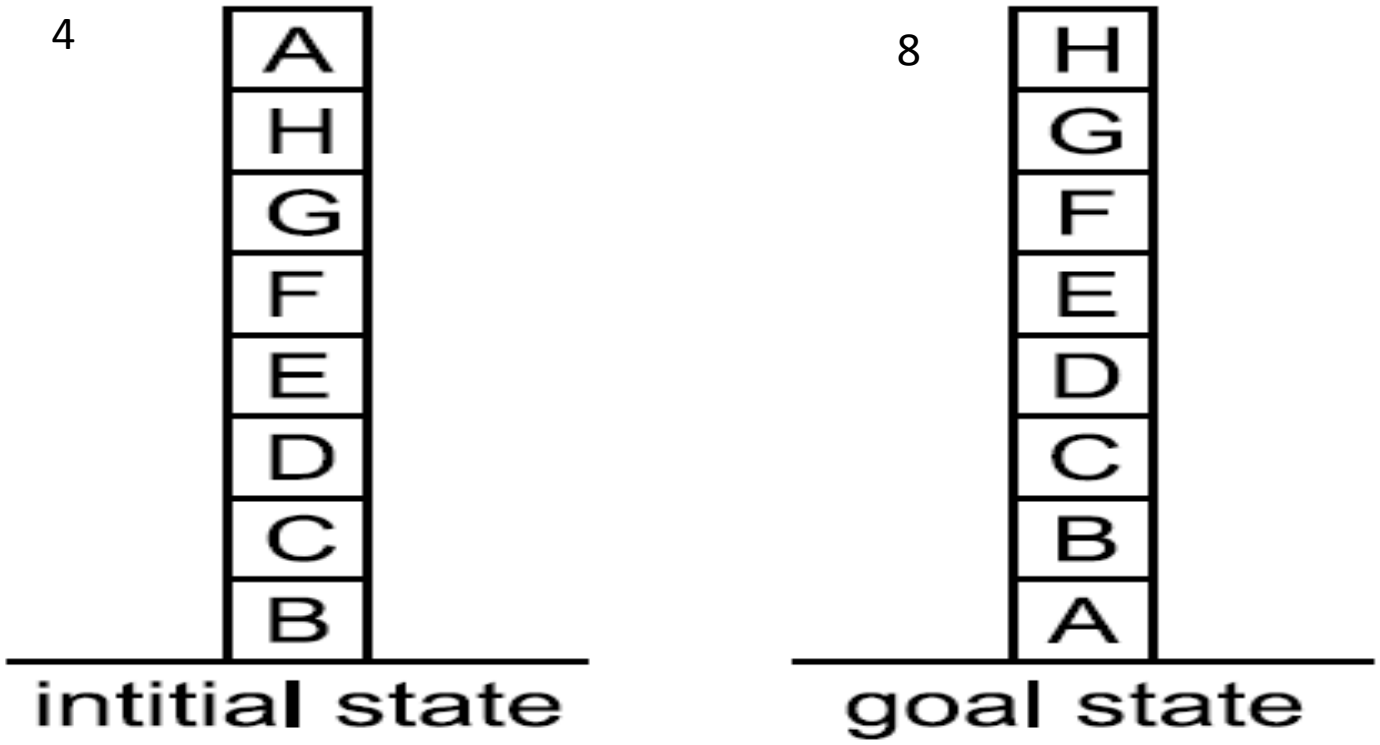
- Backtrack to some earlier node and try going in a different direction. (good way in dealing with local maxima)
- Make a **big jump** to try to get in a new section. (good way in dealing with plateaus)
- Moving in **several directions** at once. (good strategy for dealing with ridges)

HILL CLIMBING: DISADVANTAGES

- Hill climbing is a **local method:**
  Decides what to do next by looking only at the "immediate" consequences of its choices rather than by exhaustively exploring all the consequences.
- **Global information** might be encoded in heuristic functions.

