CMU 15-381

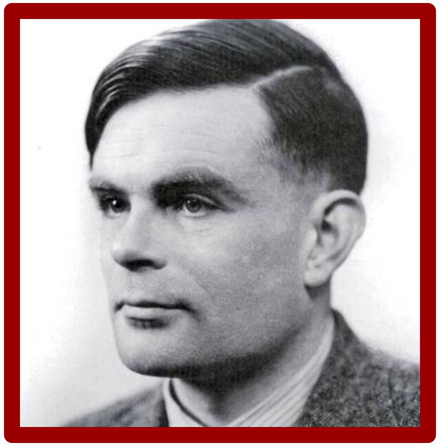
Lecture 4: Informed Search

Teachers:
Emma Brunskill
Ariel Procaccia (this time)

Uninformed vs. Informed

Uninformed

Can only generate successors and distinguish goals from non-goals



Informed

Strategies that know whether one non-goal is more promising than another

Reminder: Tree Search

function Tree-Search(problem, strategy)

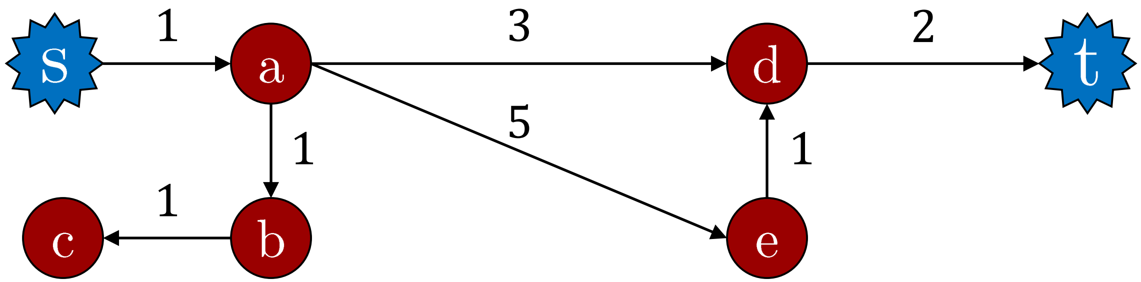
set of frontier nodes contains the start state of problem

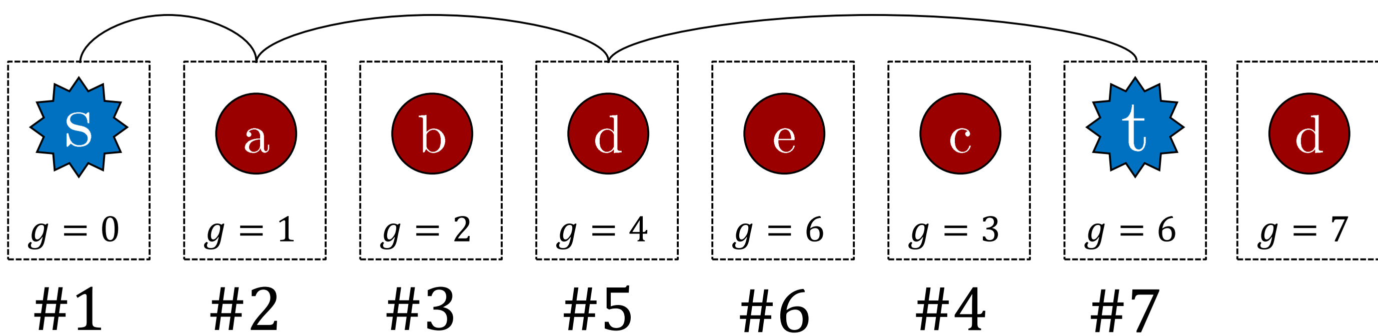
loop

- if there are no frontier nodes then return failure
- choose a frontier node for expansion using strategy
- if the node contains a goal then return the corresponding solution
- else expand the node and add the resulting nodes to the set of frontier nodes

