#EngineeringPlus Online Course Series

Artificial Intelligence

Module: 01
Lecture:

Course Instructor

Roneeta Purkayastha
Assistant Professor
Computer Science and Engineering
roneeta.purkayastha@adamasuniversity.ac.in

Index

- Informed search vs uninformed search
- Introduction to Informed Search
- Heuristics
- Hill Climbing
- Example of heuristic function
- Best first search
- Greedy best first search
- Properties of greedy best first search

Uninformed Search

- They don’t have any additional information.
- The information is only provided in the problem definition.
- The goal state can be reached using different order and length of actions.
- Examples of uninformed search include depth first search (DFS) and breadth first search (BFS).
- It doesn’t use the knowledge in the process of searching.
- It takes more time to show the solution.
- It is always complete.
- It is expensive.
- It consumes moderate time.
- There is no suggestion regarding finding the solution.
- It is lengthy to implement.

Informed Search

- They contain information on goal state.
- It helps search efficiently.
- The information is obtained by a function that helps estimate how close a current state is, to the goal state.
- Examples of informed search include greedy search and graph search.
- It uses the knowledge in the process of searching.
- It helps find the solution quickly.
- It may or may not be complete.
- It is inexpensive.
- It consumes less time.
- It gives the direction about the solution.
- It is less lengthy to implement

Introduction to Informed Search

- Uninformed search methods systematically explore the state space and find the goal.
- They are inefficient in most cases.
- Informed search methods use problem specific knowledge, and may be more efficient.
- It is possible due to a heuristic function associated with the informed search.

Heuristics

- Heuristic means “rule of thumb”.
- To quote Judea Pearl, “Heuristics are criteria, methods or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal”.
- In heuristic search or informed search, heuristics are used to identify the most promising search path.

Heuristic search techniques

- Generate-and-test
- Hill climbing
- Best first search
- Problem reduction
- Constraint satisfaction
- Means-ends analysis

Generate-and-test

- Simplest of all approaches
- Algorithm:
    1. Generate a possible solution.
    2. Test to see if this is actually a solution by comparing the chosen point to the set of acceptable goal states.
    3. If a solution has been found, quit.

    Otherwise, return to step 1.
- It is reasonable for simple problems.
- Not effective for harder problems.

Hill Climbing Algorithm

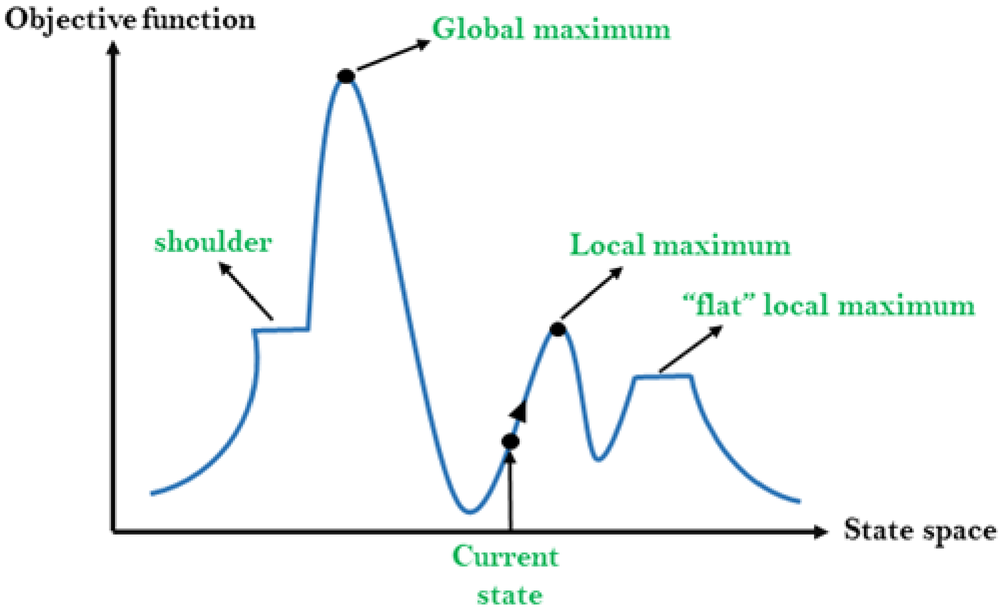
ADAMAS SCHOOL OF ENGINEERING AND TECHNOLOGY

- Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem. It terminates when it reaches a peak value where no neighbor has a higher value.
- Hill climbing algorithm is a technique which is used for optimizing the mathematical problems.
- One of the widely discussed examples of Hill climbing algorithm is Traveling-salesman Problem in which we need to minimize the distance traveled by the salesman.

