CENG 466
Artificial Intelligence

Lecture 4

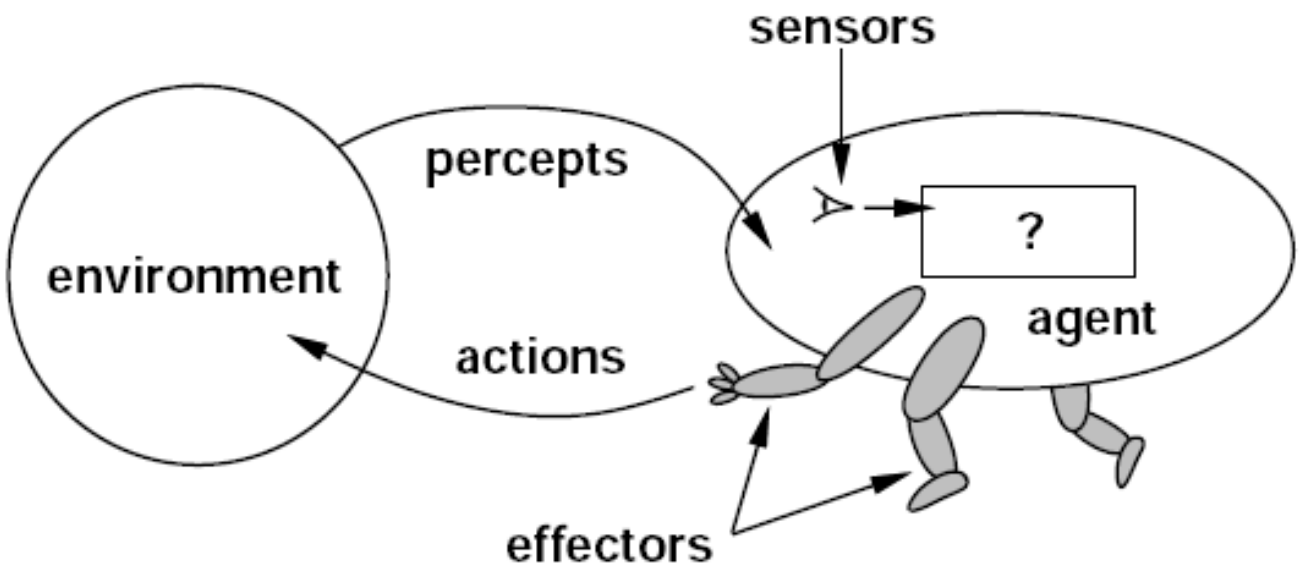
Solving Problems by Searching (II)

Topics

- Search Categories
- Uninformed Search Algorithms
- Informed Search Algorithms
- Best First Search
- Greedy Search
- A* Search
- Iterative Deepening A* Search
- Hill Climbing Search
- Simulated Annealing Search

Intelligent Agents

- An agent is something that perceives and acts in an environment
- An ideal agent always takes actions that maximizes its performance
- An agent adopts a goal and searches the best path to reach that goal



States and State-Spaces

- State: The set of all information items that describe a system at a given time.
- State space is the set of states that an intelligent agent can be in.
- An action takes the agent from one state to another one.
- State space search is finding a sequence of states starting from the initial state to the goal

Searching

- Assuming that the agent knows:
  - how to define a problem,
  - how to recognize a solution (goal),
- finding a solution is done by a search through the state space.

Search Categories

- Un-informed Searches: If we have no extra information about the problem
- Informed Searches: If we have extra information about the problem.

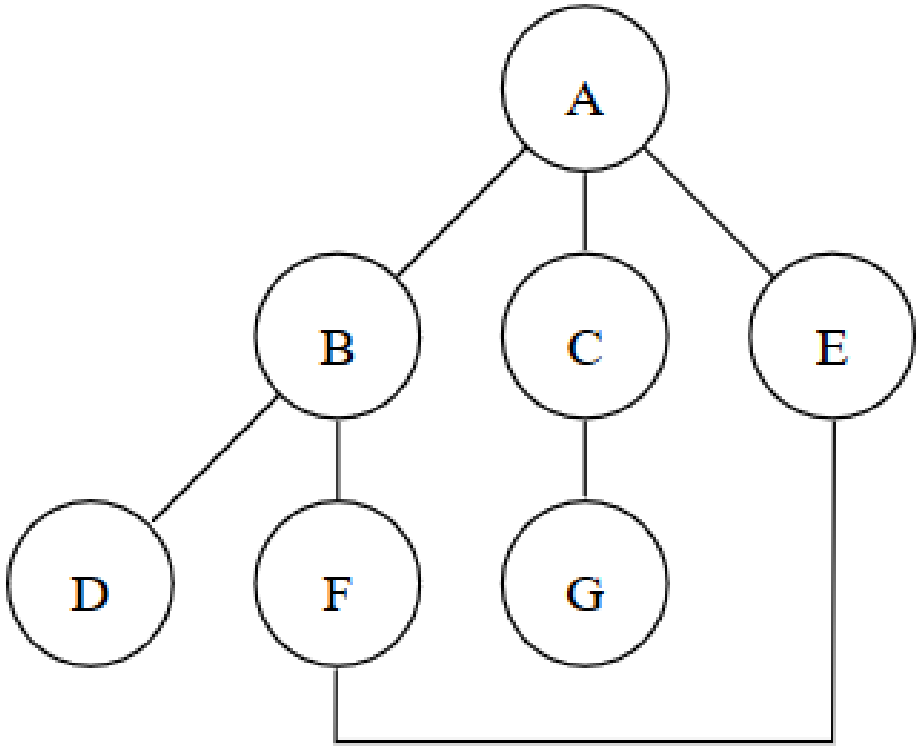
Un-informed Searches

- In un-informed searches, the agent knows:
  - The initial state
  - The goal state
- But it does not know if a state is close to the goal or not
- Therefore, these searches are blind searches

Depth Limited Search

- **Depth-limited search** avoids the problems of depth-first search by imposing a cutoff on the maximum depth of a path.
- This cutoff can be implemented with a special depth-limited search algorithm.
- For example, when finding a path from a city to another city, if we know that there are 20 cities, so we know that the solution must be of length 19 at most.

Depth Limited Search Example



Depth Limited Search Example

- If the search algorithm can remember which states have been visited before, then starting from A we have
  - A, B, D, F, E, C, G
- Otherwise the search algorithm will be in an infinite loop:
  - A, B, D, F, E, A, B, D, F, E, etc.
- Depth limited search can solve the problem.
  - Max depth = 2 ➔ A, B, D, F, C, G, E, F

