Informed Search

CS 486/686: Introduction to Artificial Intelligence

Outline

- Using knowledge
  - Heuristics
- Best-first search
  - Greedy best-first search
  - A* search
  - Variations of A*
- Back to heuristics

Last lecture

- Uninformed search uses no knowledge about the problem
  - Expands nodes based on “distance” from start node (never looks ahead to goal)
- Pros
  - Very general
- Cons
  - Very expensive
- Non-judgemental
  - Some are complete, some are not

Informed Search

- We often have additional **knowledge** about the problem
  - Knowledge is often merit of a node (value of a node)
    - Example: Romania travel problem?
- Different notions of merit
  - **Cost of solution**
  - Minimizing computation

Informed Search

- Uninformed search expands nodes based on distance from start node, $d(n_{start}, n)$
- Why not expand on distance to goal, $d(n, n_{goal})$?
- What if we do not know $d(n, n_{goal})$ exactly?
  - **Heuristic function, h(n)**

Example: Path Planning

- Romania example
  - What is a reasonable heuristic?
  - Is it always right?

Heuristics

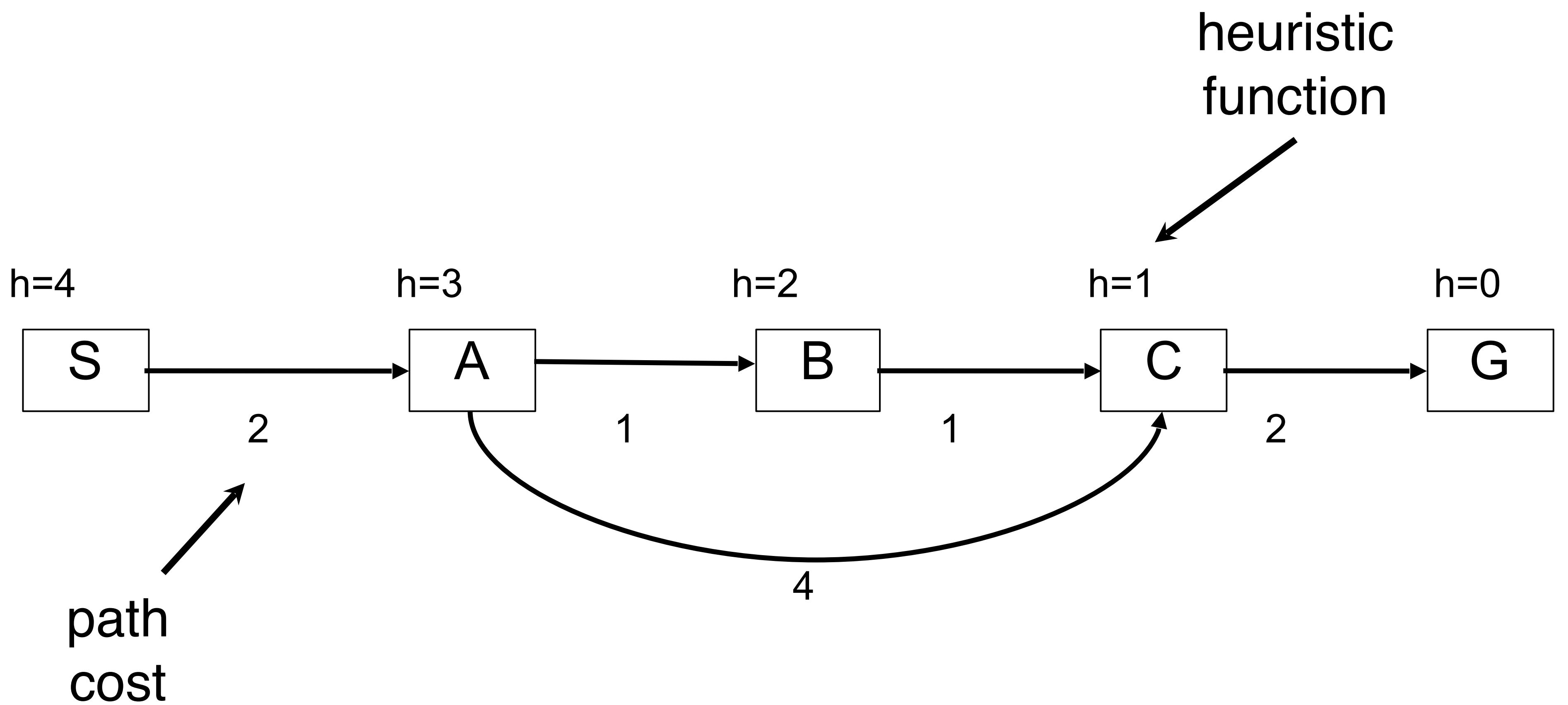
- If $h(n_1) < h(n_2)$
  - We guess it is cheaper to reach the goal from n_1 than n_2
- We require $h(n_{goal})=0$

