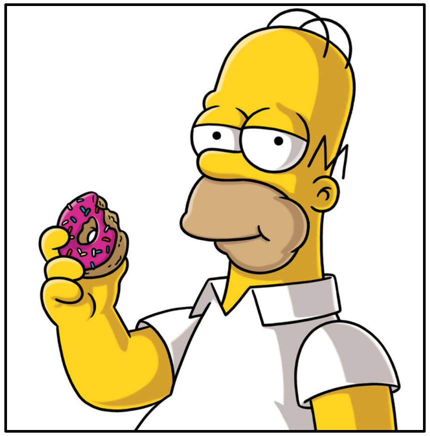
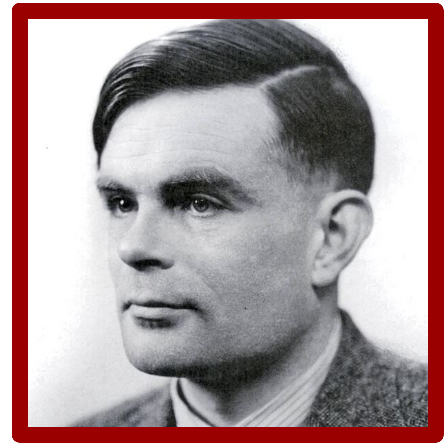
# CMU 15-781

## Lecture 4: Informed Search

Teacher:
Gianni A. Di Caro

# Uninformed vs. Informed





**Uninformed**

Can only generate successors and distinguish goals from non-goals

**Informed**

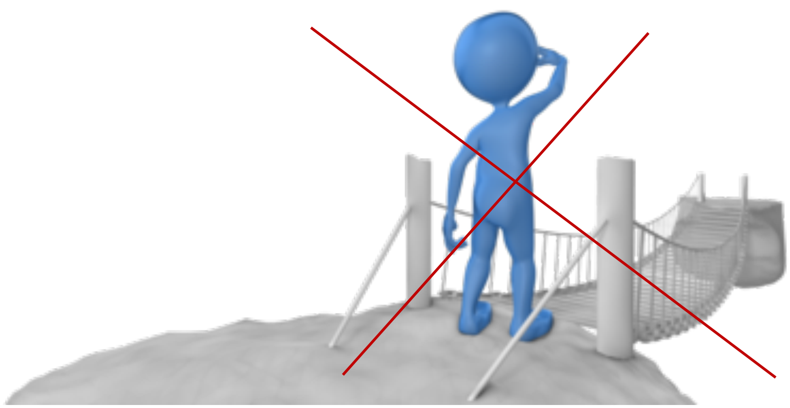
Strategies that can distinguish whether one non-goal is more promising than another

