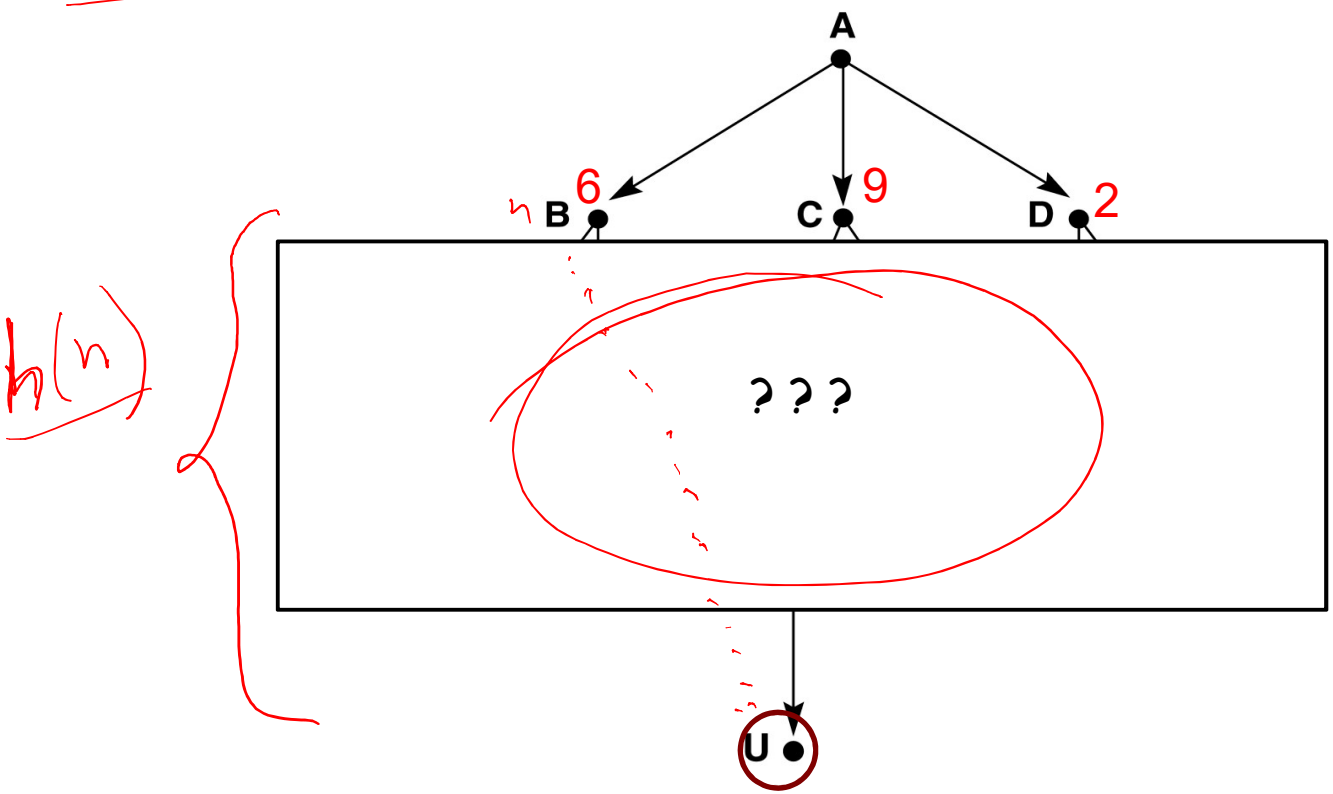
Artificial Intelligence: State Space Search part 3
Informed Search
Hill Climbing video #4

- Russell & Norvig – Sections 3.5.1, 3.5.2, 4.1.1

Today

1. State Space Representation
2. State Space Search
   a) Overview
   b) Uninformed search
      1. Breadth-first Search and Depth-first Search
      2. Depth-limited Search
      3. Iterative Deepening
      4. Uniform Cost
   c) Informed search
      1. Intro to Heuristics
      2. Hill Climbing
      3. Greedy Best-First Search
      4. Algorithms A & A*
      5. More on Heuristics
   d) Summary

h(n)



- **h(n)** = estimate of the lowest cost from *n* to *goal*

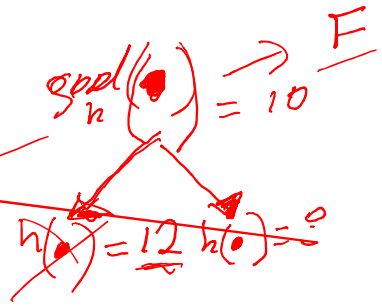
Hill Climbing

- General idea:
  - Similar to climbing a mountain in the fog with amnesia …
  - in the fog
    - --> only 1-step view of what is to come, so
    - if next step seems higher than where you are now -> go
    - otherwise, you assume you are at the top of the mountain -> stop
  - with amnesia -->
    - if you ever want to try other path, you can't because you did not keep track of where you came from