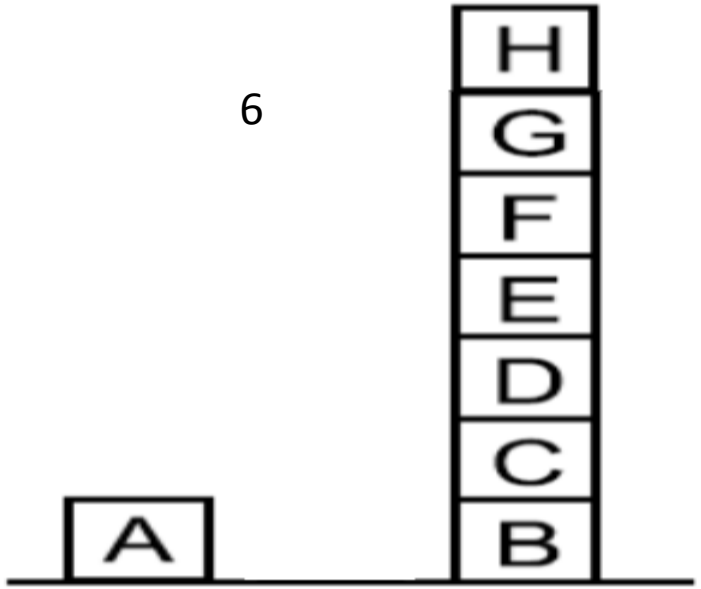
A Hill-Climbing Problem

Local: Add one point for every block that is resting on thing it is supposed to be resting on.
Subtract one point from every block that is sitting on wrong thing.