Uniform Cost Search

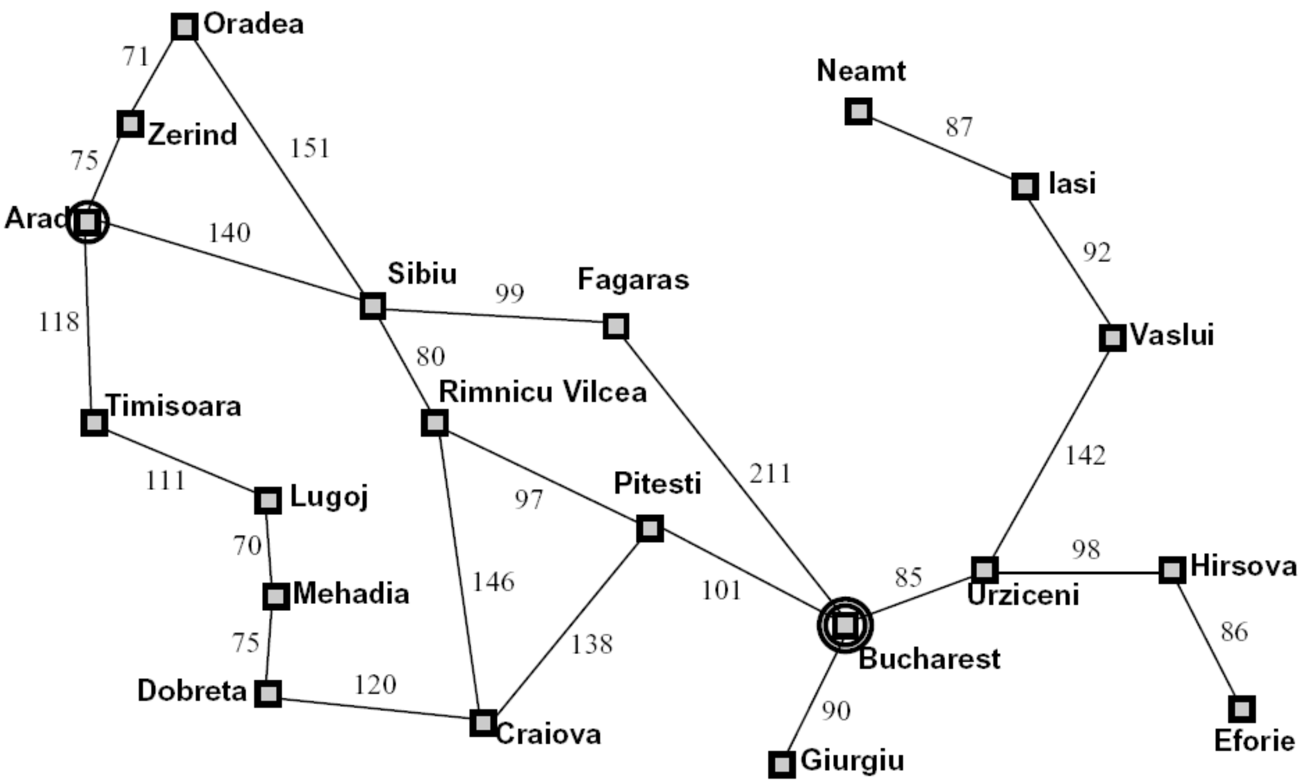
- Strategy: Expand by $g(x)$ = work done so far