- For now, just assume we have some heuristic $h(n)$

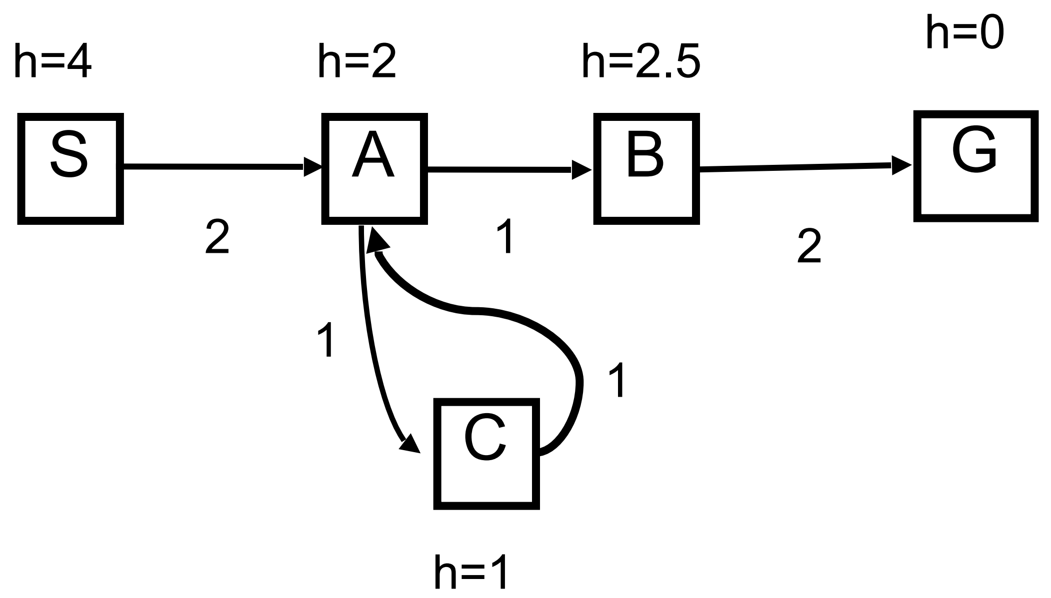
(Greedy) Best-First Search

- Expand the most promising node according to the heuristic
- Best-first is similar to DFS (how similar depends on the heuristics)
- If h(n)=0 for all n, best-first search is the same as BFS

Example: Best First search



Example: Best First Search



Judging Best First Search

- Good news
  - Informed search method
- Bad news
  - Not optimal
  - Not complete: but OK if we check repeated states
  - Exponential space: might need to keep all nodes in memory
  - Exponential time ($O(b^m)$)
    - but if we choose a good heuristic then we can do much better! (See Good news)