Vanilla HC vs Steepest Ascent HC

General Hill Climbing

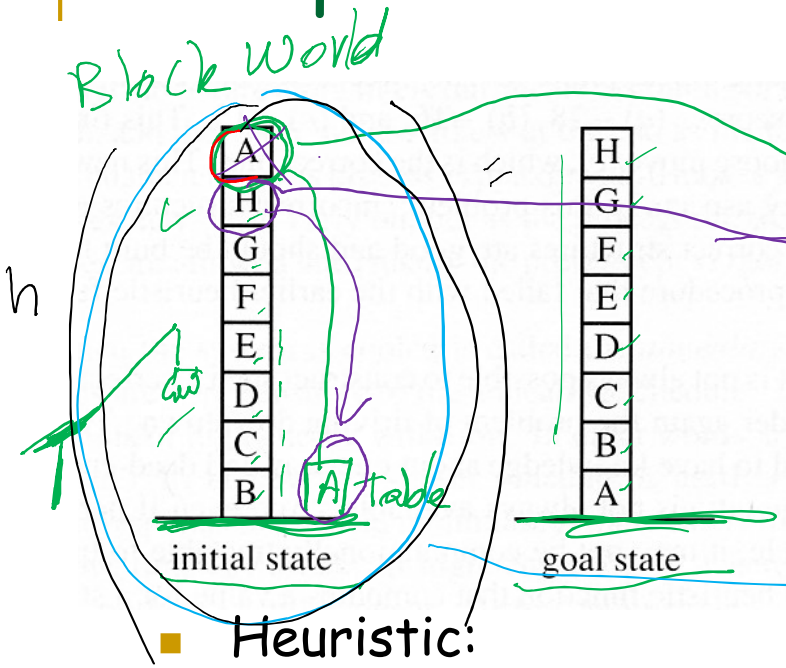
- uses h(n)
- does not use an open list (amnesia)



1. Vanilla Hill Climbing
   - take 1st successor s with better h() than current state n
   - i.e. if lower h(n) is better, chose 1st s with h(s) < h(n) // deep diving

2. Steepest ascent hill climbing:
   - generate all successor states S
   - run h() on all s ∈ S
   - among all successors s with better h() than current state n, take the successor s with the best h(n)