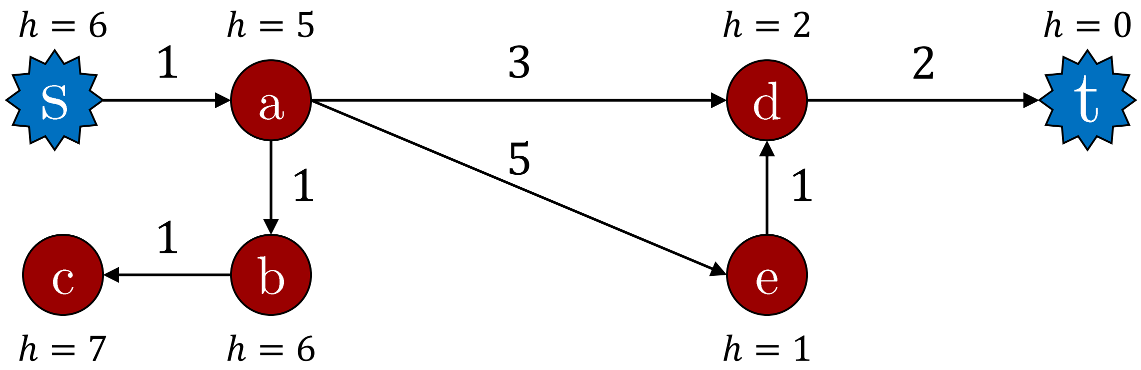
Example: Heuristic

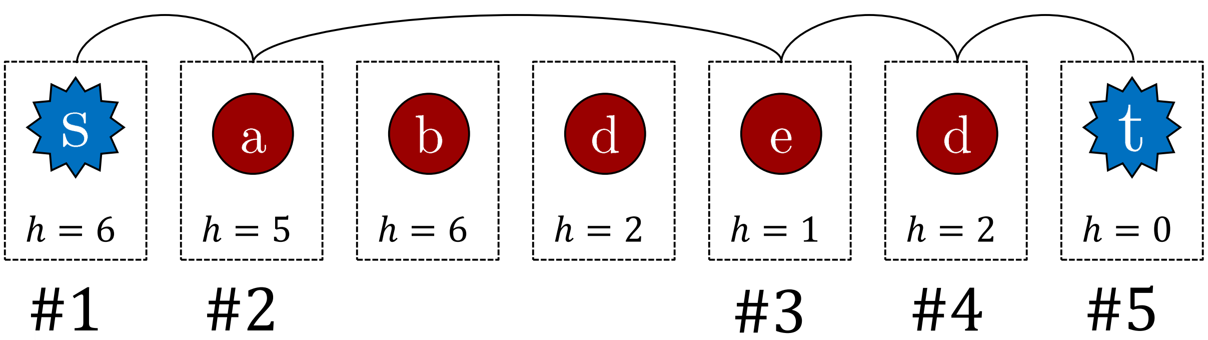
City	Aerial dist
Arad	366
Sibiu	253
Rimnicu Vilcea	193
Fagaras	176
Pitesti	100



Greedy Search

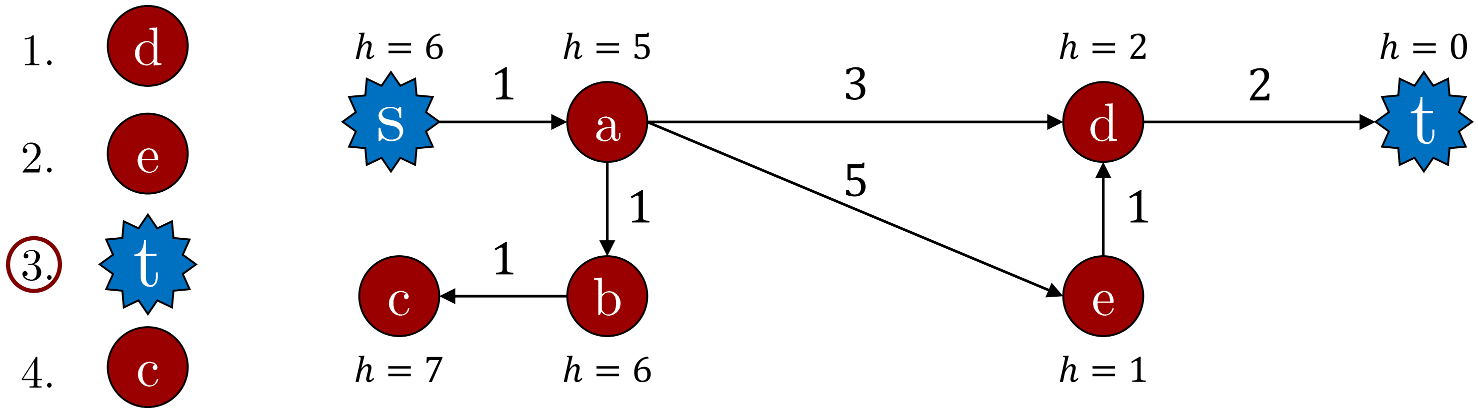
- Strategy: Expand by $h(x)$ = heuristic evaluation of cost from x to goal