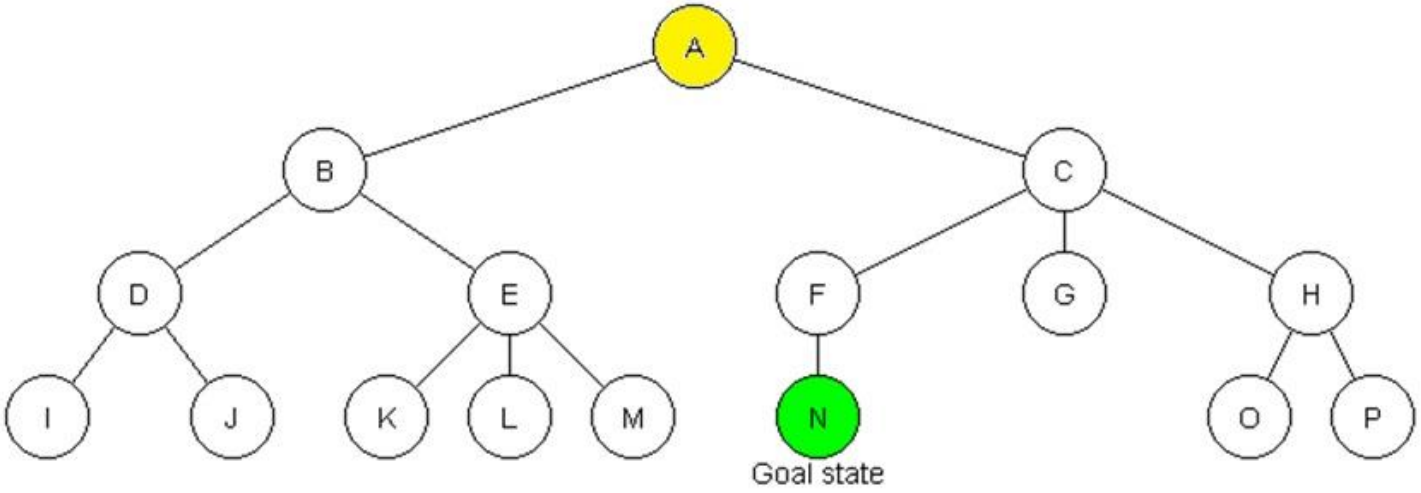
Iterative Deepening Search

- The hard part about depth-limited search is picking a good limit.
- However, for most problems, we will not know a good depth limit until we have solved the problem.
- **Iterative deepening search** starts with depth **n**, then depth **n+k**, then depth **n+2k**, and so on.
- In effect, iterative deepening combines the benefits of depth-first and breadth-first search.

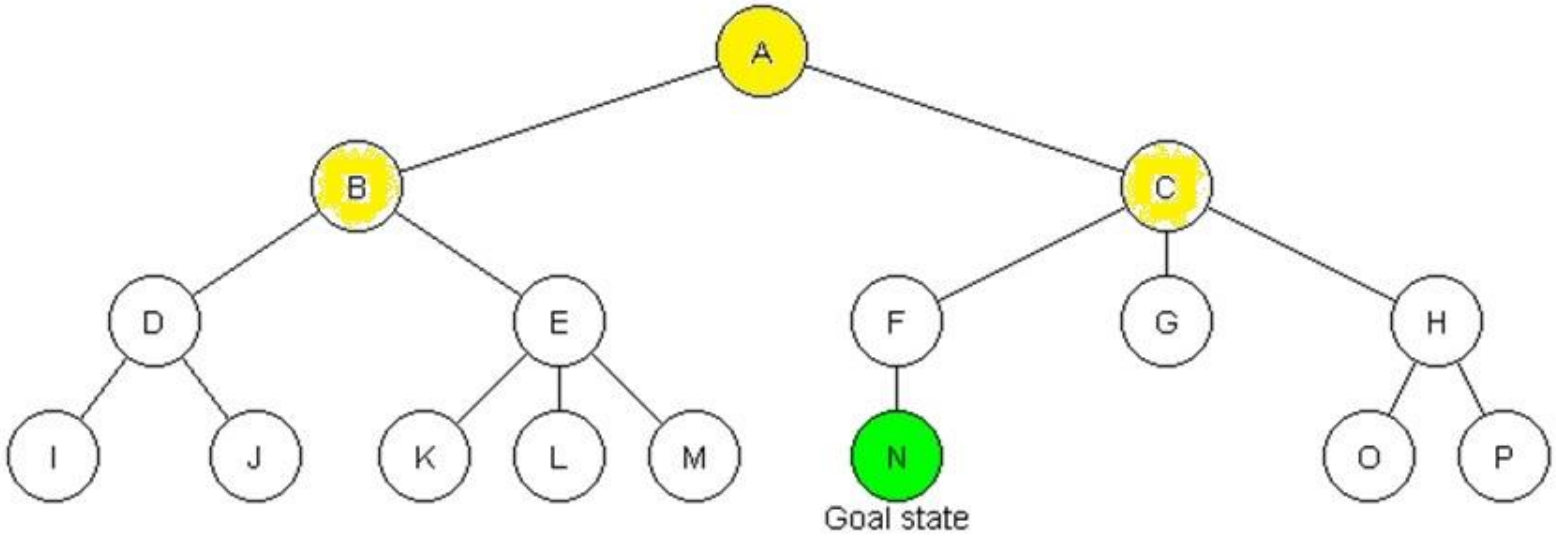
Iterative Deepening Search Example

- Assume we are looking for a goal state in chess game.
- The state-space is very large
- The search for a goal cannot be done by a depth search algorithm because the whole graph cannot be expanded