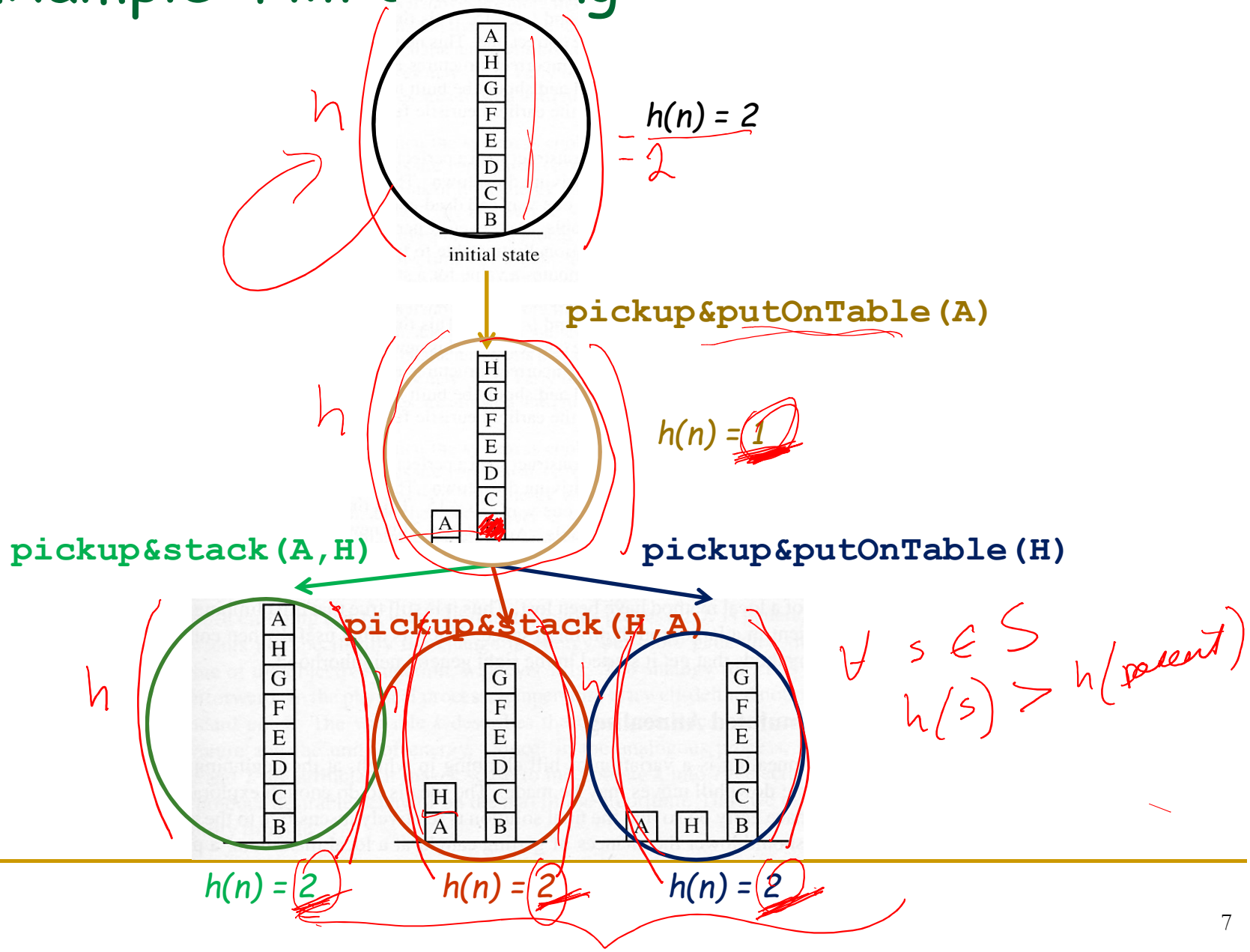
Example: Hill Climbing



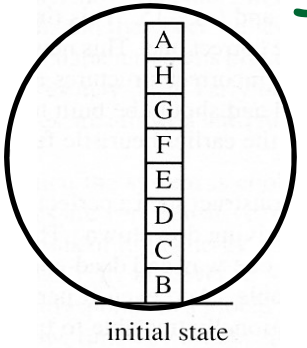
- Operators:
  - pickup&putOnTable(Block)
  - pickup&stack(Block1,Block2)
- Heuristic:
  - 0pt if a block is sitting where it is supposed to sit
  - +1pt if a block is NOT sitting where it is supposed to sit
  - so lower h(n) is better
    - h(initial) = 2
    - h(goal) = 0

source: Rich & Knight, Artificial Intelligence, McGraw-Hill College 1991.

Example: Hill Climbing

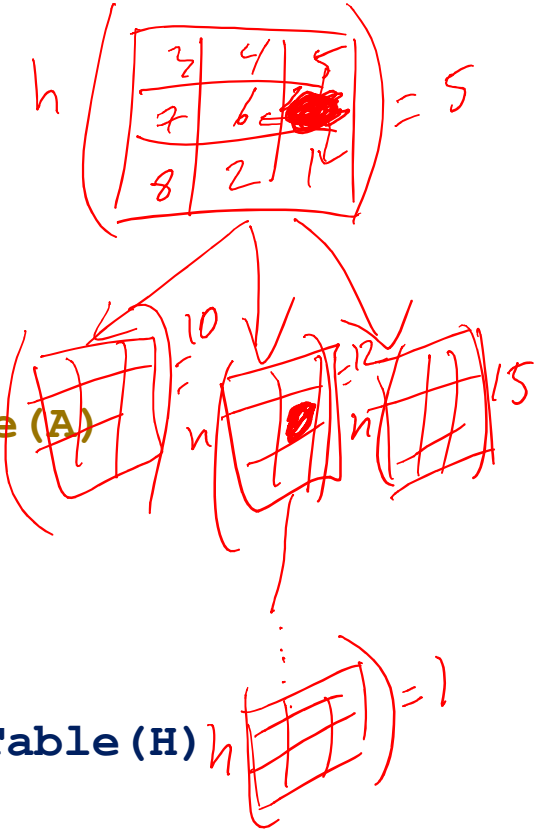


Example: Hill Climbing



h(n) = 2

pickup&putOnTable(A)