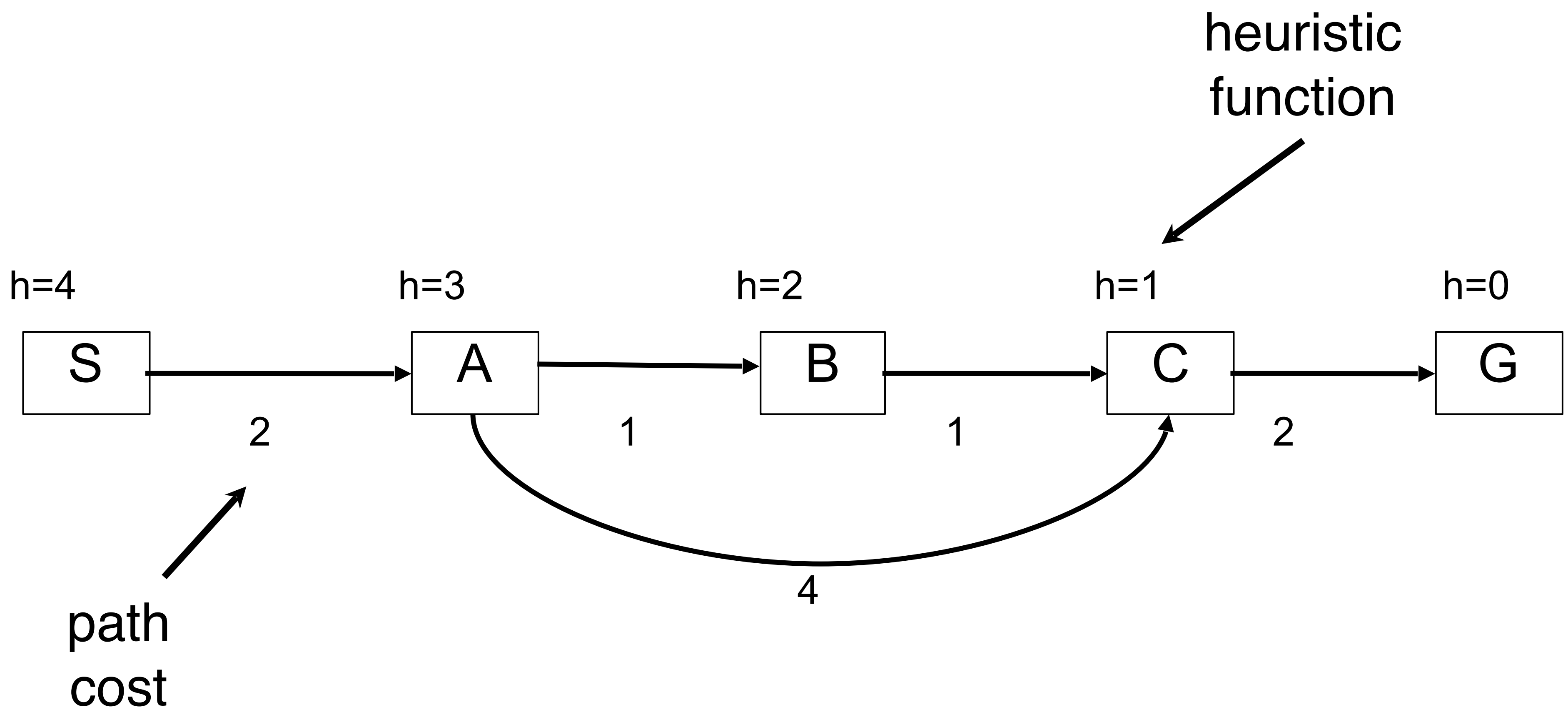
A* Search

- Best-first search is too greedy
- Solution?
  - Let g be the cost of the path so far
  - Let h be a heuristic function
  - Let $f(n)=g(n)+h(n)$
    - estimate of cost of current path
- A* search
  - Expand node in fringe with lowest f-value

