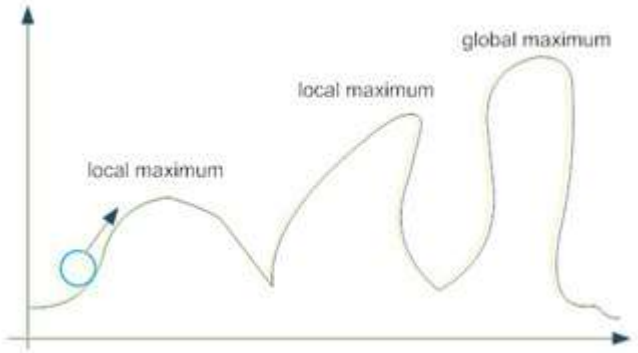
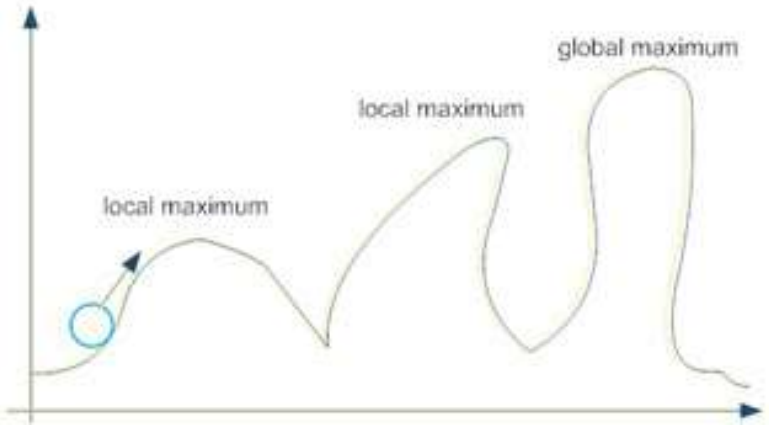
Hill climbing

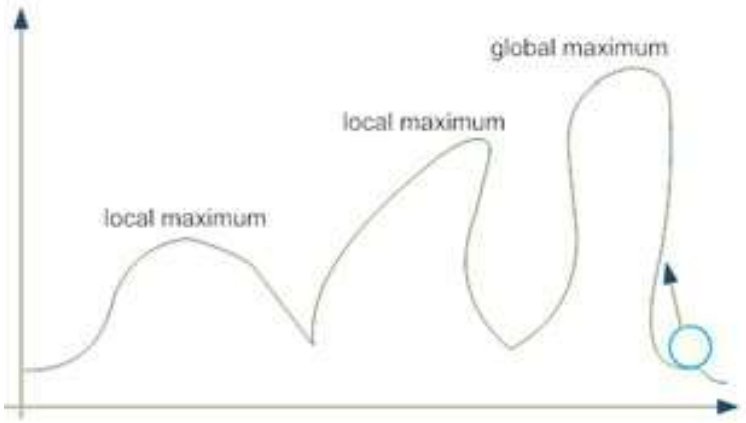
- the idea is to keep improving current state, and stop when we can’t improve any more.



- Hill-climbing is analogous to the way one might set out to climb a mountain in the absence of a map: always move in the direction of increasing altitude

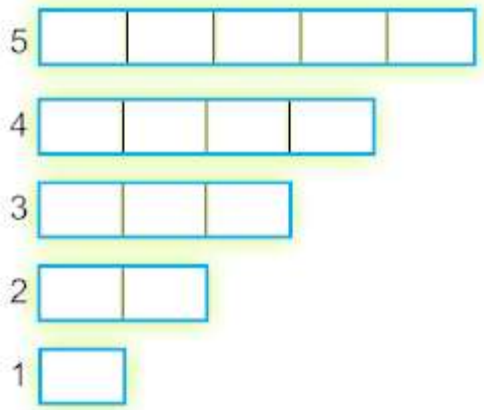
Hill climbing contd.