# Reminder: Tree Search

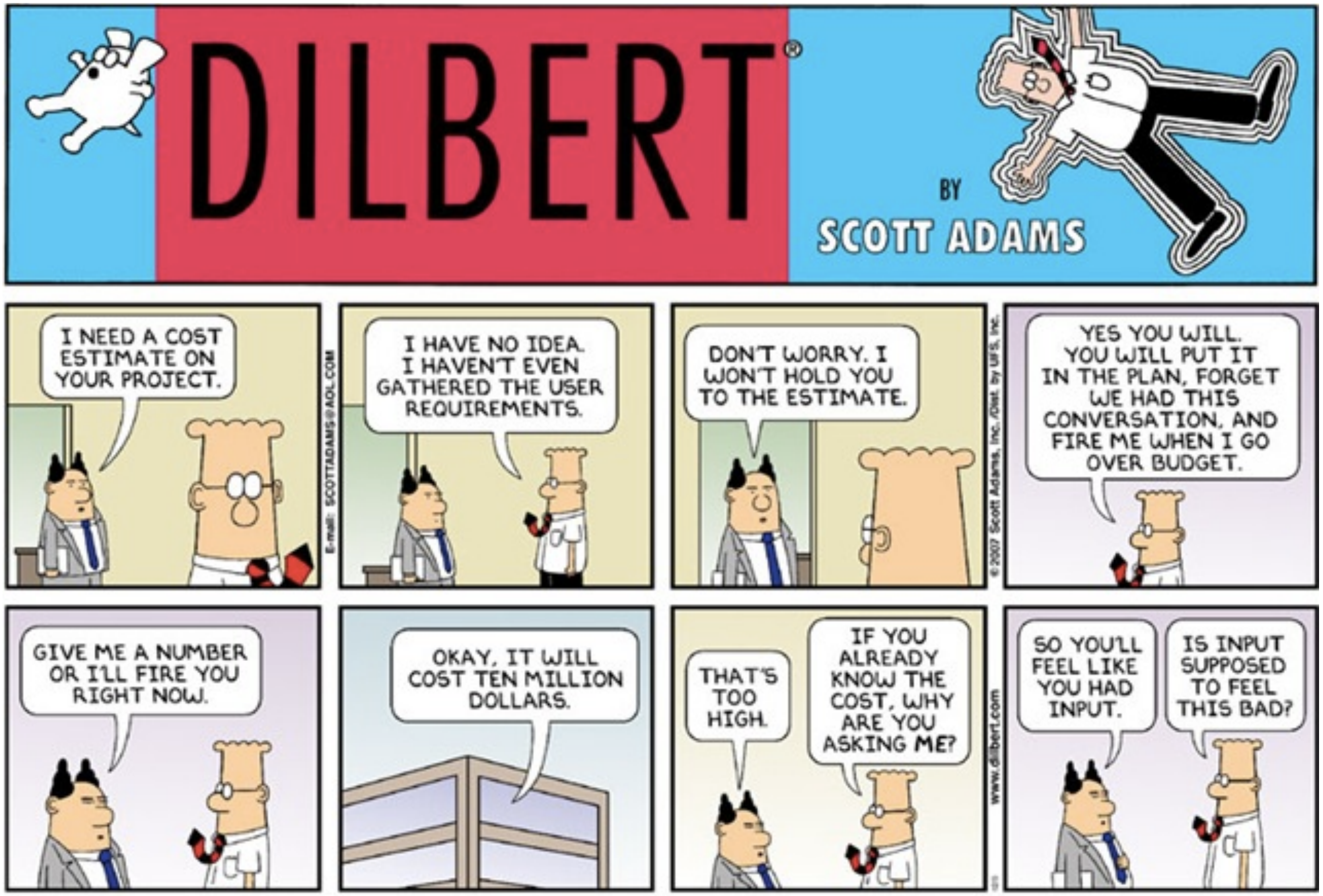
```
function Tree-Search(problem, strategy)
set of frontier nodes contains the start state of problem
loop
    if there are no frontier nodes
        return failure
    choose a frontier node for expansion using strategy
    if the node contains a goal
        return corresponding solution
    else expand the node
            add the resulting nodes to the set of frontier nodes
```

# Recap: Strategies for Uninformed Search

- BFS: Shallowest unexpanded node
- DFS / IDP: Deepest unexpanded node
- UCS: Lowest cost-to-come (from start)
- LS: Local highest-value successor node