A* Search

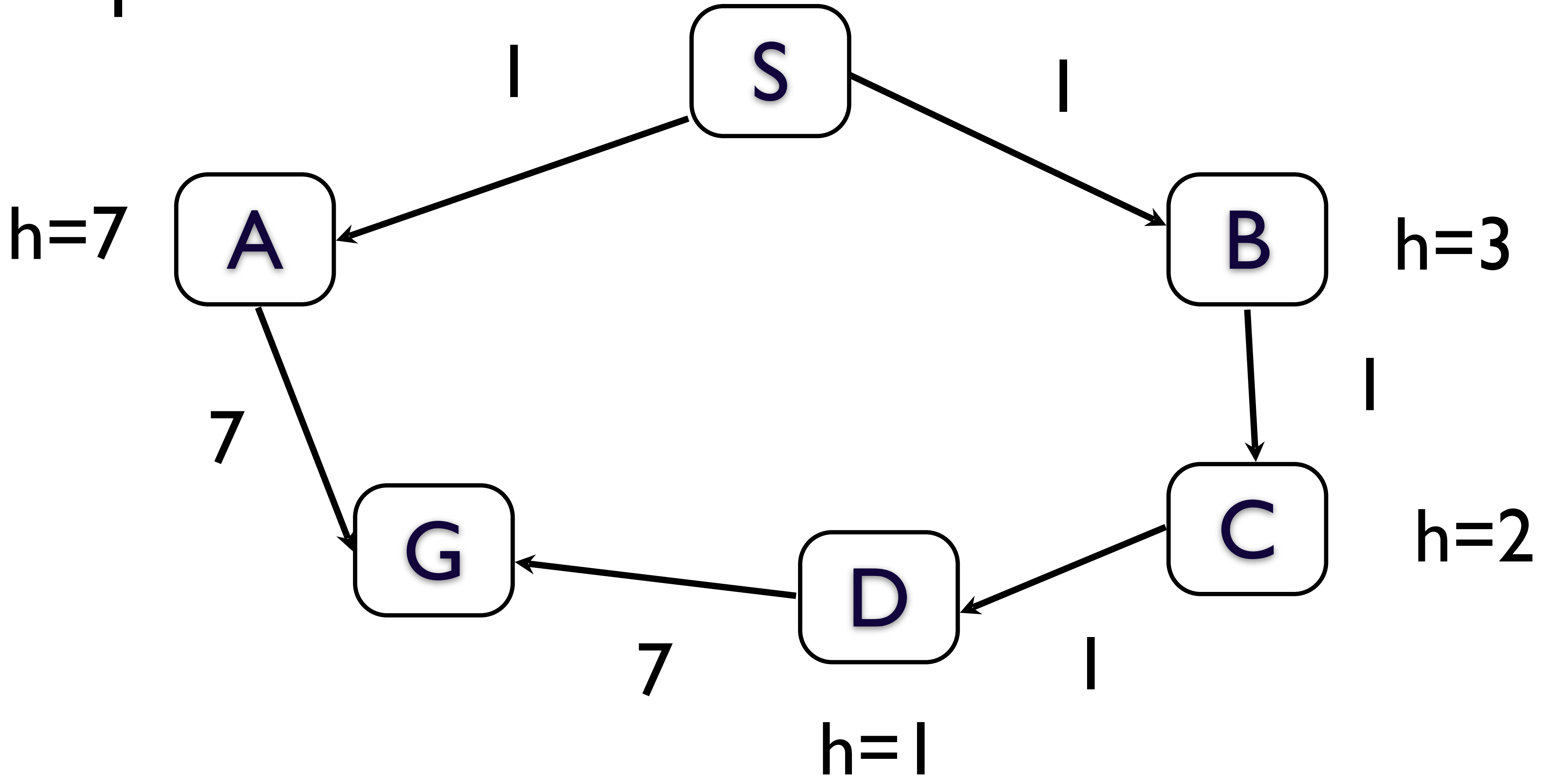
- Algorithm
  - At every step, expand node n from front of the queue
  - Enqueue the successor n’ with priorities f(n’)=g(n’)+h(n’)
  - Terminate when **goal state is popped from the queue**