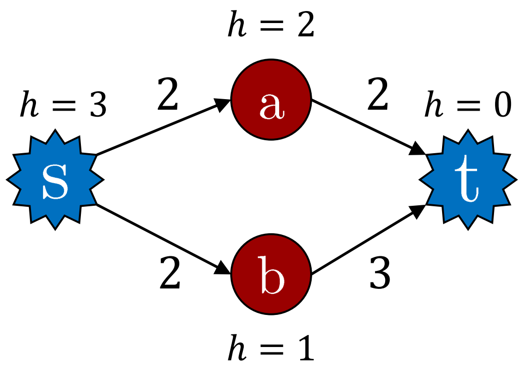
A* Search

- Strategy: Expand by $f(x) = h(x) + g(x)$
- Poll 1: Which node is expanded fourth?



A* Search

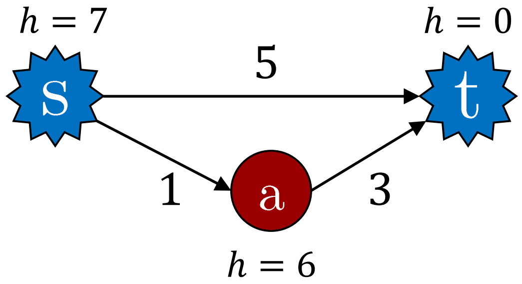
- Should we stop when we discover a goal?



- No: Only stop when we expand a goal

A* Search

- Is A* optimal?



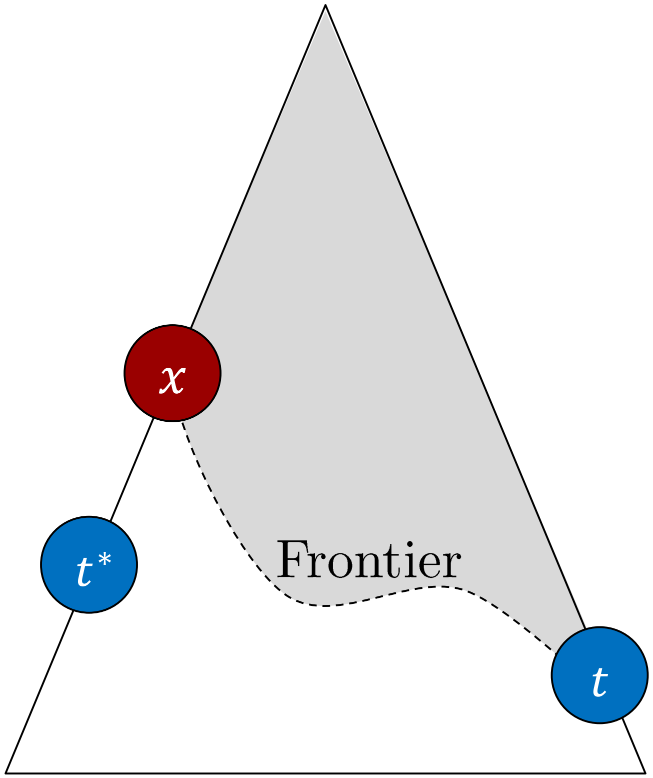
- Good path has pessimistic estimate
- Circumvent this issue by being optimistic!