Features of Hill Climbing

- **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
- **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the cost.
- **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

State-space Diagram for Hill Climbing



Different regions in the state space landscape

ADAMAS SCHOOL OF ENGINEERING AND TECHNOLOGY

- **Local Maximum:** Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.
- **Global Maximum:** Global maximum is the best possible state of state space landscape. It has the highest value of objective function.
- **Current state:** It is a state in a landscape diagram where an agent is currently present.
- **Flat local maximum:** It is a flat space in the landscape where all the neighbor states of current states have the same value.
- **Shoulder:** It is a plateau region which has an uphill edge.

Types of Hill Climbing Algorithm

- Simple hill Climbing
- Steepest-Ascent hill-climbing
- Stochastic hill Climbing

Simple hill Climbing

- Simple hill climbing is the simplest way to implement a hill climbing algorithm.
- **It only evaluates the neighbor node state at a time and selects the first one which optimizes current cost and set it as a current state**.
- Less time consuming
- Less optimal solution and the solution is not guaranteed

Algorithm for Simple Hill Climbing

- **Step 1:** Evaluate the initial state, if it is goal state then return success and Stop.
- **Step 2:** Loop Until a solution is found or there is no new operator left to apply.
- **Step 3:** Select and apply an operator to the current state.
- **Step 4:** Check new state:
  - If it is goal state, then return success and quit.
  - Else if it is better than the current state then assign new state as a current state.
  - Else if not better than the current state, then return to step2.
- **Step 5:** Exit.

Steepest-Ascent hill climbing

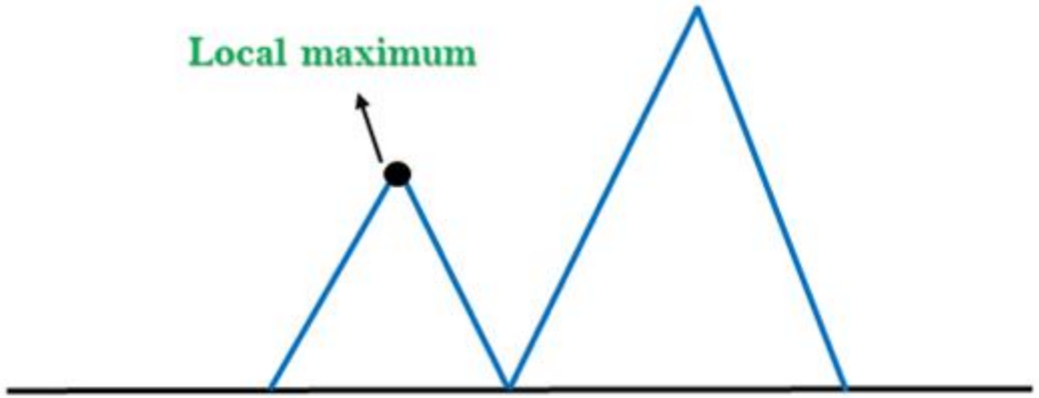
- The steepest-Ascent algorithm is a variation of simple hill climbing algorithm.
- This algorithm examines all the neighboring nodes of the current state and selects one neighbor node which is closest to the goal state.
- This algorithm consumes more time as it searches for multiple neighbors.

Algorithm for Steepest-Ascent hill climbing

- **Step 1:** Evaluate the initial state, if it is goal state then return success and stop, else make current state as initial state.
- **Step 2:** Loop until a solution is found or the current state does not change.
  - Let SUCC be a state such that any successor of the current state will be better than it.
  - For each operator that applies to the current state:
    - Apply the new operator and generate a new state.
    - Evaluate the new state.
    - If it is goal state, then return it and quit, else compare it to the SUCC.
    - If it is better than SUCC, then set new state as SUCC.
    - If the SUCC is better than the current state, then set current state to SUCC.
- **Step 5:** Exit.