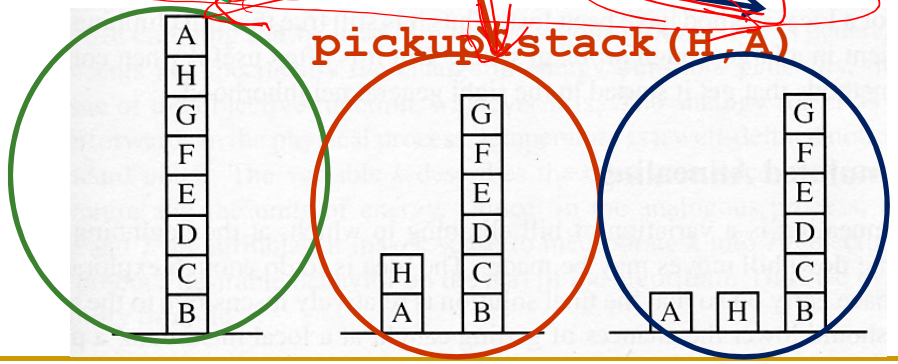
hill-climbing will stop, because all children have higher h(n) than the parent… --> local minimum

h(n) = 1

pickup&stack(A,H)

pickup&putOnTable(H)

pickup&stack(H,A)



h(n) = 2 h(n) = 2 h(n) = 2

Don't be confused… a lower h(n) is better…

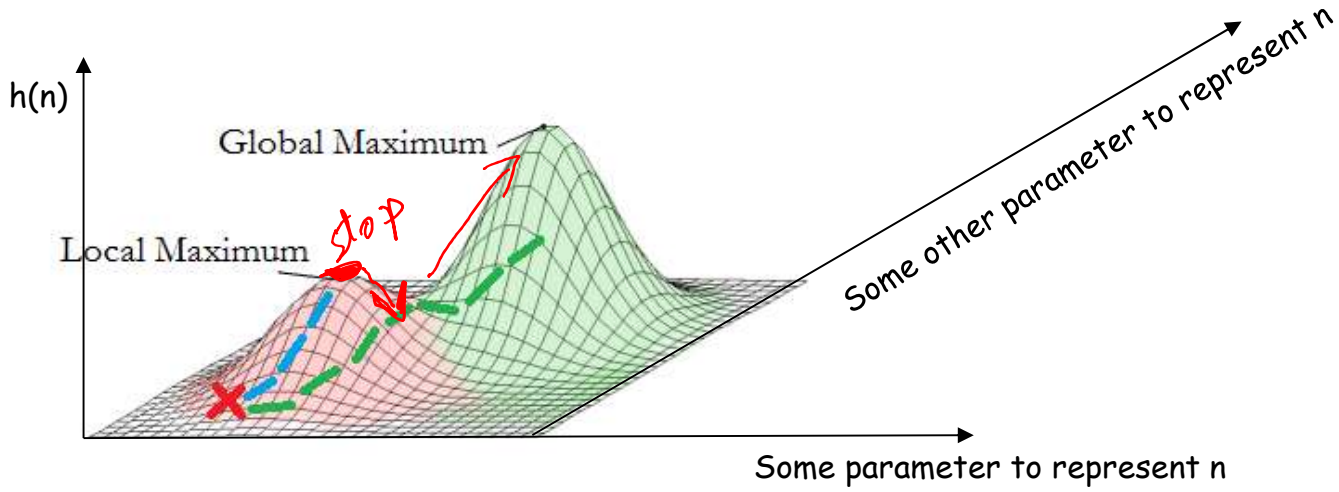
Steepest Ascent Hill Climbing

```
currentNode = startNode;
    loop do
        L = CHILDREN(currentNode);
        nextEval = +INFINITY;
        nextNode = NULL;

        for all c in L
           if (HEURISTIC-VALUE(c) < nextEval) // lower h is better
                nextNode = c;
                nextEval = HEURISTIC-VALUE(c);

          if nextEval >= HEURISTIC-VALUE(currentNode)
             // Return current node since no better child state exist
             return currentNode;

          currentNode = nextNode;
```