Admissible Heuristics

- h is admissible if for all nodes x,
$$h(x) \leq h^*(x),$$
where h^* is the cost of the optimal path to a goal
- Example: Aerial distance in the pathfinding example
- Example: $h \equiv 0$

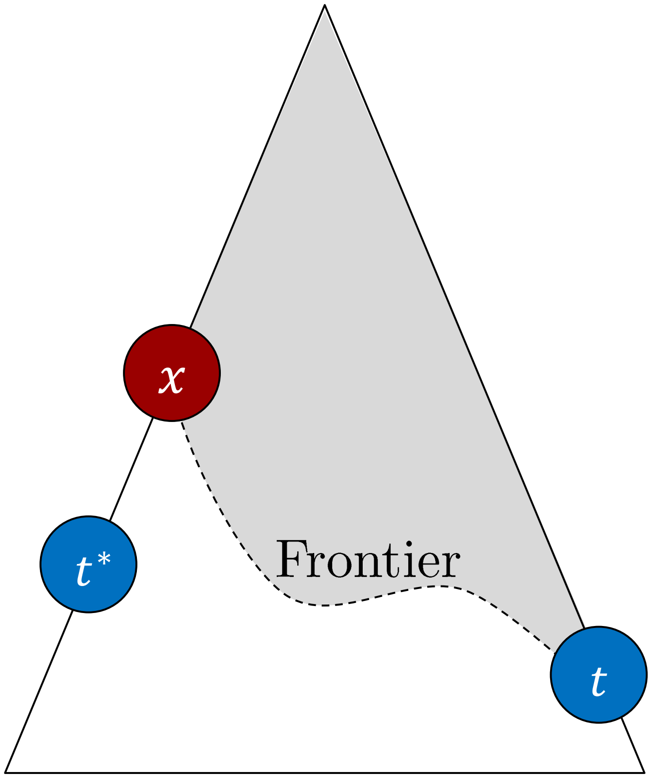
Optimality of A*

- Theorem: A* tree search with an admissible heuristic returns an optimal solution
- Proof:
    - Assume suboptimal goal t is expanded before optimal goal t^*



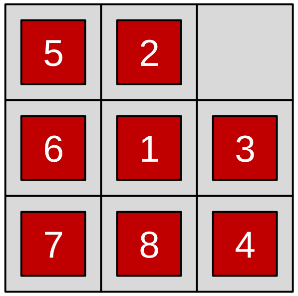
Optimality of A*

- Proof (cont.):
  - There is a node x on the optimal path to t^* that has been discovered but not expanded
  - $f(x) = g(x) + h(x)$
    $\leq g(x) + h^*(x)$
    $= g(t^*) < g(t) = f(t)$
  - x should have been expanded before t! ■

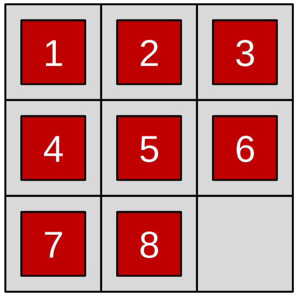


8-PUZZLE HEURISTICS

- h_1: #tiles in wrong position
- h_2: sum of Manhattan distances of tiles from goal
- Poll 2: Which heuristic is admissible?
  1. Only h_1
  2. Only h_2
  3. Both h_1 and h_2
  4. Neither one

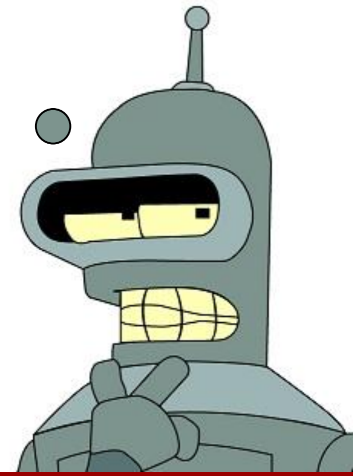


Example state



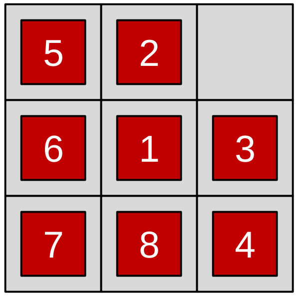
Goal state

Heuristic for designing admissible heuristics: relax the problem!