# Estimate of Cost-To-Go

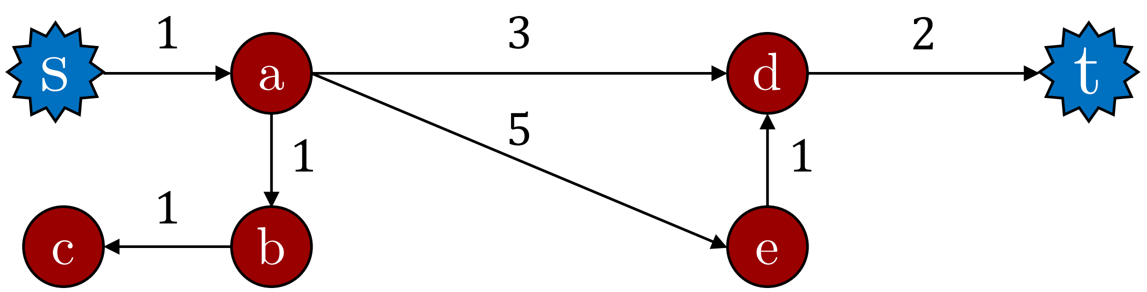


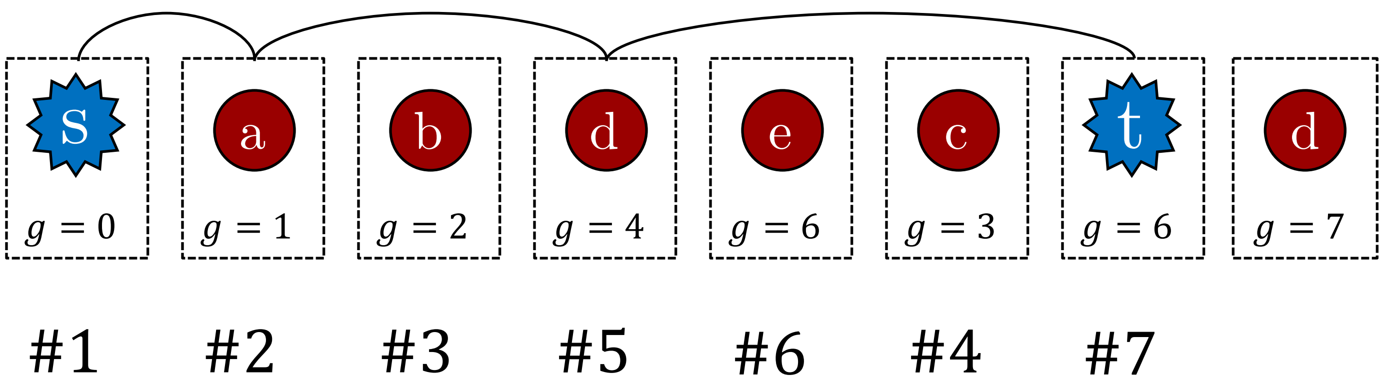
# Strategy: Best-First search

- General strategy template for *informed* search
- A node $n$ in the frontier set is selected for expansion based on an **evaluation function** *f(n)*, which is a cost estimate
- Backward (to-come) and (estimates of) Forward (to-go) costs
- The node with the ***lowest* cost estimate is expanded first**
- Data structure: *Priority queue* using *f(n)* for ordering

# Uniform Cost Search

- Strategy: Expand by $f(x) = g(x) =$ cost-to-come



# UCS vs. Dijkstra's algorithm

DA

- *All* nodes are initially inserted into the PQ (*explicit graph description* is given as input)
- Initial distance ($g(s)$, *cost-to-come*) from start: $d[s]=0$, $d[x \neq s] = \infty$
- The node with the *minimal estimated distance* is selected at each step
- Shortest paths to all other nodes or to a single one

UCS

- Explicit graph description is not given as input
- Nodes are inserted to the PQ *lazily* during the search, based on node expansion choices
- Can naturally handle the presence of *multiple goals*

# Dijkstra's algorithm (1959)

**Input:** Graph $G=(V,E)$

$(\forall x \neq s)\ dist[x] = +\infty$

$dist[s] = 0$

$S = \varnothing$

$Q = V$ // Ordered by $dist[]$

**while** $Q \neq \varnothing$ **do**

    $u = extract\_min(Q)$

    $S = S \cup \{u\}$

    **foreach** $vertex\ v \in Adjacent(u)$ **do**

        $dist[v] = min(dist[v],\ dist[u] + c(u,v))$ // "Relaxation"