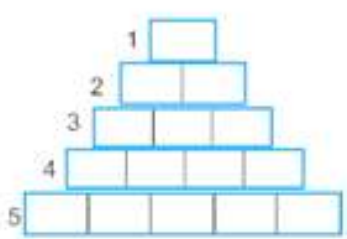


Hill climbing example 1

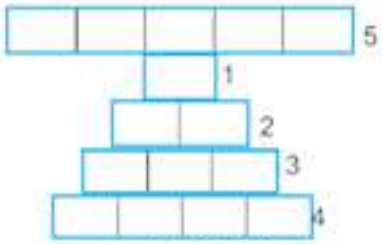
- Heuristic:



- count +1 for every figure that sits on the correct figure. The goal state has the value +5;
- count -1 for every figure that sits on an incorrect figure.

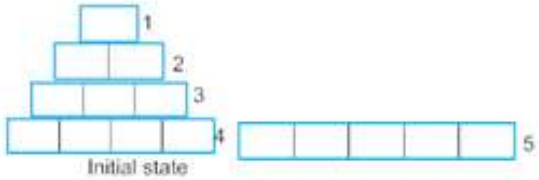


Goal state



Initial state

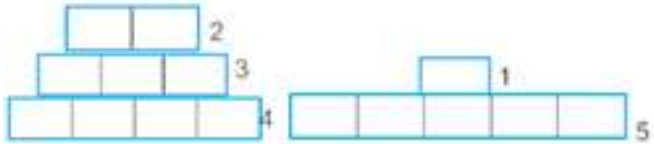
