Informed Search

Problem Solving by search

Prof.   Deepak C Vegda
email.:deepakvegda.it@ddu.ac.in

Informed search

- Uses certain available information for the searching process
- *like:* how far the goal is?
- It uses heuristic function to generate the information
- thus, known as heuristic search algorithm also.

Heuristic Search

- attempts to optimize a problem by improving the solution based on a given **heuristic** function or a cost measure.

Heuristic Function

- A **heuristic function** is a function that ranks alternatives in various search algorithms at each branching step based on the available information (heuristically) in order to make a decision about which branch to follow during a search.

- Well designed heuristic functions can play an important part in efficiently guiding a search process toward a solution. Sometimes very simple heuristic functions can provide a fairly good estimate of whether a path is any good or not. In other situations, more complex heuristic functions should be employed.

Heuristic Example : 8-puzzle

The first picture shows the current state and the second picture the goal state.
Heuristics → is the number of tiles out of place.
h(n) = 5
because the tiles 2, 8, 1, 6 and 7 are out of place.

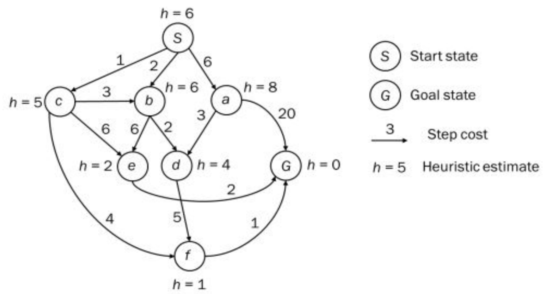
2	8	3
1	6	4
	7	5

Initial State

1	2	3
8		4
7	6	5

Goal state

Heuristic search



Heuristic Search Algorithms

- Algorithms that use a heuristic function are as follows
  - Generate and Test
  - Hill Climbing
  - Best First Search
  - A*
  - AO*

Hill Climbing

- This algorithm also called discrete optimization algorithm.
- It utilizes a simple heurisitic function.
- Hill Climbing = Depth First Search + Heuristic Function
- There is practically no difference between hill climbing and depth first search except that the childern of the node that has been exapanded are sorted by the remaining distance.

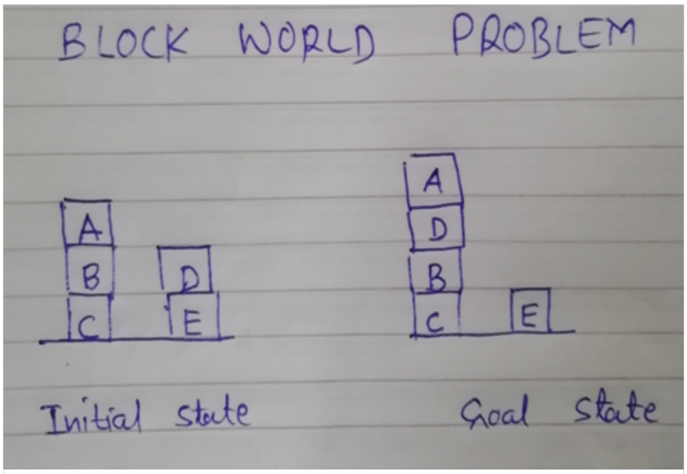
Implementation of Hill Climbing

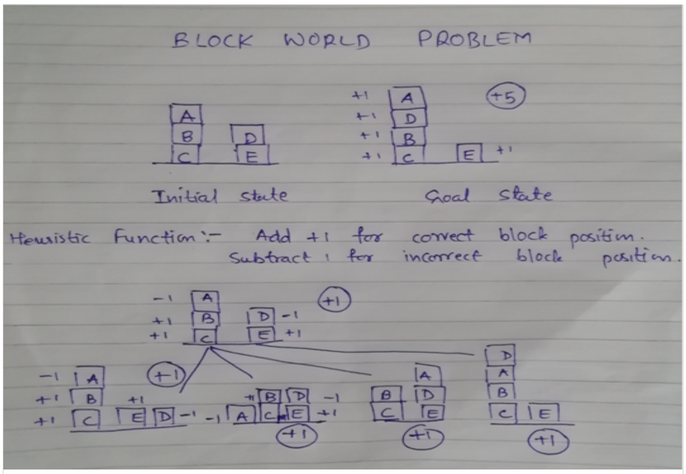
- There are two ways to implement hill climbing
  - Simple hill climbing
  - Steepest-Ascent hill climbing or gradient search

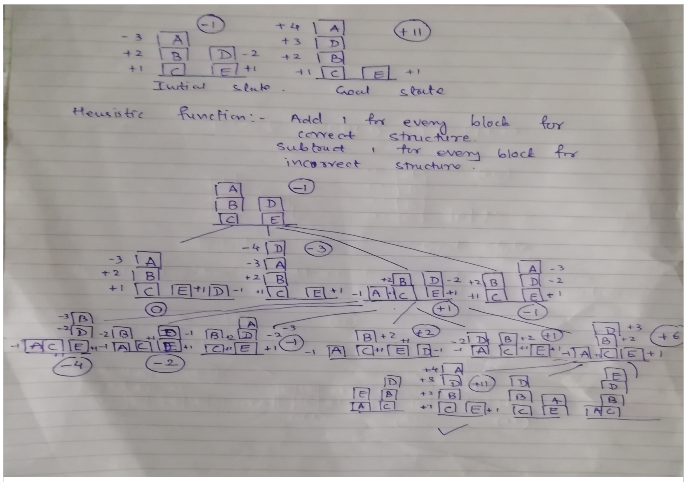
Some points about Hill climbing
-local search
-greedy approach
-No Backtrack