Example: Depth = 0

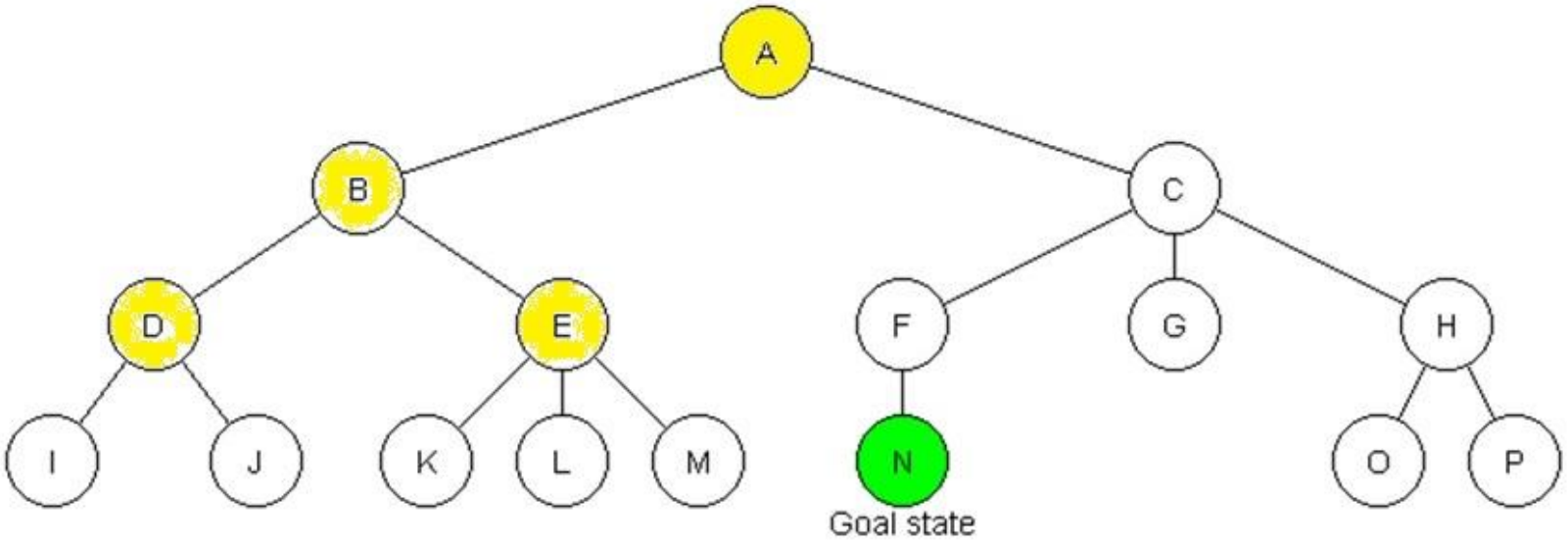


Example: Depth =1



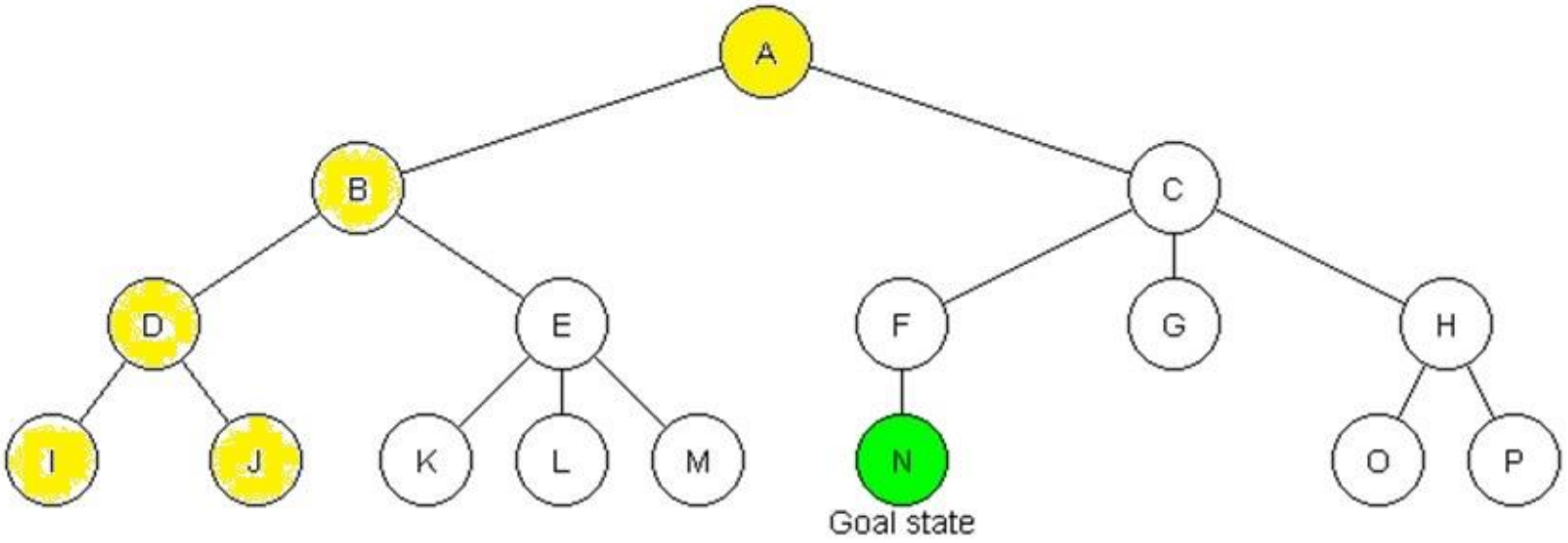
Visits: A, B, C

Example: Depth = 2



Visits: A, B, D, E, C, F, G, H

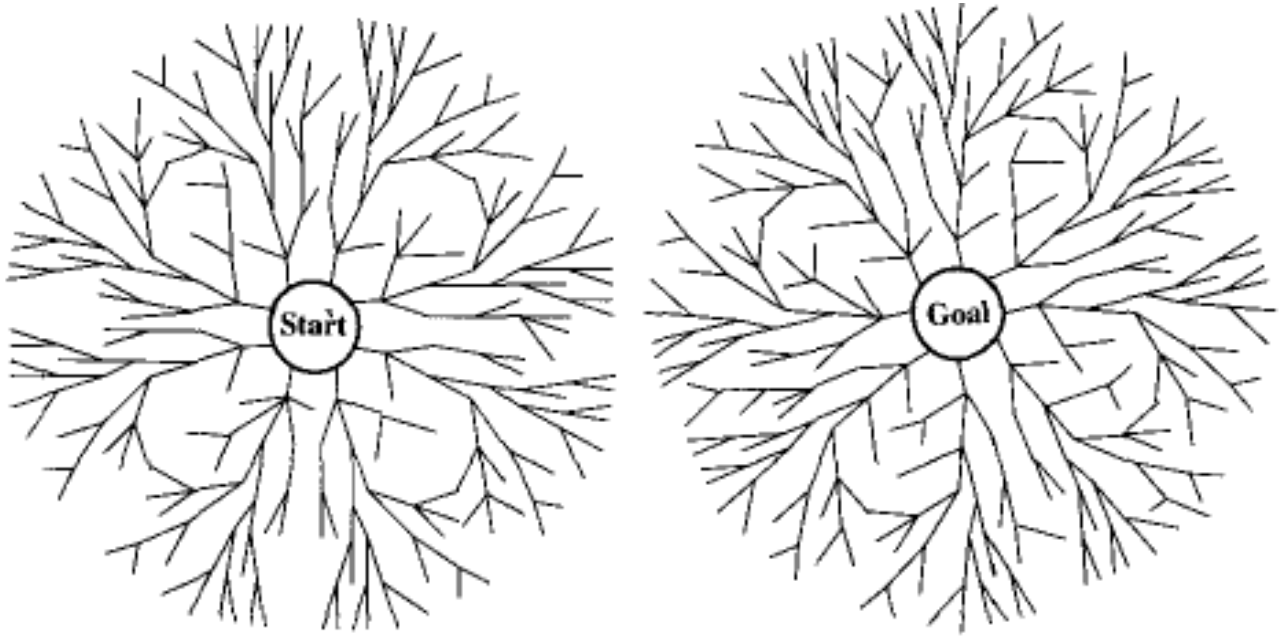
Example: Depth = 3



Visits: A, B, D, I, J, E, K, L, M C, F, N

Bidirectional Search

- The idea in bidirectional search is to search both forward from the initial state, and backward from the goal at the same time.
- The algorithm stops when the two searches meet in the middle



Avoiding Repeating States

- There are three ways to avoid repeated states:

1. Do not return to the state that we just came from.
2. Do not create paths with cycles in them.
3. Do not generate any state that was ever generated before. This requires every state that is generated to be kept in memory.

Informed Searches

- Informed searches have some extra information about the problem
- At each state, we can estimate how far we are from the goal
- Using this information, we can have better search algorithms

How to Use the Information in Searches

- If the state space of a problem is large, we expand only some of the nodes.
- Choosing the next node to expand is the main difference between the search algorithms.
- An informed search algorithm uses its extra information when it decides which node should be expanded

Informed Search Algorithms

- Best First Search
- Greedy Search
- A* Search
- Hill Climbing Search
- Simulated Annealing Search

Best First Search

- Best First Search puts the nodes in order so that, the node with the best value is expanded first.
- The best node is the node, that appears to be best according to the evaluation function.
- In most cases, evaluation function only estimates the value of the nodes.

Evaluation of States

- The aim of searching in state space graph is finding the best path to a goal.
- To find the best path we should define a value for each state.
- This value shows how far we are from the goal
- Evaluation function calculates the distance to goal.
- If exact calculation of the distance to goal is not possible, we use estimates, this is called heuristic function.

Evaluation in Best-First Search

- Best first search algorithms use some estimated measure of the cost, and try to minimize it.
- Two basic approaches for estimating cost are:
  - The greedy search which tries to expand the node closest to the goal.
  - The A* search which tries to expand the node on the least-cost solution path.