Example: A* search



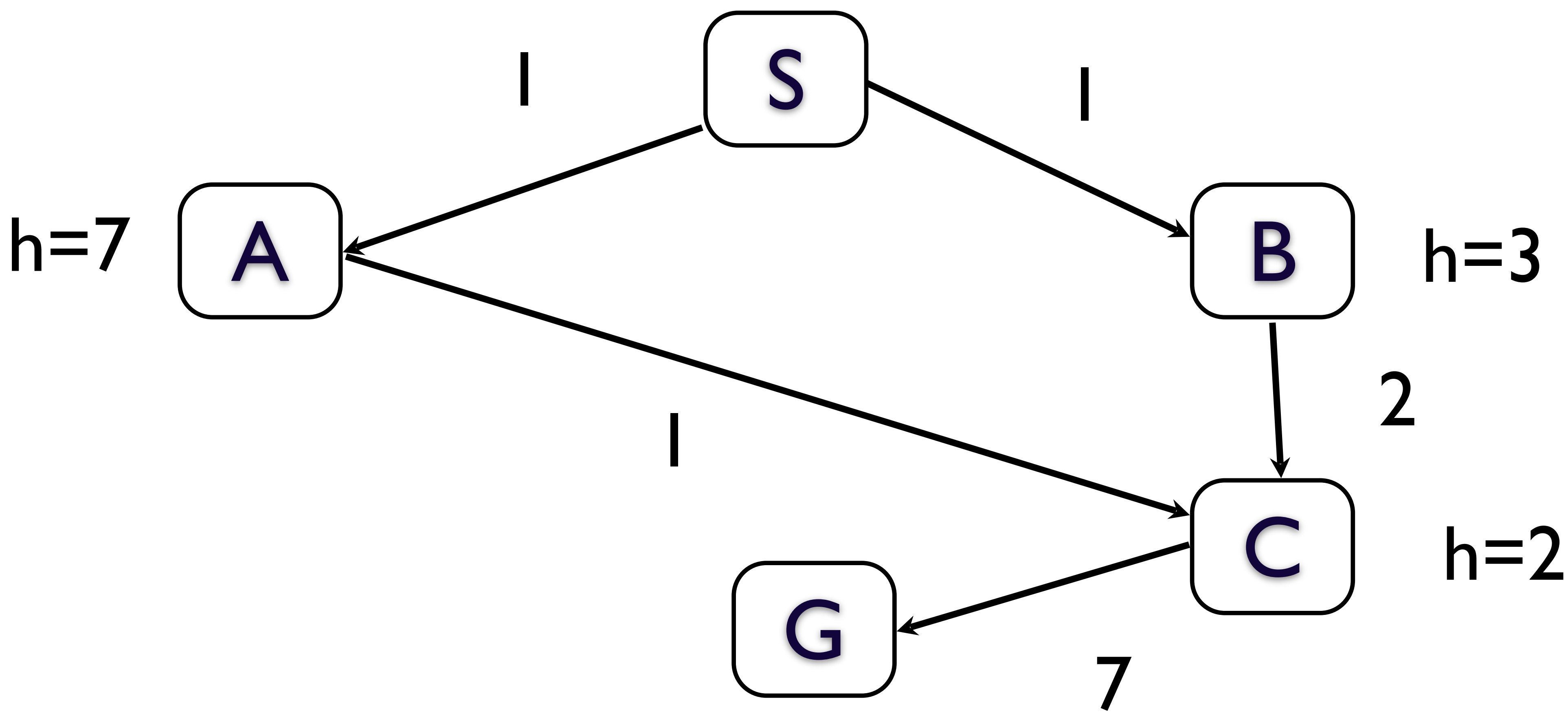
When Should A* Terminate?