    **end do**

**end do**

# Problem description + Heuristic Knowledge

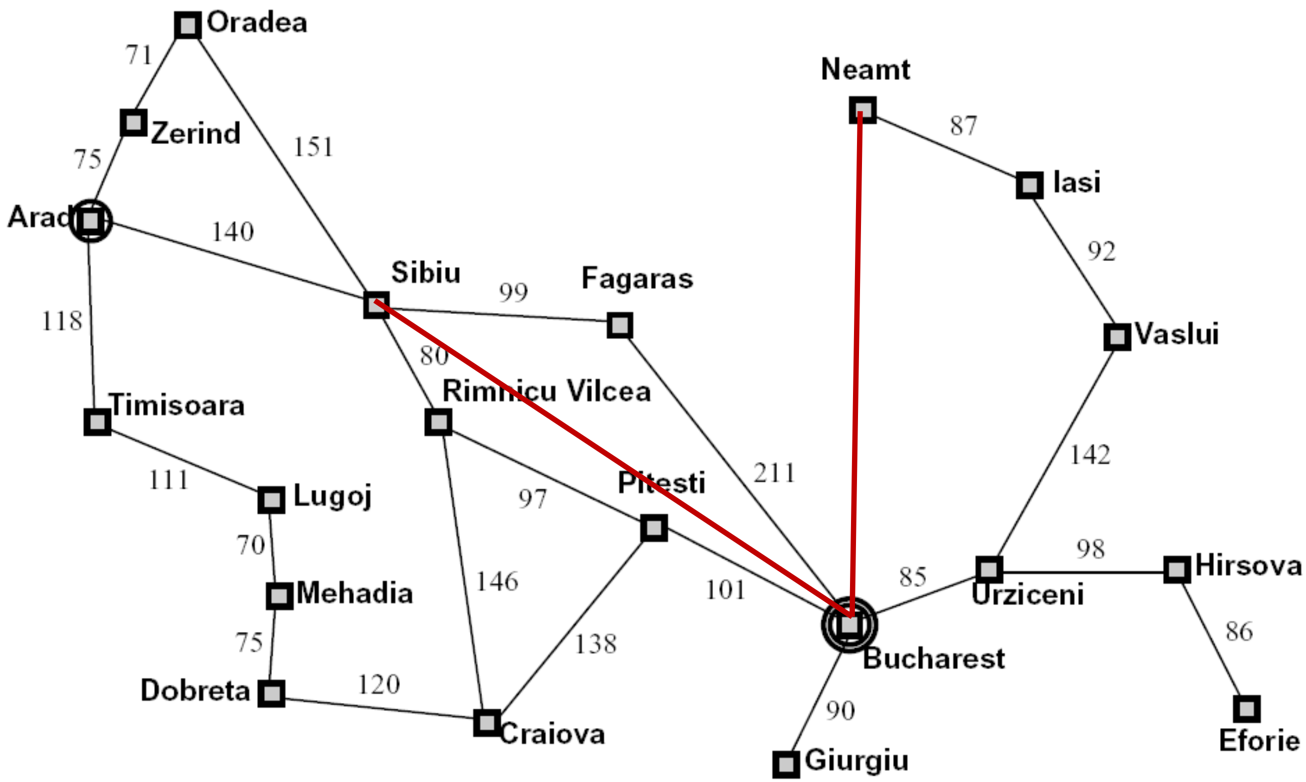
- So far, only *problem description* (successors, step costs) has been used to search the state space
- What about using (also) additional, heuristic knowledge, $h(x)$, to direct state expansion by looking forward?

***Heuristics*** *are rules of thumb, educated guesses, intuitive judgments or, simply, common sense*

The term derives from the ancient Greek *keuriskein*, meaning *serving to find out, or discover*. Archimedes' *Eureka!* means "I have found it!"