Greedy Search

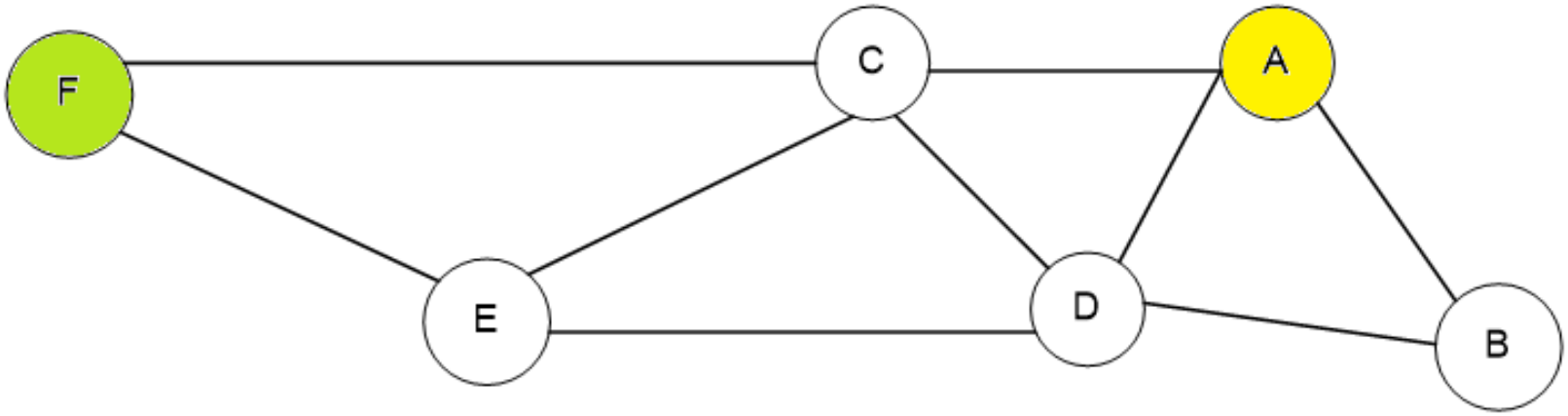
- Greedy search is one of the simplest best-first search strategies
- Greedy search minimizes the estimated cost to reach the goal.
- The cost of reaching the goal from a state can be estimated but cannot be determined exactly.
- A function that calculates such costs is called a **heuristic cost function**

Best First Search Example

- Assume a graph of cities and the roads connecting them is given.
- The initial city and the goal city are defined.
- The evaluation function is based on the geographical coordinates of the cities.

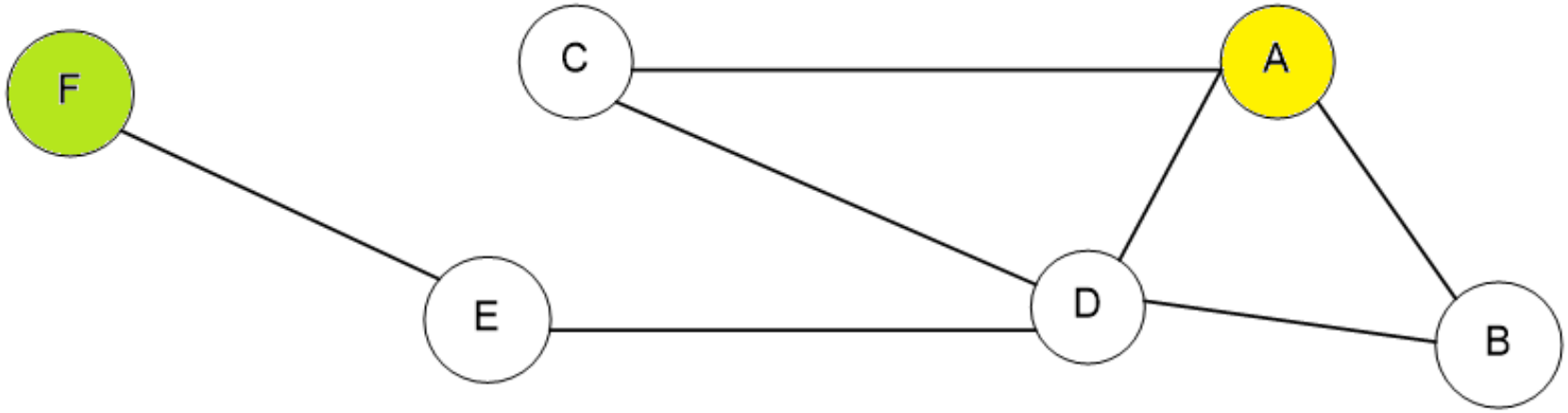
Greedy Search Solution (I)

- Initial city is A. Goal is F. The evaluation function expands C because its location is closer to the location of the Goal



Greedy Search Solution (II)

- Not in all cases the evaluation function gives a good estimation.



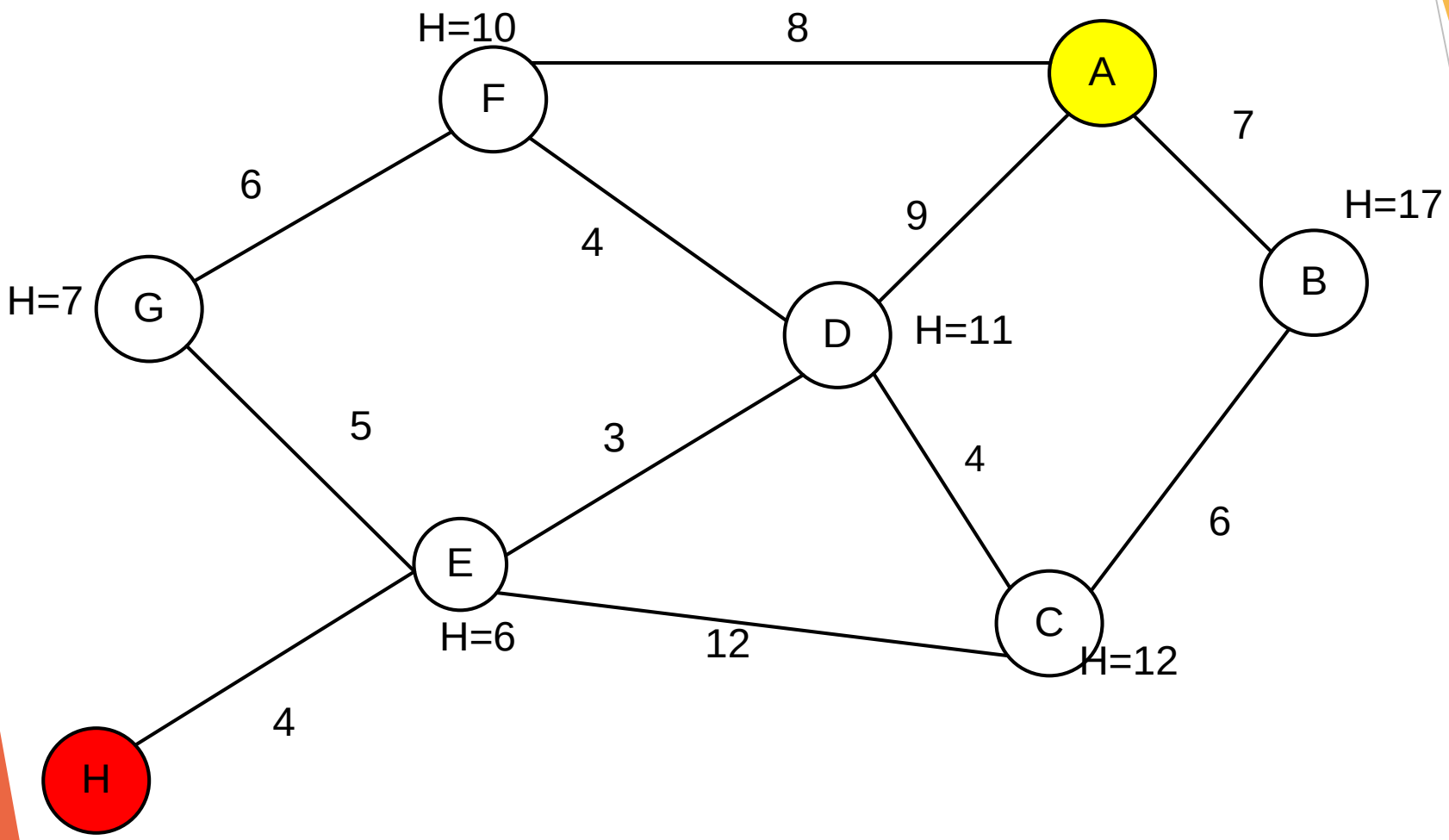
A* Search

- Greedy search minimizes the estimated cost to the goal, h(n).
- Uniform-cost search minimizes the cost of the path so far, g(n).
- A* combines these two strategies to get the advantages of both.