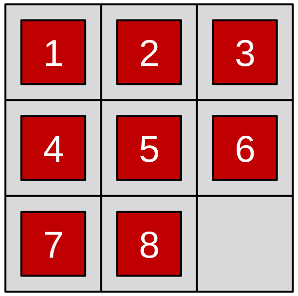


8-PUZZLE HEURISTICS

- h_1: #tiles in wrong position
- h_2: sum of Manhattan distances of tiles from goal
- h dominates h' iff $\forall x, h(x) \geq h'(x)$
- Poll 3: What is the dominance relation between h_1 and h_2?
  1. h_1 dominates h_2
  2. h_2 dominates h_1
  3. h_1 and h_2 are incomparable



Example state



Goal state

8-PUZZLE HEURISTICS

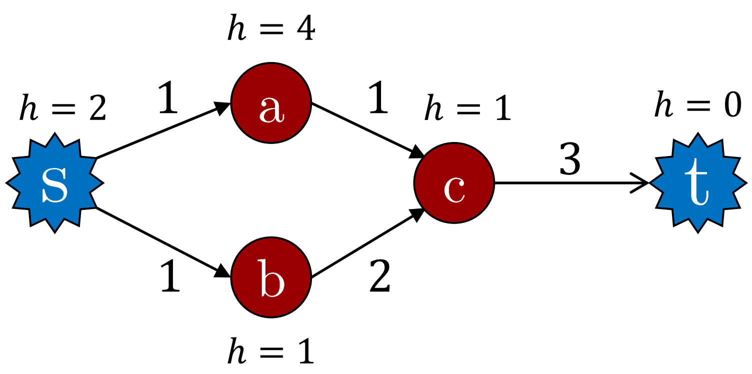
- The following table gives the search cost of A* with the two heuristics, averaged over random 8-puzzles, for various solution lengths

Length	$A^*(h_1)$	$A^*(h_2)$
16	1301	211
18	3056	363
20	7276	676
22	18094	1219
24	39135	1641

- Moral: Good heuristics are crucial!

A* Graph Search

- Graph Search is the same as tree search, but never expand a node twice
- Is optimality of A* under admissible heuristics preserved? No!

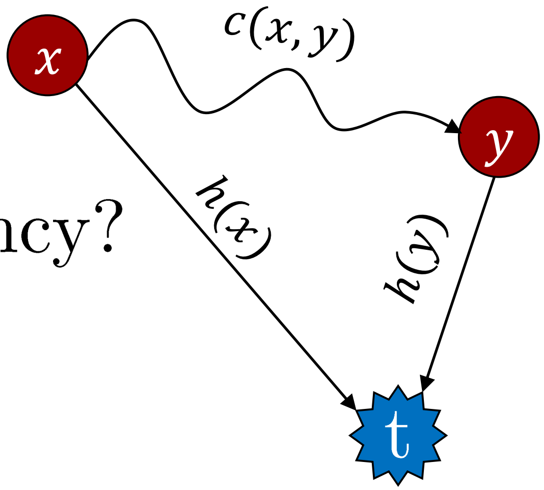


Consistent Heuristics

- $c(x, y)$ = cost of cheapest path between x and y
- h is consistent if for every two nodes x, y,

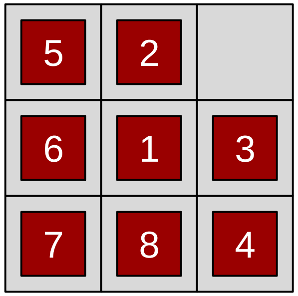
$$h(x) \leq c(x, y) + h(y)$$

- Assume $h(t) = 0$ for each goal t
- Poll 4: What is the relation between admissibility and consistency?
  1. Admissible ⇒ consistent
  2. Consistent ⇒ admissible
  3. They are equivalent
  4. They are incomparable