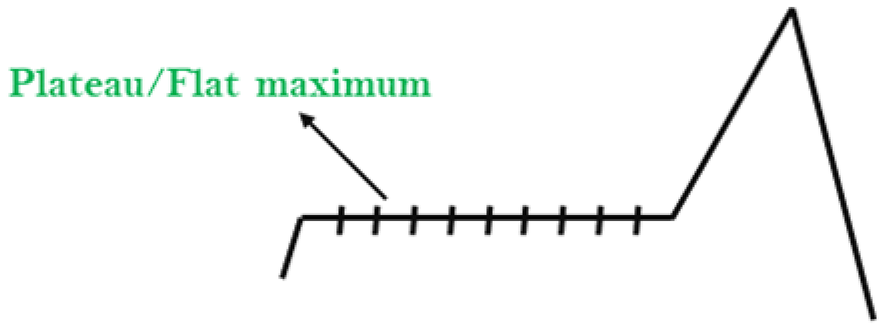
Problems in Hill Climbing Algorithm

- **Local Maximum**



Problems in Hill Climbing Algorithm

- **Plateau**



Problems in Hill Climbing Algorithm

- **Ridges**

Example of Heuristic Function

- **A heuristic function at a node n is an estimate of the optimum cost from the current node to a goal.**
- It is denoted by **h(n).**
- **h(n) = estimated cost of the cheapest path from node n to a goal node**

Example of Heuristic Function

- **Example 1**: We want a path from Kolkata to Pune
- Heuristic for Pune may be straight-line distance between Kolkata and Pune
- h(Kolkata) = euclideanDistance(Kolkata, Pune)

Example of Heuristic Function

- Example 2: **8-puzzle**: **Misplaced Tiles**
- Heuristics is the number of tiles out of place.

2	8	3
1	6	4
	7	5

Initial State

1	2	3
8		4
7	6	5

Goal state

- h(n) = 5 because the tiles 2, 8, 1, 6 and 7 are out of place.
- **Manhattan Distance Heuristic**: This heuristic sums the distance that the tiles are out of place.
- The distance of a tile is measured by the sum of the differences in the x-positions and the y-positions.
- Here, using the Manhattan distance heuristic, h(n) = 1 + 1 + 0 + 0 + 0 + 1 + 1 + 2 = 6

Best First Search

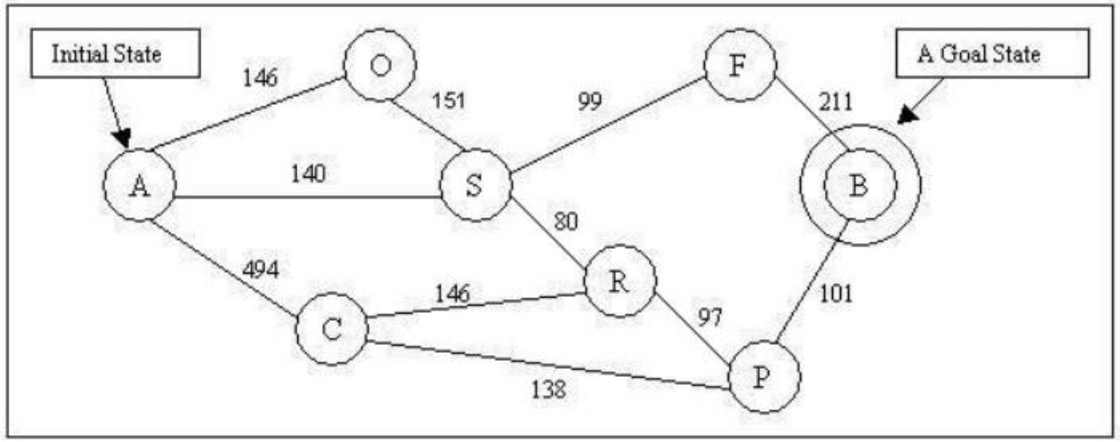
- Uniform Cost Search is a special case of the best first search algorithm.
- The generic best first search algorithm is outlined below.

```
Best First Search
Let fringe be a priority queue containing the initial state
Loop
      if fringe is empty return failure
      Node ← remove-first (fringe)
                  if Node is a goal
                        then return the path from initial state to Node
          else generate all successors of Node, and
             put the newly generated nodes into fringe
             according to their f values
End Loop
```