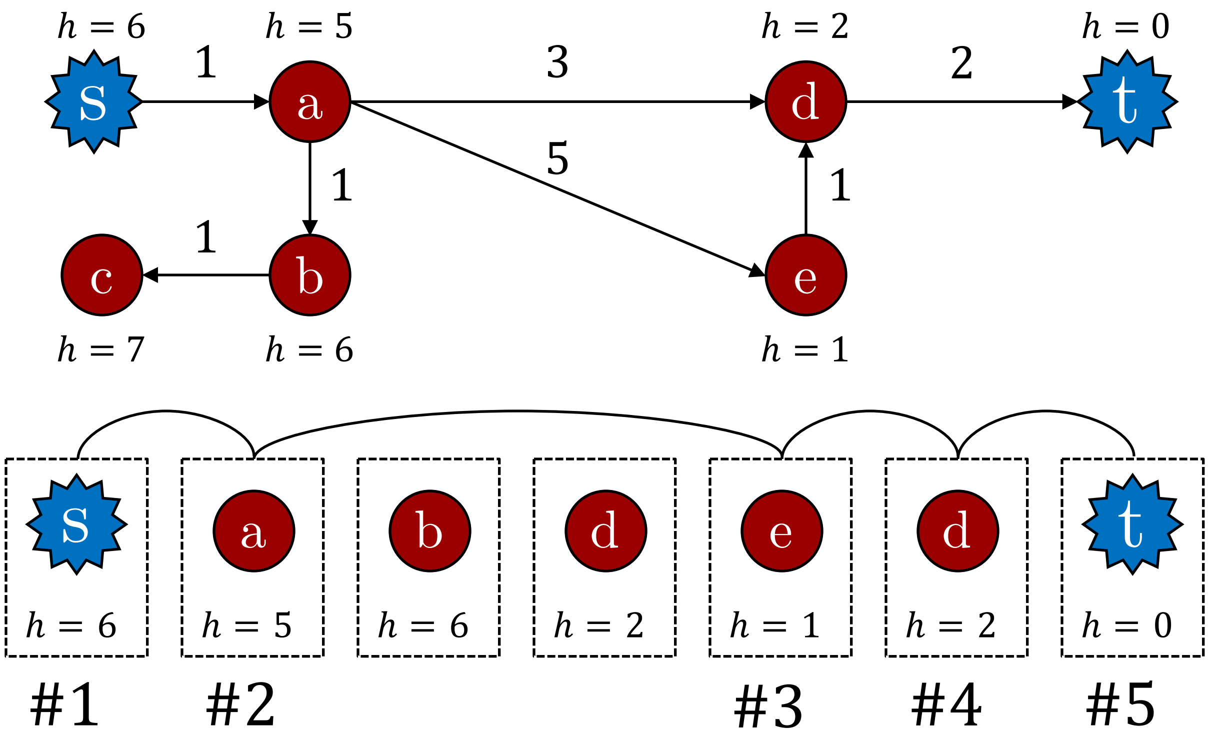

# Example: Heuristic

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



# Greedy Best-First Search

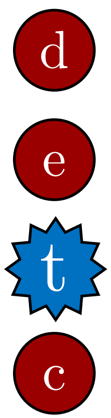
- Strategy: Expand by $h(x)$ = **heuristic evaluation of cost-to-go(al) from $x$**

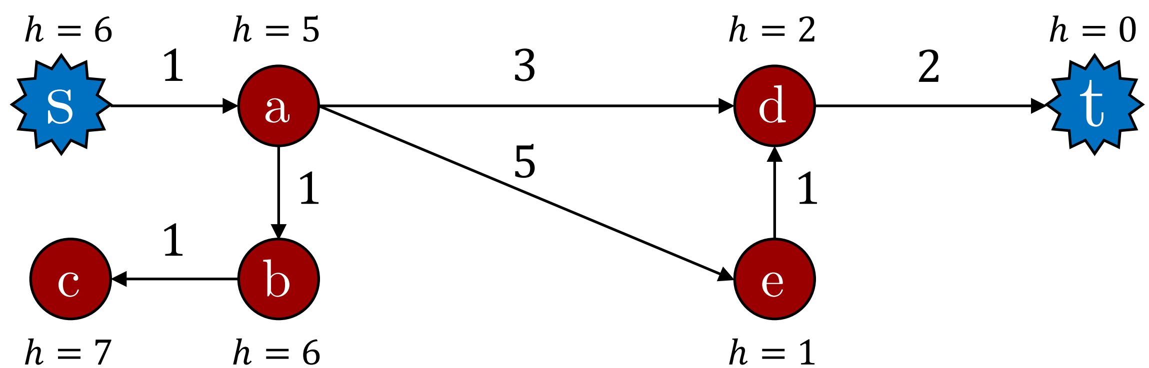


# A* Search (1968)

- Strategy: Combine *cost-to-come* (past) and *heuristic estimate of cost-to-go* (future), expand by $f(x) = h(x) + g(x)$
- Poll 1: Which node is expanded fourth?

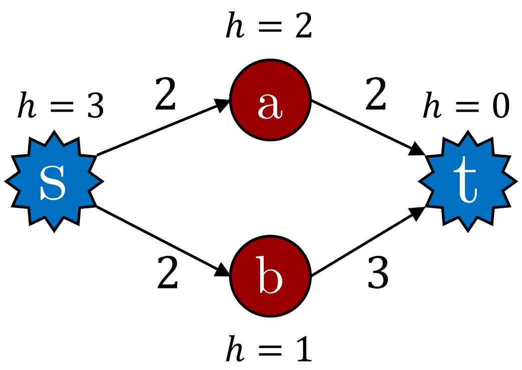
1. d
2. e
3. g
4. c




# A* Search

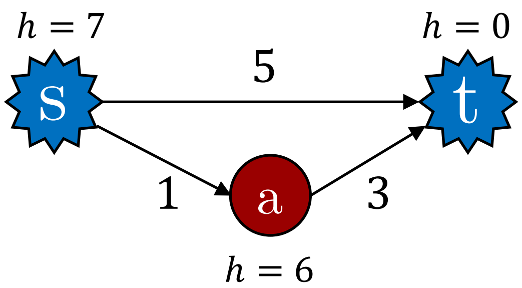
- Should we stop when we discover a goal?




- No: Only stop when we *expand a goal!* (same as in UCS)

# A* SEARCH

- Is A* optimal?




- The good path has a pessimistic estimate
- Circumvent this issue by being optimistic!