MOVE 1



Initial state

MOVE 2a

MOVE 2b

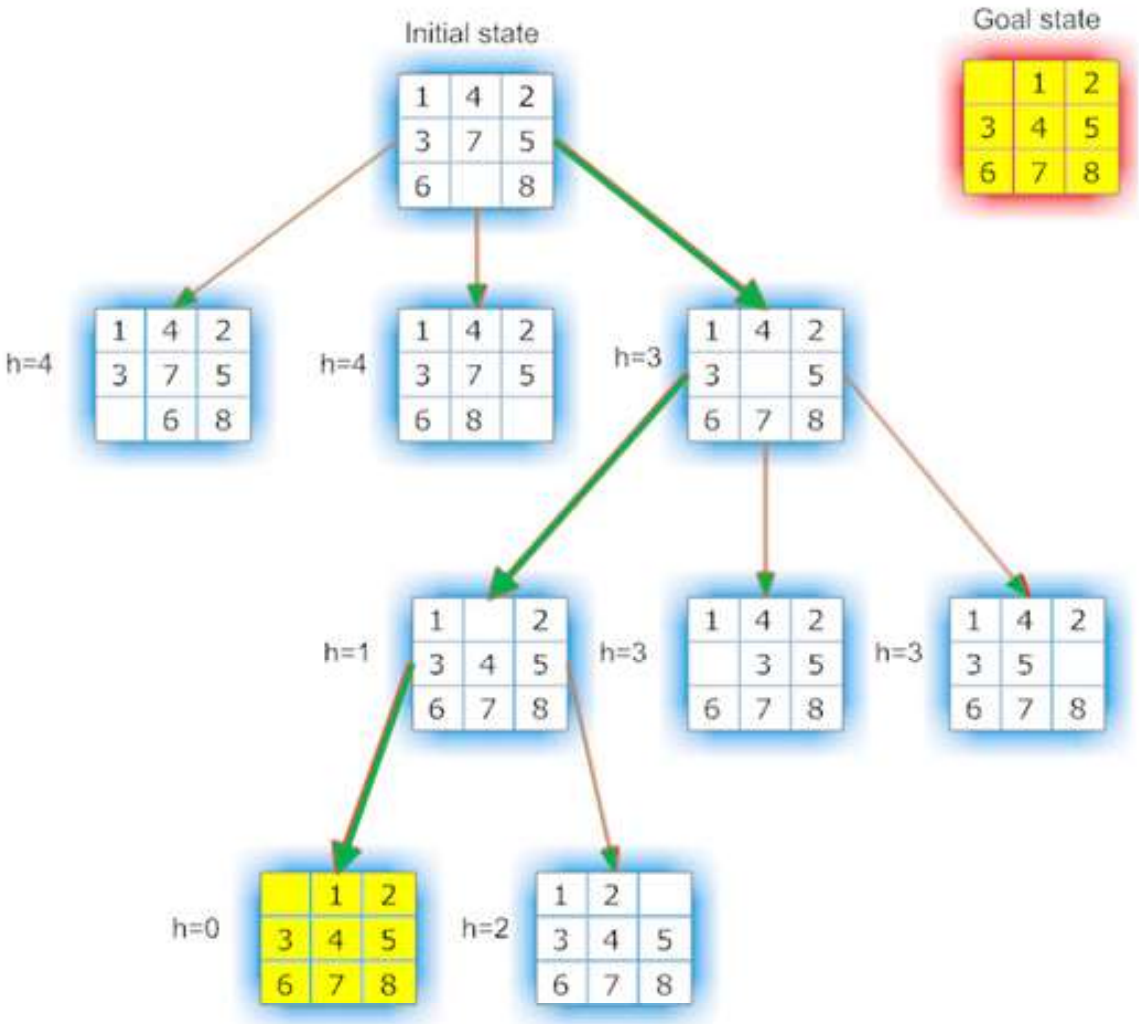


Another heuristic:

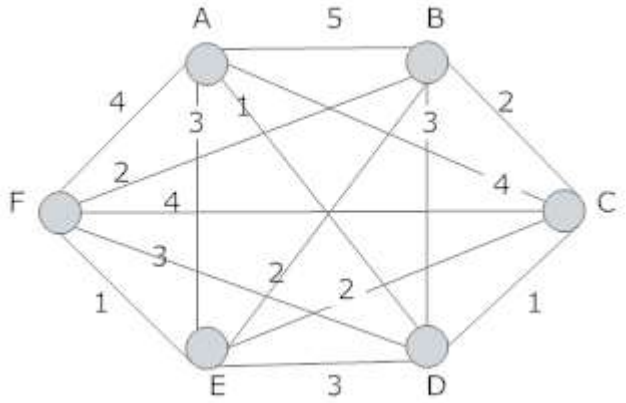
- count +n for every piece that sits on a correct stack of n pieces. The goal state has the value +10.

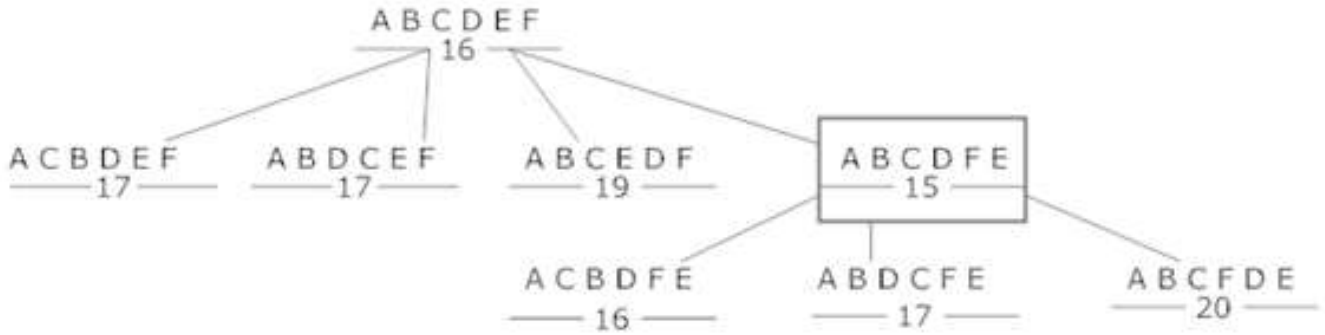
- count -n for every block that sits on an incorrect stack of n.