$$f(n) = g(n) + h(n)$$

Example: A* Search

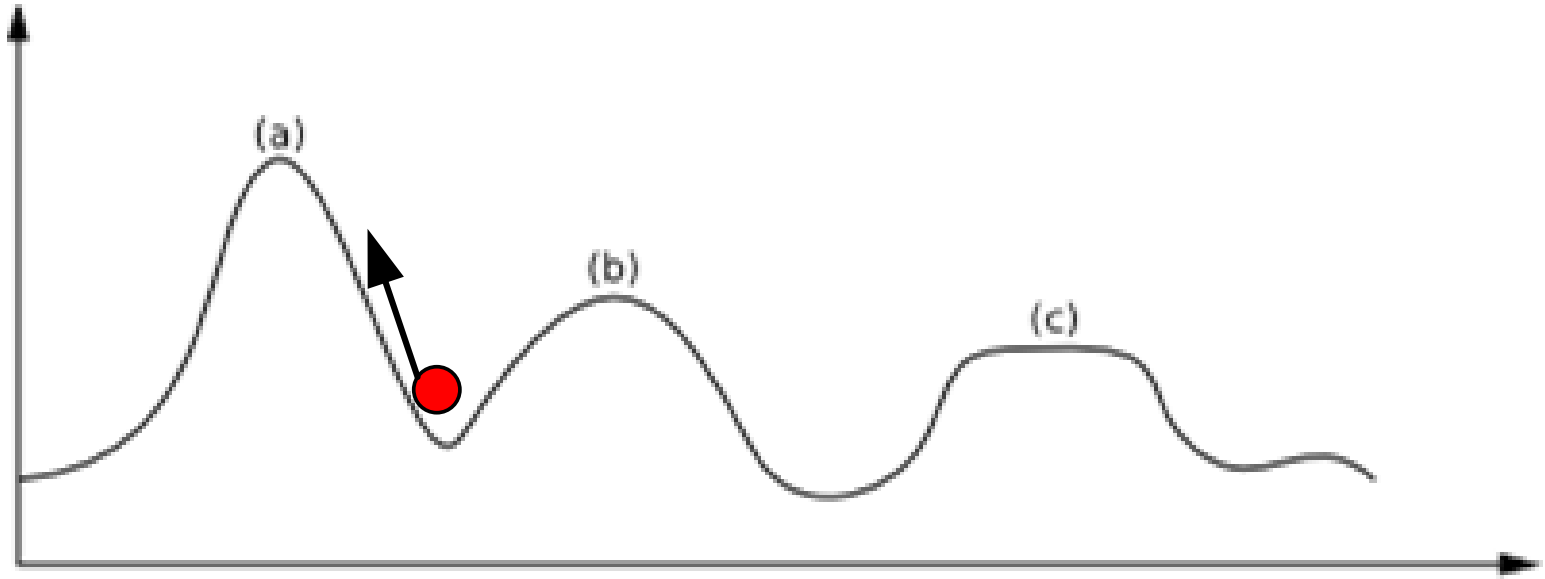
- Assume the estimation to goal [h(.)] and the distance between cities are as shown in the figure.
- Initial city is A
- Goal is H


H=10
8
A
F
7
6
9
H=17
4
B
H=7
G
D
H=11
5
3
4
6
E
C
H=6
12
H=12
4
H

Hill-Climbing Search

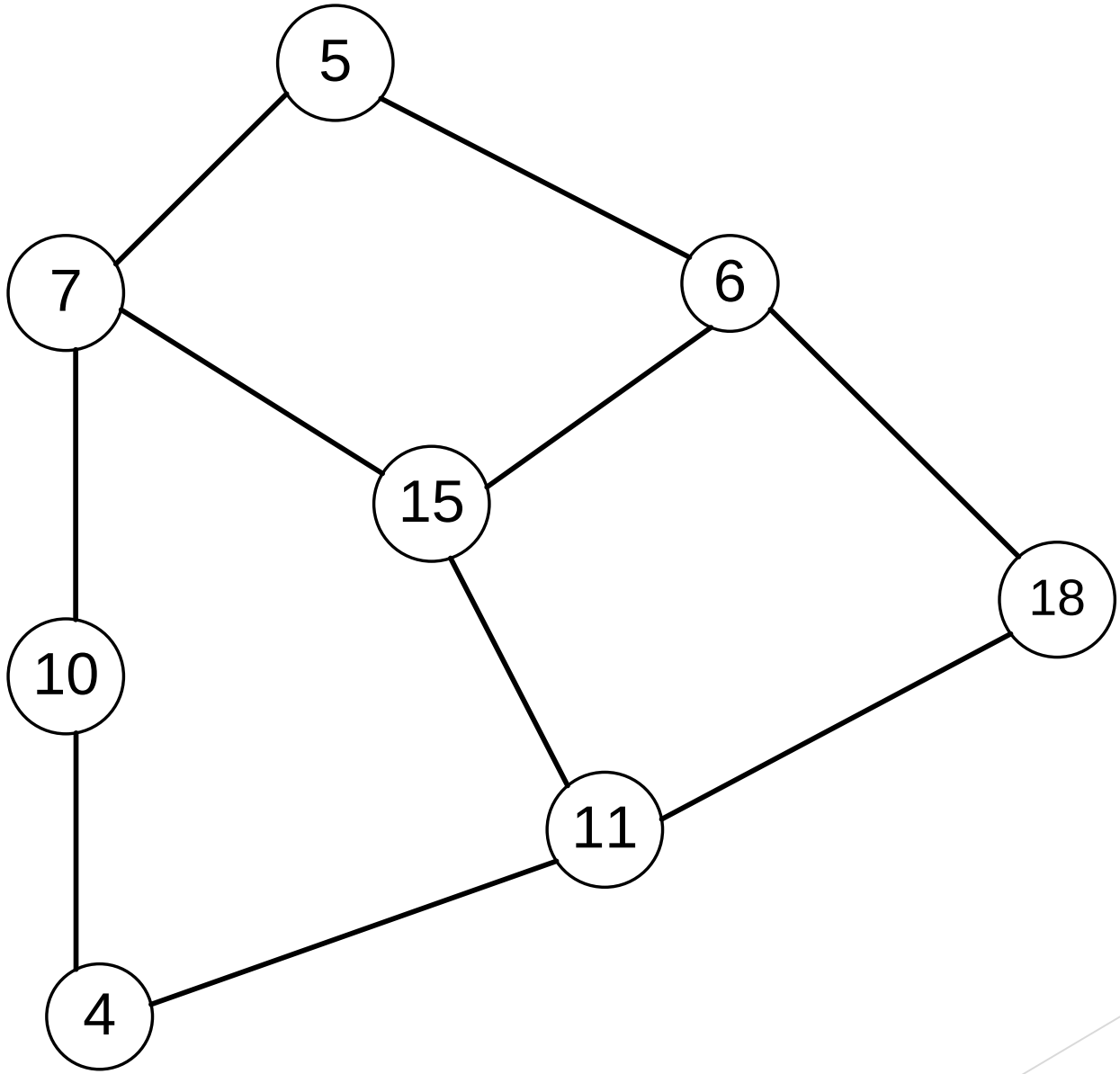
- The hill-climbing search algorithm moves in the direction of increasing values.
- The algorithm only follows the neighbors having larger values.
- The algorithm may stop at local maximum nodes.