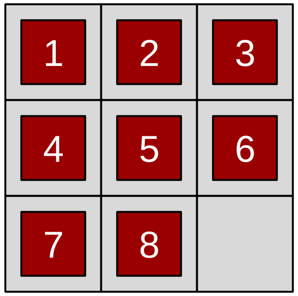


8-puzzle Heuristics, Revisited

- h_1: #tiles in wrong position
- h_2: sum of Manhattan distances of tiles from goal
- Poll 5: Which heuristic is consistent?
  1. Only h_1
  2. Only h_2
  3. Both h_1 and h_2
  4. Neither one

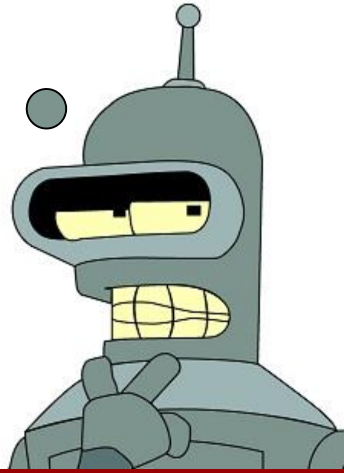


Example state



Goal state

Heuristic for designing consistent heuristics: design an admissible heuristic!



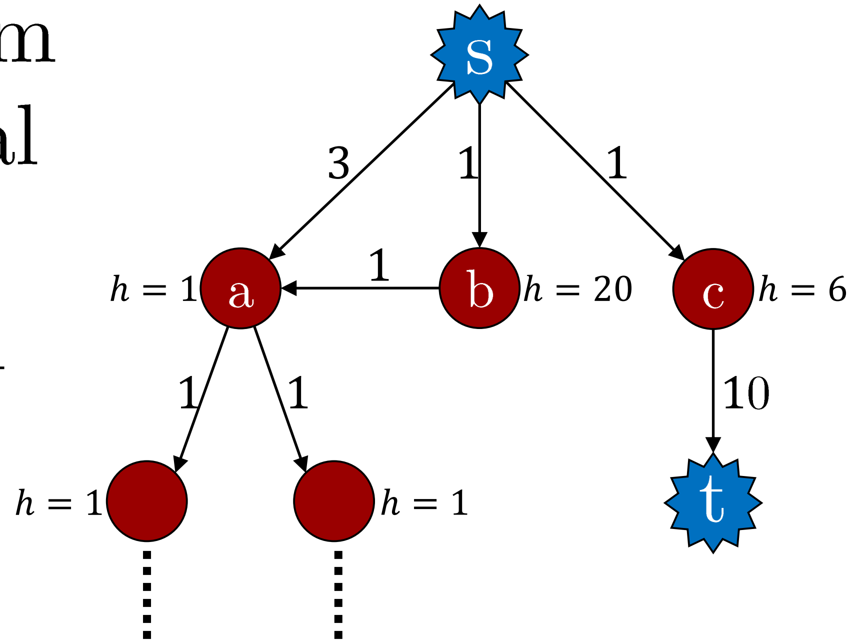
Optimality of A*, Revisited

- Theorem: A* graph search with a consistent heuristic returns an optimal solution
- Proof:*
  - Assume $h(x) \leq c(x, y) + h(y)$
  - Values of $f(x)$ on a path are nondecreasing: if y is the successor of x, $f(x) = g(x) + h(x) \leq g(x) + c(x, y) + h(y) = g(y) + h(y) = f(y)$
  - When A* selects x for expansion, the optimal path to x has been found: otherwise there is a frontier node y on optimal path to x that should be expanded first
  - Nodes expanded in nondecreasing $f(x)$
  - First goal state that is expanded must be optimal ■

* Just for fun

A* IS OPTIMALLY EFFICIENT

- Theorem: Any algorithm that returns the optimal solution given a consistent heuristic will expand all nodes surely expanded by A*
- But this is not the case when the heuristic is only admissible



Alg B: Conduct exhaustive search except for expanding a; then expand a only if it has the potential to sprout cheaper solution

Summary

- Terminology:
  - Search problems
  - Algorithms: tree search, graph search, uniform cost search, greedy, A*
  - Admissible and consistent heuristics
- Big ideas:
  - Properties of the heuristic ⇒ A* optimality
  - Don't be too pessimistic!