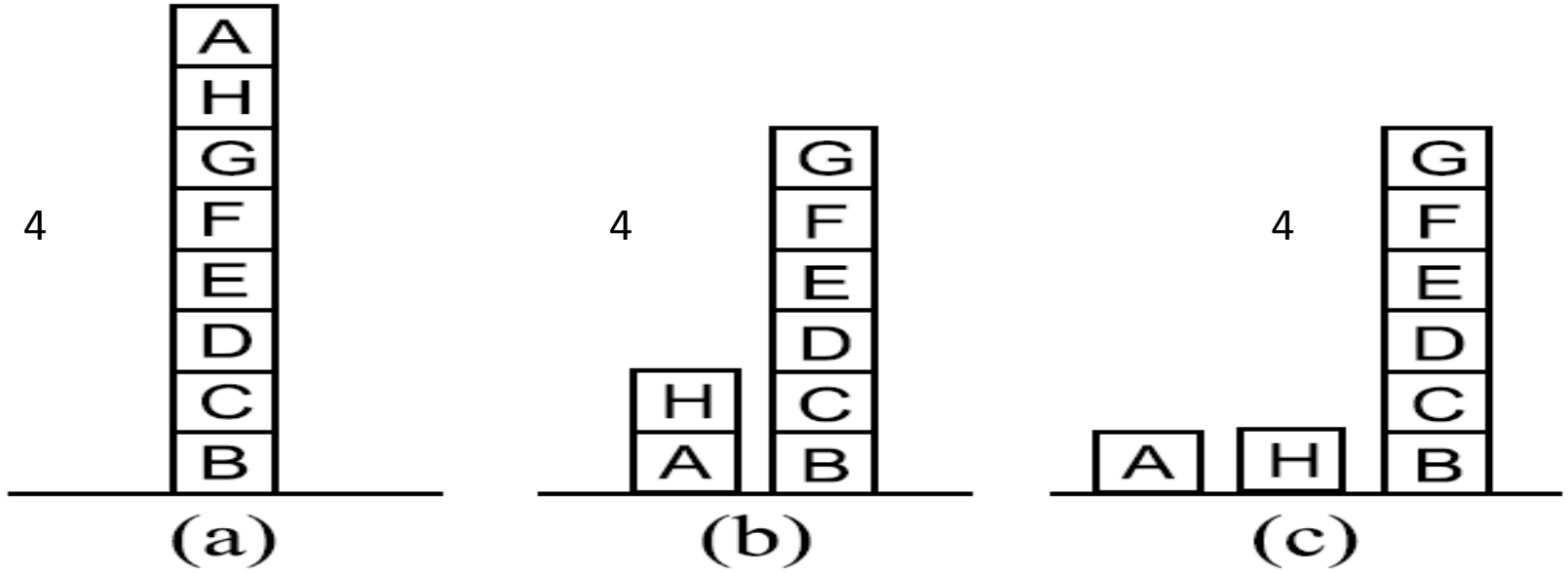


One Possible Moves

6



Three Possible Moves

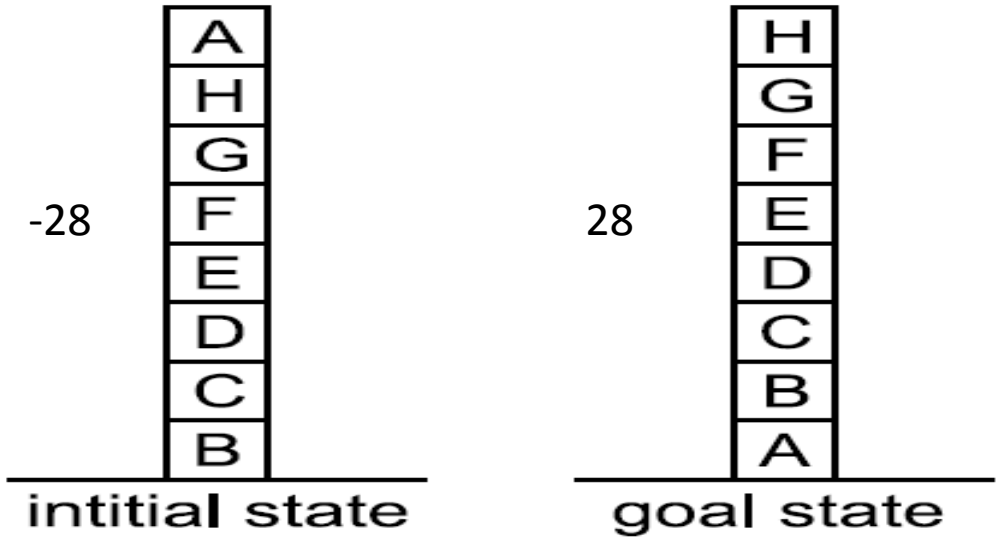


Hill Climbing will Halt because all states have lower score than the Current state.

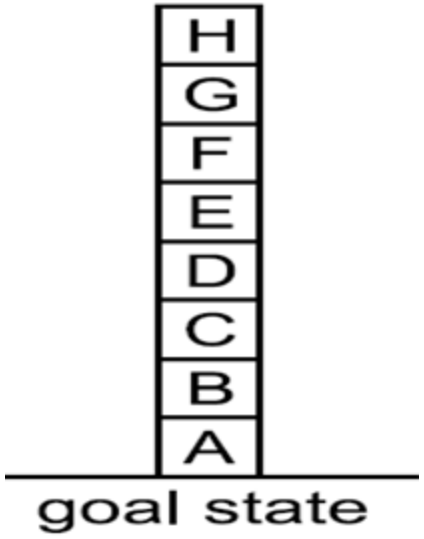
A Hill-Climbing Problem

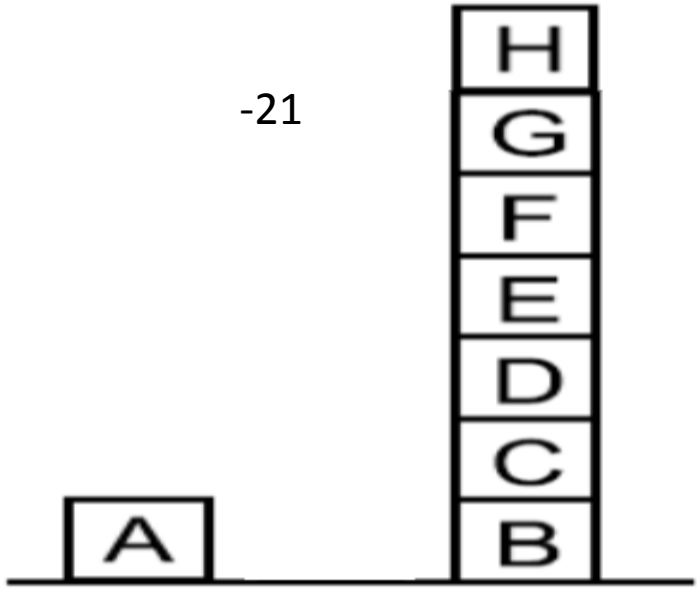
Global: For each block that has the correct support structure(i.e. the complete structure underneath it is exactly as it should be), add one point for every block in the support structure.

For each block that has an incorrect support structure, subtract one point for every block in the existing support structure.



One Possible Moves





Three Possible Moves