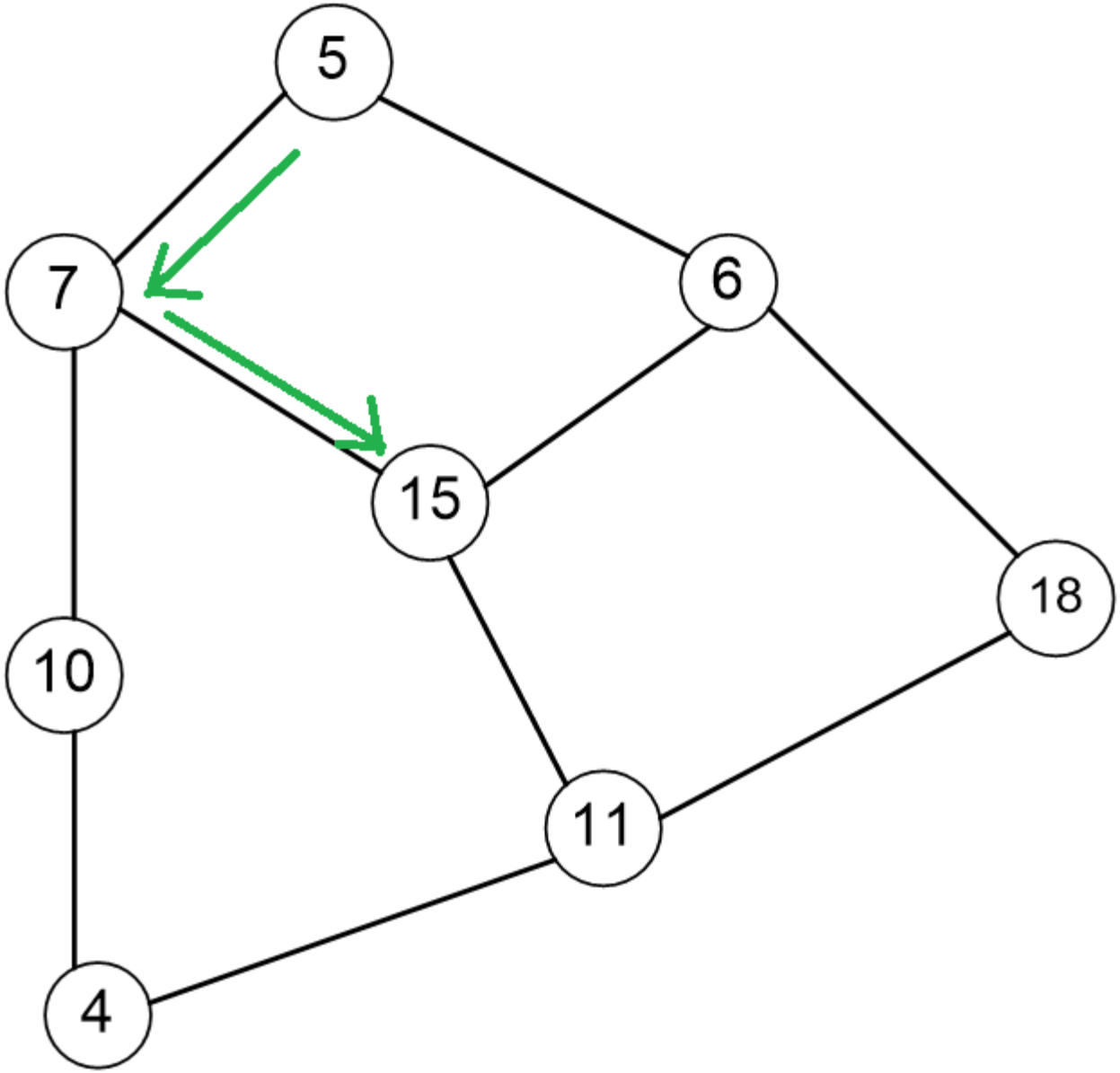
Hill-Climbing Search



Example: Hill-Climbing Search

- Assume a graph of nodes with different values is given.
- Starting from an initial node, find the node with the maximum value.


5
7
6
15
18
10
11
4


5
7
6
15
18
10
11
4

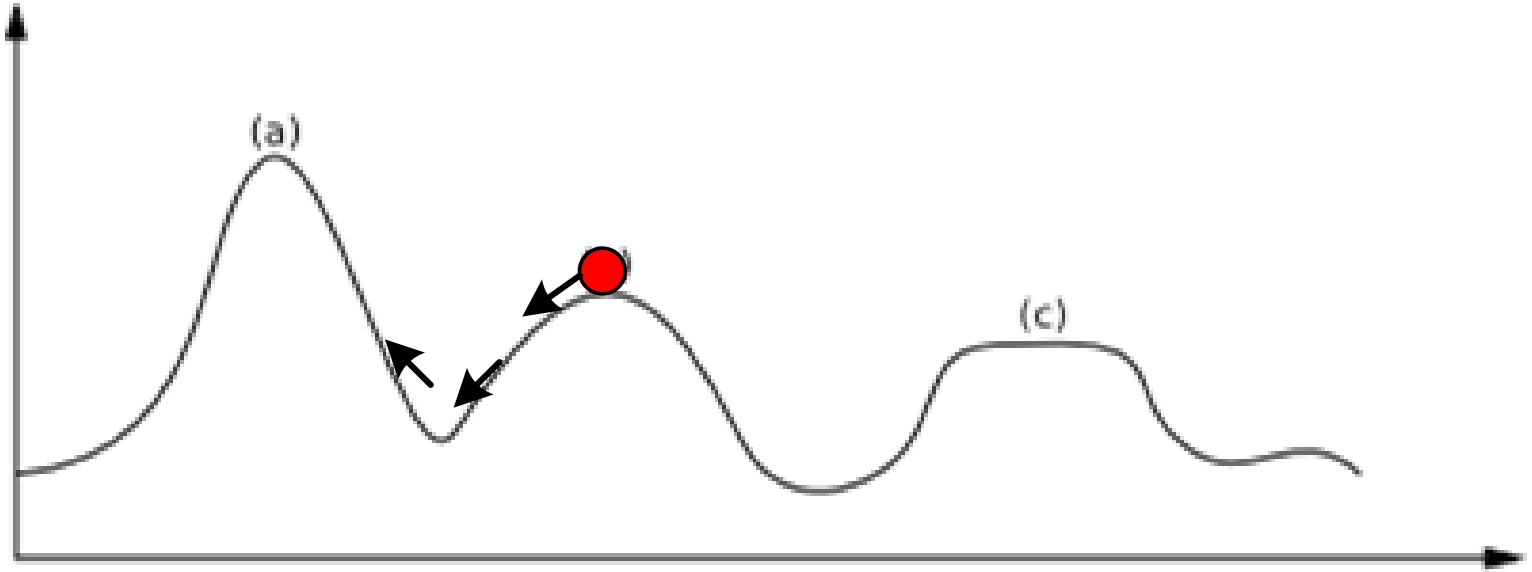
Problem with Hill-Climbing Search

- Hill-Climbing stops at local maximums.
- In the previous example, starting from 5, Hill-Climbing finds 15 as the maximum value. But the node with 18 has the maximum value.
- As a solution when the hill-climbing stops at a maximum point we re-start it from a random point.
- If the algorithm goes to the same point, that point is probably a global maximum.