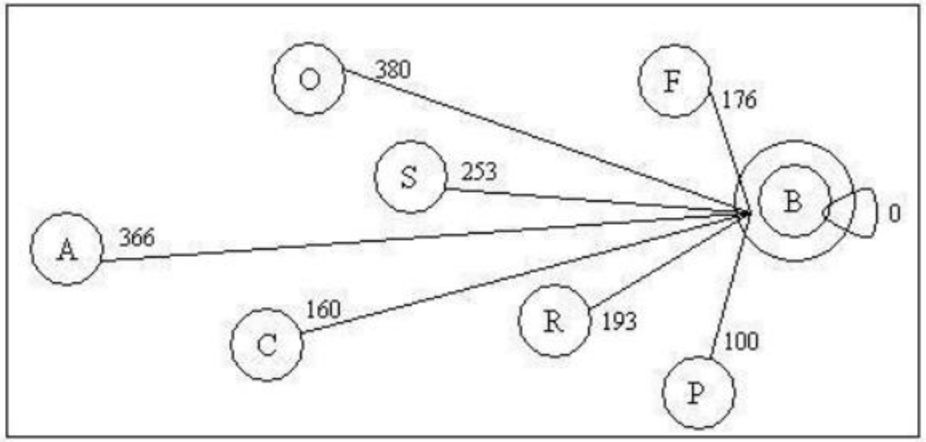
Greedy best first search

- In greedy search, the idea is to expand the node with the smallest estimated cost to reach the goal.
- We use a heuristic function $f(n) = h(n)$
- $h(n)$ estimates the distance remaining to a goal.
- Greedy algorithms often perform very well.
- They tend to find good solutions quickly, although not always optimal ones.

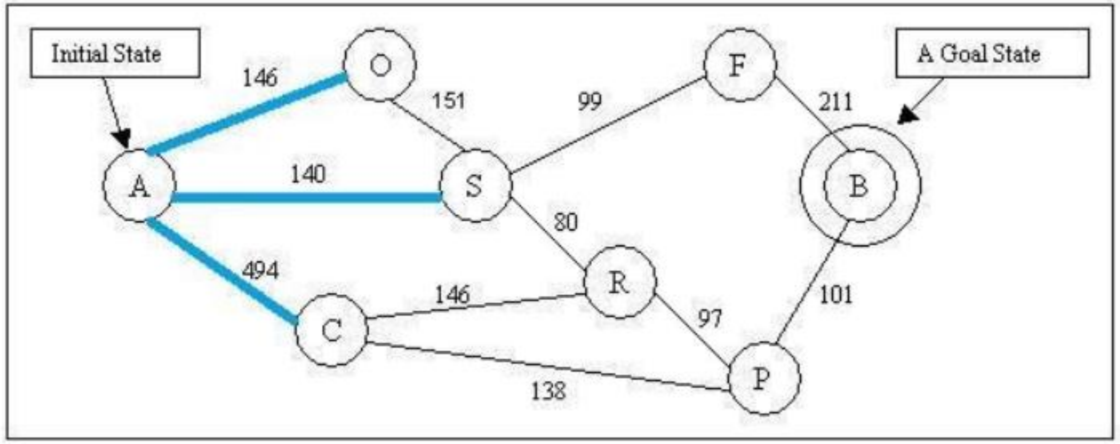
Example

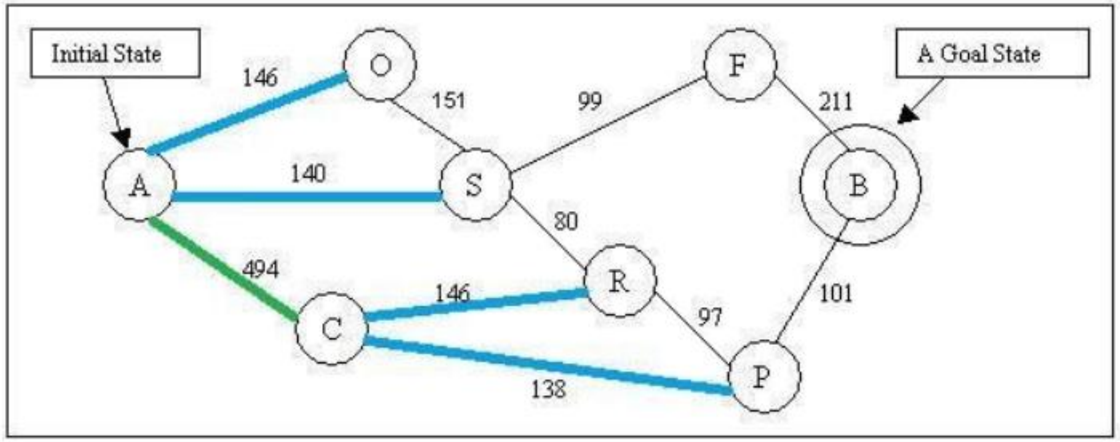


This is a map of a city. We are going to find out a path from city A to city B.