Simple hill climbing algorithm

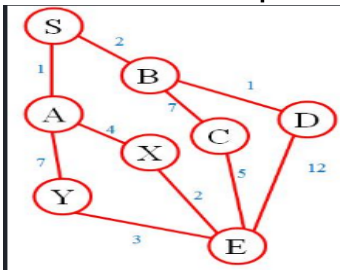
1. Evaluate the initial state if goal then return (success) . Else continue with initial state as the current state .
2. Loop until a solution is found or until there are no new operator to apply to current node :
   a) Select a new operator and apply current state to produce a new state .
   b) Evaluate the new state.
      i. if it is a goal then return (success) .
      ii. if not goal but better than current state then make it the current state .
      iii. if it is not better than current state then continue the loop.





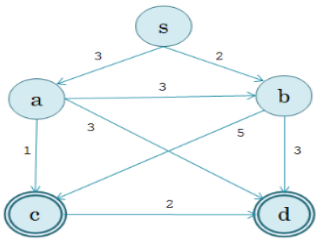


Example



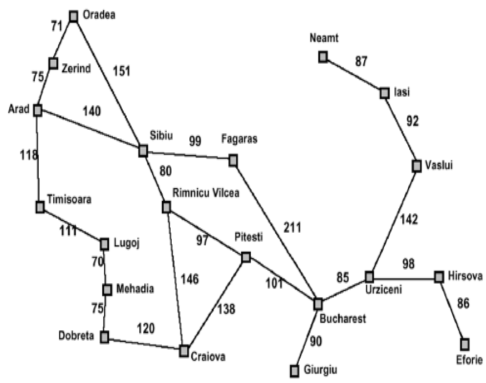
- Values for h:
  A:5, B:6, C:4, D:15, X:5, Y:8

Example



h(s)	h(a)	h(b)	h(c)	h(d)
1	3	3	0	0

Example



Straight−line distance to Bucharest

Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Steepest-Ascent 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 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 .

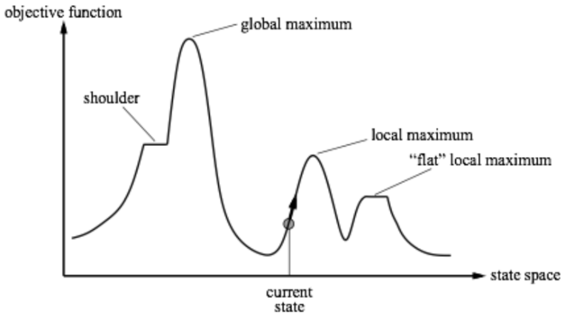
   ii) Evaluate the new state . if it is a goal state , return it and quit . If not, compare it to SUCC . If it is better , then set SUCC to this state . if it is not better , then leave SUCC alone .

   iii) if the SUCC is better than current state , then set current state to SUCC .

Difference between simple & steepest-ascent hill climbing

- Steepest-ascent hill climbing or gradient search considers all the moves from the current state and selects the best one as the next state.
- In the simple hill climbing the first state that is better than the current state is selected.

Why Hill climbing terminates without solutions?



Problems with Hill Climbing Technique

- Local Maximum : A state that is better than all its neighbors but not so when compared to states to states that are farther away.

Solution

- Utilize backtracking technique. Maintain a list of visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path

Problems with Hill Climbing Technique

- Plateau :A flat area of the search space in which all neighbouring states have the same value.

Solution

- Make a big jump. Randomly select a state far away from the current state. Chances are that we will land at a non-plateau region

Problems with Hill Climbing Technique

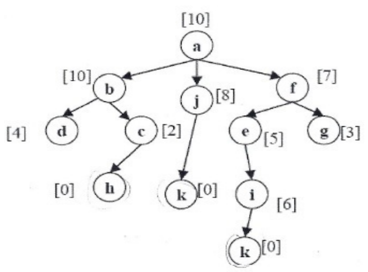
- Ridge :The orientation of the high region, compared to the set of available moves, makes it impossible to climb up. However, two moves executed serially may increase the height.

Solution

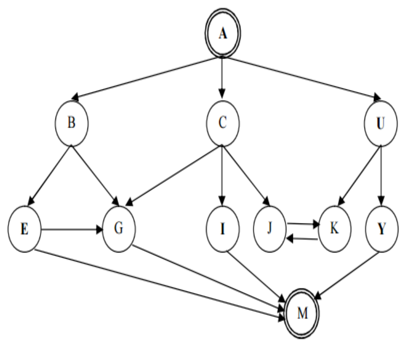
- In this kind of obstacle, use two or more rules before testing. It implies moving in several directions at once

Final Words on Hill-climbing

- Success of hill-climbing depends on the shape of the state space landscape.
- If there are few local maxima and plateaus, random-start hill climbing with sideways moves works well.
- However, for many real problems, the state space landscape is much more rugged.
- NP-complete problems are hard because they have exponential number of local maxima to get stuck on.
- In spite of all the problems, random-hill climbing with sideways moves works and other approximation techniques work reasonably well on such problems.



Solve it....



h(n):

h(A) = 7
h(B) = 5
h(C) = 3
h(U) = 4
h(E) = 2
h(G) = 3
h(I) = 6
h(J) = 2
h(K) = 1
h(Y) = 2
h(M) = 0

Solve it....

Initial State

1	2	3
8		4
7	6	5

Goal State

2	8	1
	4	3
7	6	5

Solve it....