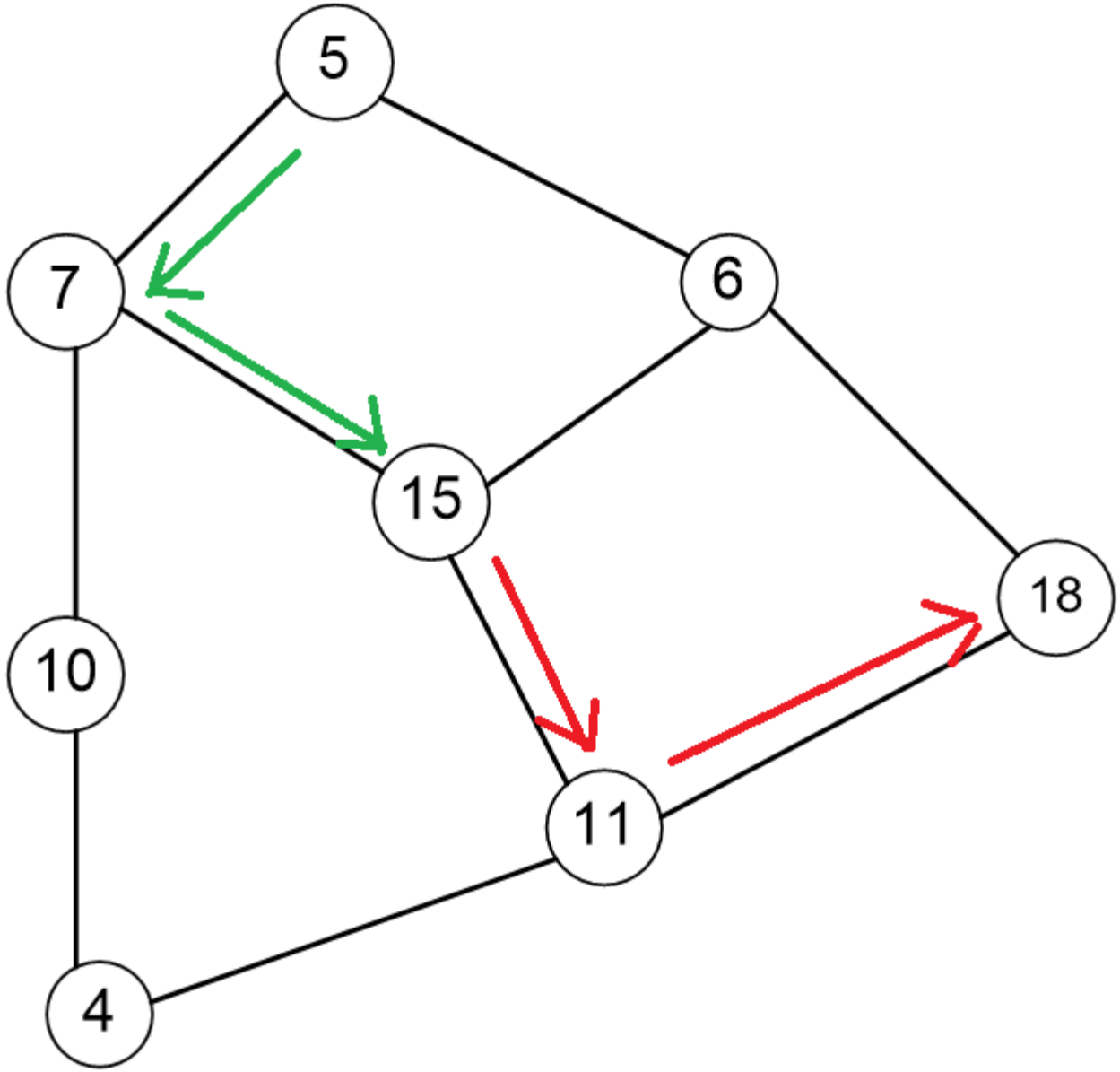
Simulated Annealing

- Simulated Annealing solves the local maximum problem in hill-climbing algorithm by allowing it to follow down-hill paths in a limited range.

Example: Simulated Annealing

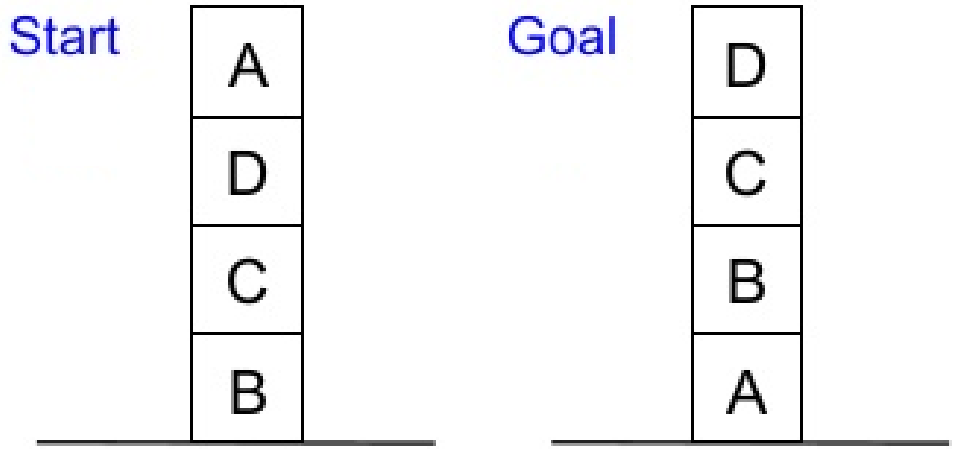


Example: Simulated Annealing



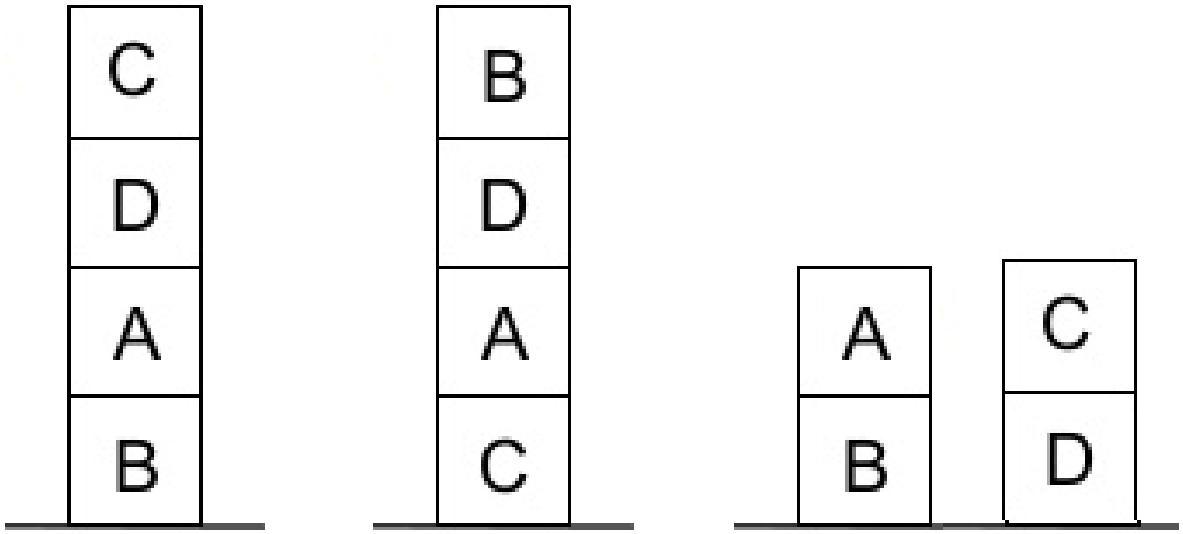
Example

- Assume four boxes are stacked on each other as shown in the following figure (left). The goal it putting them in right order as shown below (right)



Example

- Each order of the blocks is a state. Some examples are shown below:

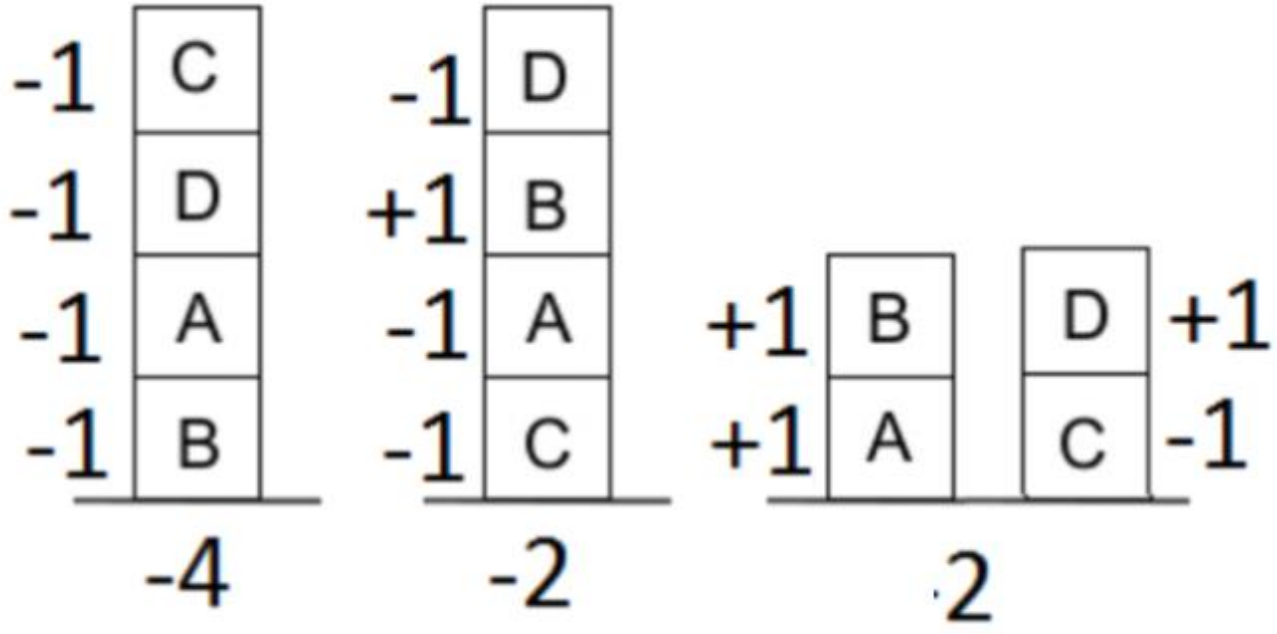


Example

- Assigning a score to each state:
  - If a block is on top of right block give it a score of +1
  - If a block is on top of wrong block give it a score of -1
  - Add up the scores to get the score of the state

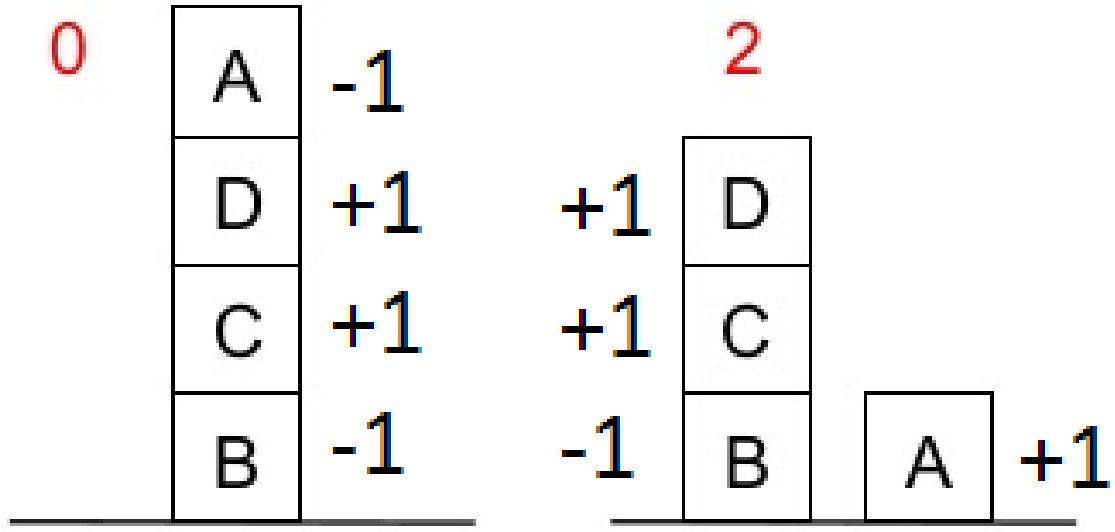
Example

- Example scores of states:

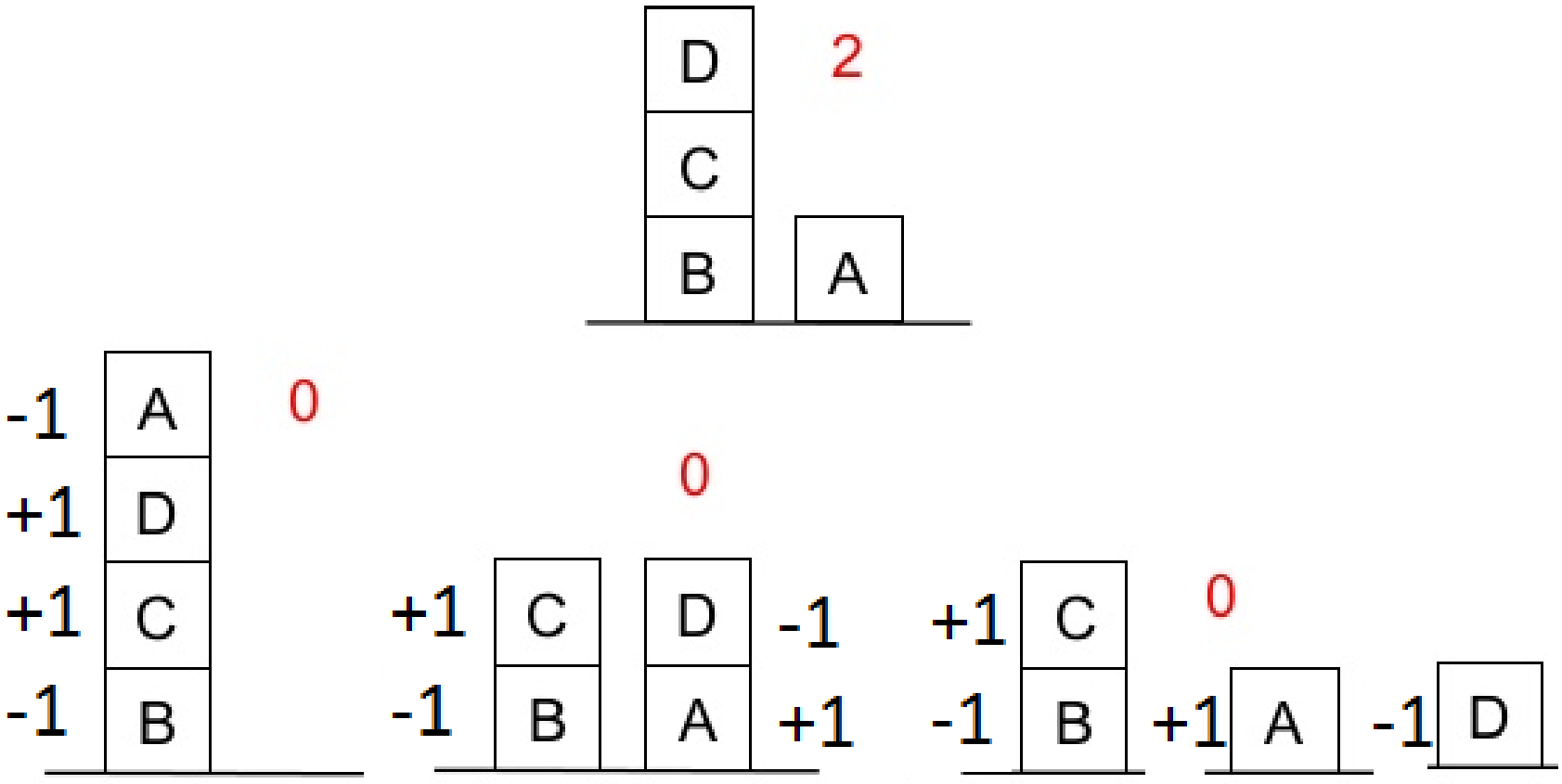


Example

- Starting from the first state, move to next state:



Example



Questions?

Future Intelligent Agents !!!