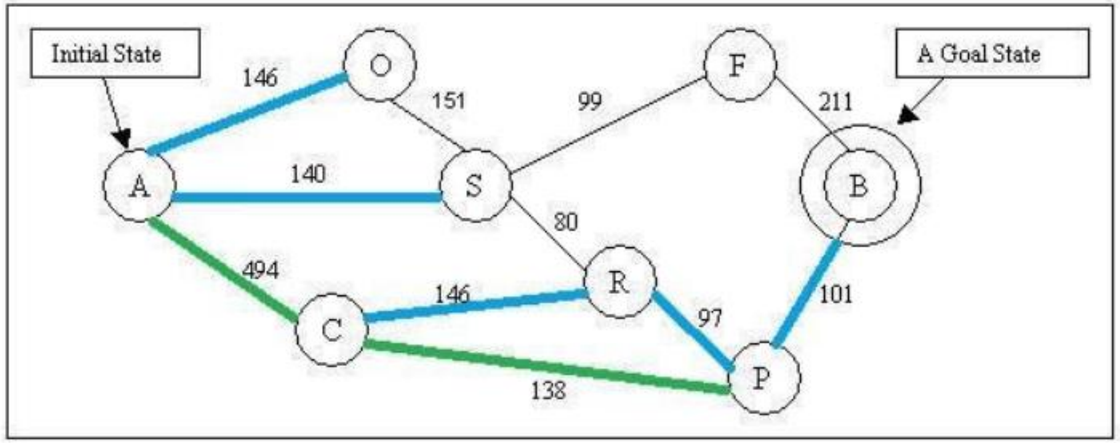


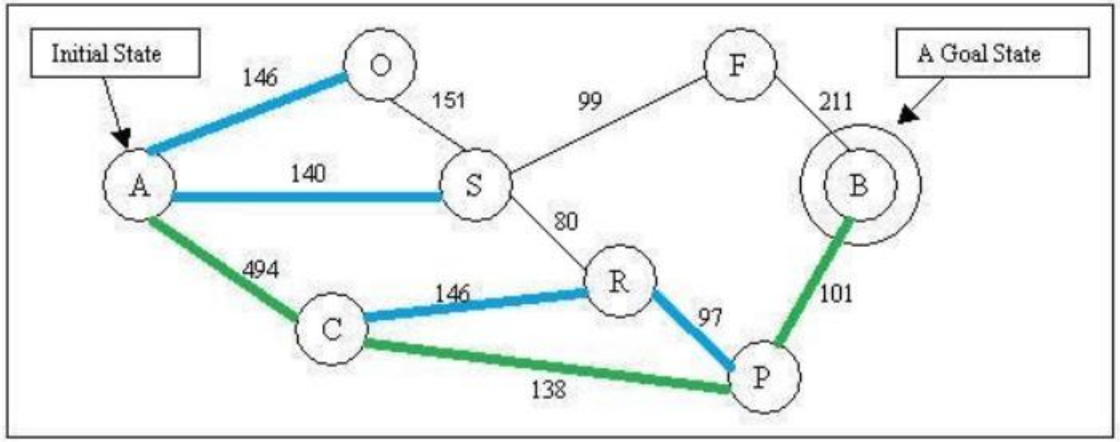
Example: contd.





Example: contd.





Properties of greedy best first search

- If the search space is finite, then, similar to DFS, greedy best-first search can be **complete**.
- As we can see in the example above, greedy best-first search isn't guaranteed to find the shortest path. So, it is **not optimal**.
- In worst case, the time complexity of greedy best-first search is , same to DFS. However, greedy best-first search performance can be improved largely by using a well-designed heuristic function. In most cases, the time complexity of greedy best-first search can be better than BFS's .
- The space complexity of greedy best-first search is similar to its time complexity.

References

- Artificial Intelligence, Russell & Norvig, Pearson

Web Resources

https://ai-master.gitbooks.io/heuristic-search/content/properties-of-greedy-best-first-search.html