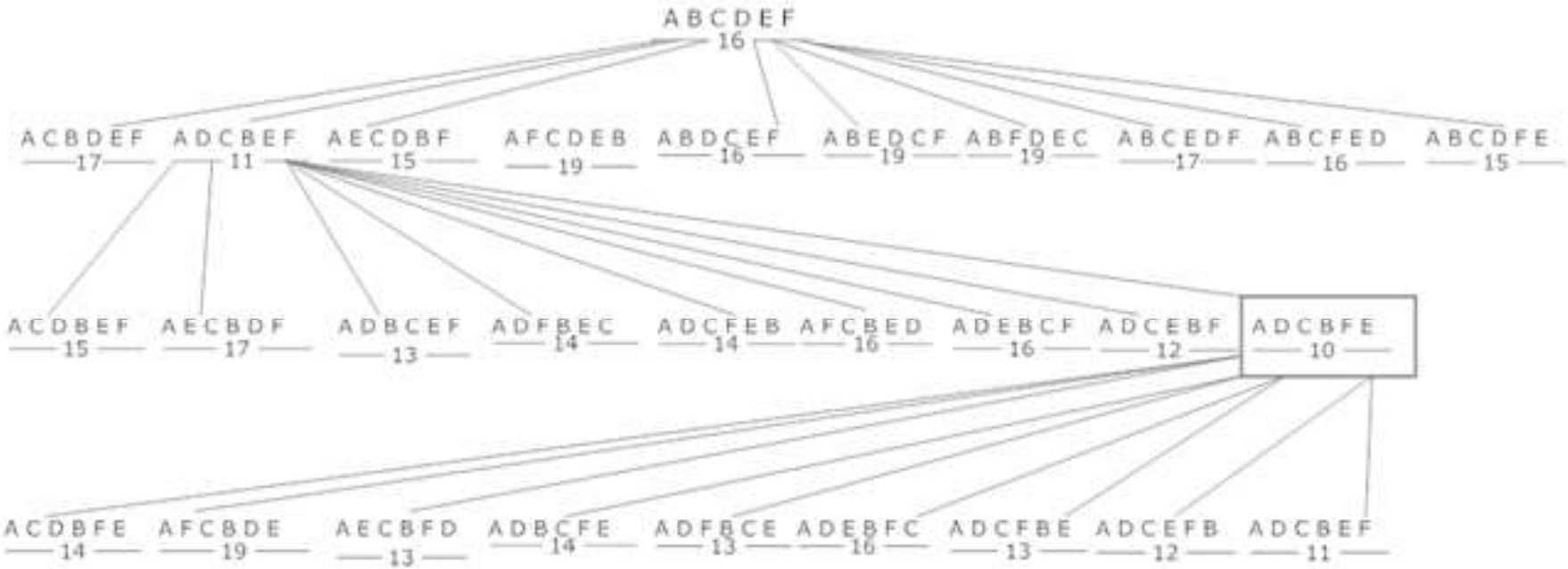
Hill climbing Example 2



Hill climbing Example 3






ABCDEF
16
ACBDEF
17
ADCBEF
11
AECDBF
15
AFCDEB
19
ABDCEF
16
ABEDCF
19
ABFDEC
19
ABCEDF
17
ABCFED
16
ABCDFE
15
ACDBEF
15
AECBDF
17
ADBCEF
13
ADFBEC
14
ADCFEB
14
AFCBED
16
ADEBCF
16
ADCEBF
12
ADCBFE
10
ACDBFE
14
AFCBDE
19
AECBFD
13
ADBCFE
14
ADFBCE
13
ADEBFC
16
ADCFBE
13
ADCEFB
12
ADCBEF
11

Hill climbing: Advantages and Disadvantages

Advantages:

- easy to implement; the algorithm is very simple and easy to reproduce;
- Requires no memory (since there is no backtracking);
- Since it is very simple, can be easily used to get an approximate solution when the exact one is hard or almost impossible to find (for instance, for very large TSP instances (or other similar NP_Complete problems), an approximate solution may be satisfactory when the exact one is not known and difficult to find).

Disadvantages:

- the evaluation function may be sometimes difficult to design;
- If the number of moves is enormous, the algorithm may be inefficient.
- By contrary, if the number of moves is less, the algorithm can get stuck easily.
- It's often cheaper to evaluate an incremental change of a previously evaluated object than to evaluate from scratch.
- Inner-loop optimization often possible.