Adapted from https://en.wikipedia.org/wiki/Hill_climbing

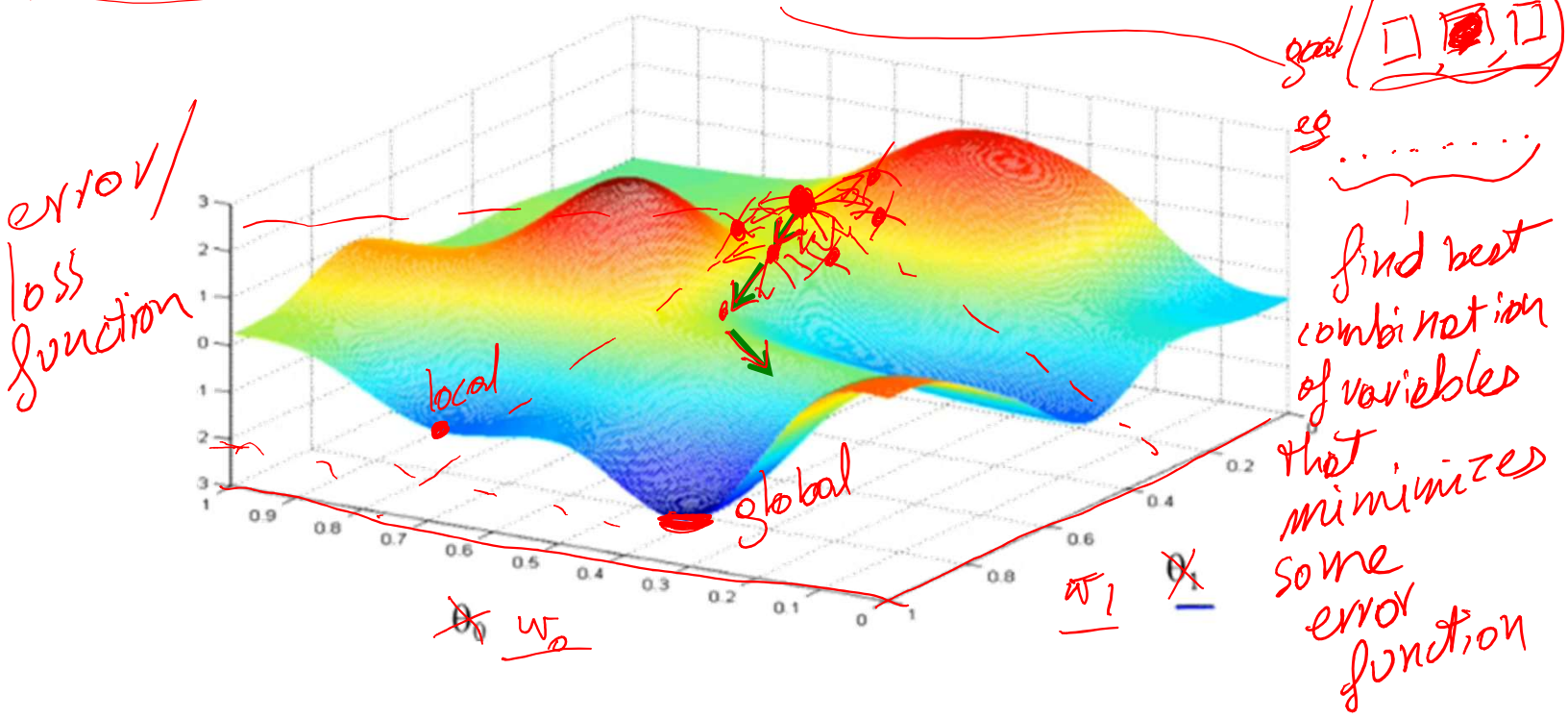
Problems with Hill Climbing

- Foothills (or local maxima)
  - reached a local maximum, not the global maximum
  - a state that is better than all its neighbors but is not better than some other states farther away.
  - at a local maximum, all moves appear to make things worse.
  - ex: 8-puzzle: we may need to move tiles temporarily out of goal position in order to place another tile in goal position



Use of Hill Climbing

- mostly for optimization problems
- i.e. goal defined not as a function of the state alone, but with respect of other states (eg. the best state I can reach)



Source: Andrew Ng

Today

1. State Space Representation ✓
2. State Space Search ✓
   a) Overview ✓
   b) Uninformed search ✓
      1. Breadth-first Search and Depth-first Search ✓
      2. Depth-limited Search ✓
      3. Iterative Deepening ✓
      4. Uniform Cost ✓
   c) Informed search ✓
      1. Intro to Heuristics ✓
      2. Hill climbing ✓
      3. Greedy Best-First Search
      4. Algorithms A & A*
      5. More on Heuristics
   d) Summary

h(n)

Problem with Hill-Climbing

- used mostly for optimization problems
  - where the goal state is defined with respect to other states
  - ex. shortest path, longest....
- if goal state is independent of other states
  - we should be able to backtrack, and find another path to the goal
  - i.e. we should use an OPEN list
  - i.e. Greedy Best First Search

Up Next

1. State Space Representation
2. State Space Search
   a) Overview
   b) Uninformed search
      1. Breadth-first and Depth-first
      2. Depth-limited Search
      3. Iterative Deepening
      4. Uniform Cost
   c) Informed search
      1. Intro to Heuristics
      2. Hill climbing
      3. Greedy Best-First Search
      4. Algorithms A & A*
      5. More on Heuristics
   d) Summary