- Only when G has been popped from the queue

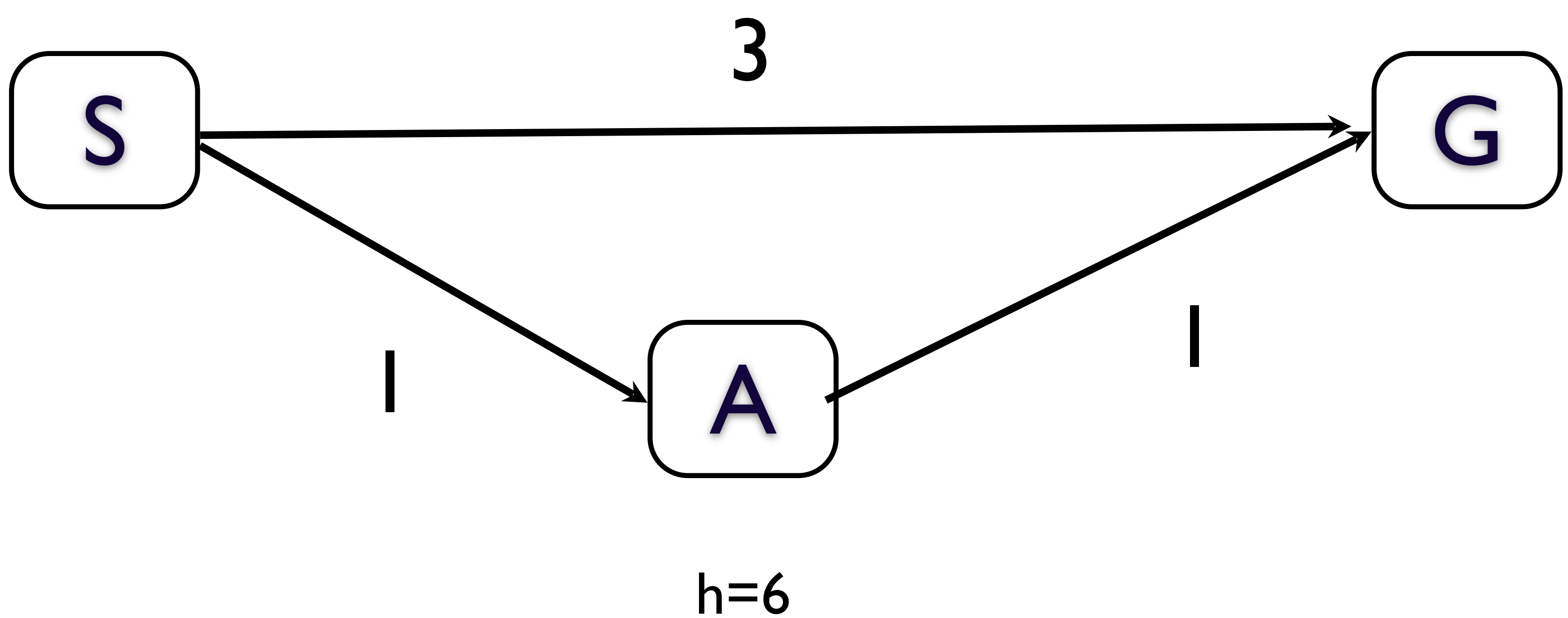


A* and Revisiting States

- What if we revisit a state that was already expanded?



Is A* Optimal?



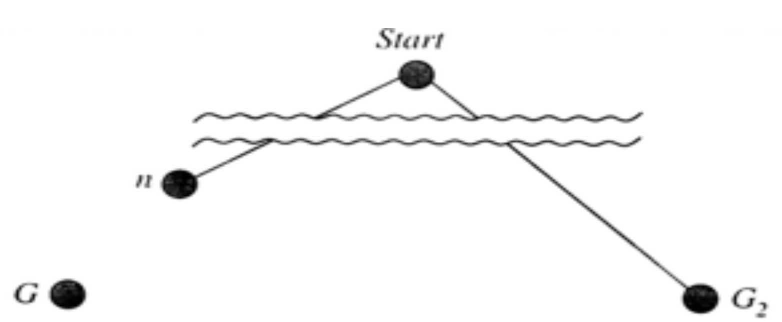
Admissible Heuristics

- Let $h^*(n)$ be the shortest path from n to any goal state
- A heuristic is **admissible** if $h(n) \leq h^*(n)$ for all n
- Admissible heuristics are optimistic
- Always have $h(n_{goal})=0$ for any admissible heuristic

Optimality of A*

- If the heuristic is admissible then A* with tree-search is optimal

Proof by contradiction
Let goal G_2 be in the queue. Let n be an unexpanded node on the shortest path to optimal goal G.
Assume that A* chose G_2 to expand. Thus, it must be that $f(n)>f(G_2)$



But
$f(G_2)=g(G_2)$ since $h(G_2)=0$
$>= g(G)$ since G_2 is suboptimal
$>= f(n)$ since h is admissible

Contradiction. Therefore, A* will never select G_2 for expansion.

Optimality of A*

- For graphs we require consistency
  - $h(n) \leq cost(n,n') + h(n')$
  - Almost any admissible heuristic function will also be consistent
- A* search on graphs with a consistent heuristic is optimal

Judging A*

- Good news
  - Complete
  - Optimal (if heuristic is admissible)
  - Time complexity: Exponential in worst case but a good heuristic helps a lot
- Bad news
  - A* keeps all generated nodes in memory
  - On many problems A* runs out of memory

Memory-Bounded Heuristic Search

- Iterative Deepening A* (IDA*)
  - Basically depth-first search but using the f-value to decide which order to consider nodes
  - Use f-limit instead of depth limit
    - New f-limit is the smallest f-value of any node that exceeded cutoff on previous iteration
  - Additionally keep track of next limit to consider
  - IDA* has same properties as A* but uses less memory

Memory-Bounded Heuristic Search

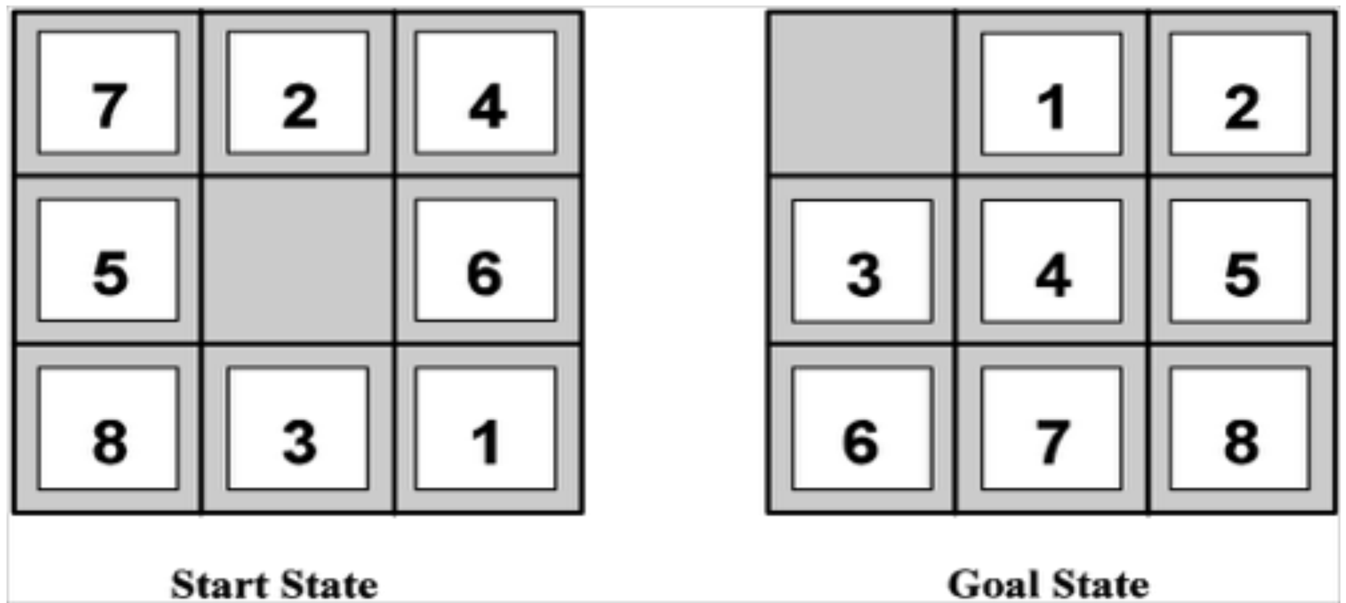
- Simplified Memory-Bounded A* (SMA*)
  - Uses all available memory
  - Proceeds like A* but when it runs out of memory it drops the worst leaf node (one with highest f-value)
  - If all leaf nodes have same f-value, drop oldest and expand newest
  - Optimal and complete if depth of shallowest goal node is less than memory size

Heuristic Functions

- A good heuristic function can make all the difference!

- How do we get heuristics?

8 Puzzle



- Relax the game
    1. Can move from A to B is A is next to B
    2. Can move from A to B if B is blank
    3. Can move from A to B

8 Puzzle

- 3 leads to misplaced tile heuristic
  - Number of moves = number of misplaced tiles
  - Admissible
- 1 leads to Manhatten distance heuristic
  - Admissible

8 Puzzle

- h_1=misplaced tiles, h_2=Manhatten distance
- Note: h_2 **dominates** h_1
  - $h_2(n) \geq h_1(n)$ for all n
  - Even though both h_1 and h_2 are admissible heuristics, h_2 is a better heuristic

8 Puzzle and Heuristics

Depth	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6
4	112	13	12
8	6384	39	25
12	3644035	227	73
24	-	39135	1641

Designing Heuristics

- Relax the problem
- Precompute solution costs of subproblems and storing them in a pattern database
- Learning from experience with the problem class
- ...

Summary

- What you should know
  - Thoroughly understand A*
  - Be able to trace simple examples of A* execution
  - Understand admissibility of heuristics
  - Completeness, optimality