# Admissible Heuristics

- $h$ is admissible if for all nodes $x$,

$$h(x) \leq h^*(x),$$

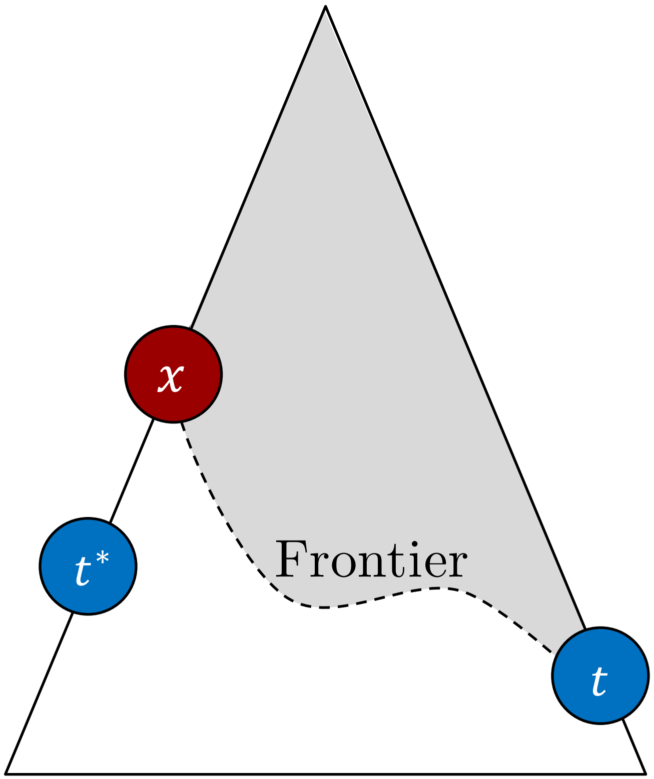
where $h^*$ is the cost of the optimal path to a goal $\rightarrow$

An admissible heuristic is a *lower bound* on real cost

- Example: Aerial distance in the path finding example
- Example: $h \equiv 0$
- $\rightarrow$ ***The tighter the bound, the better***

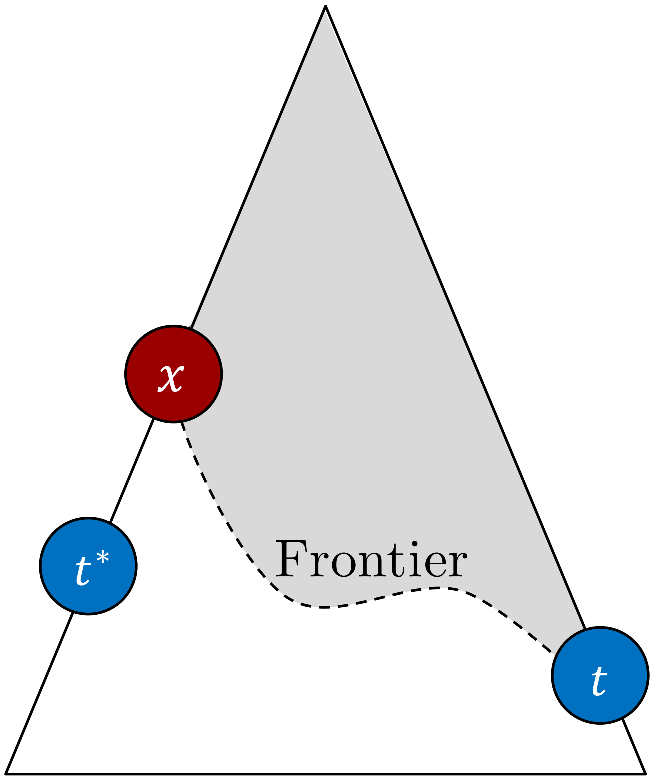
# OPTIMALITY OF A*

- Theorem: A* tree search with an admissible heuristic returns an optimal solution
- Proof (by contradiction):
  - Assume that a suboptimal goal $t$ is expanded before the optimal goal $t^*$



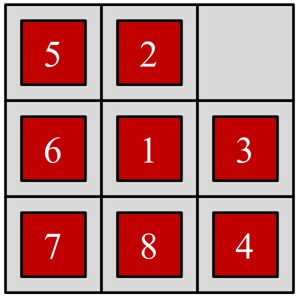
# Optimality of A*

- Proof (cont.):
  - There is a node $x$ in the frontier, on the optimal path to $t^*$ that has been discovered but not expanded yet
  - $f(x) = g(x) + h(x)$
    $\leq g(x) + h^*(x)$
  - But since x is on the optimal path to $t^*$:
    $= g(t^*) < g(t) = f(t)$ ($h(t)$=0)
  - $x$ should have been expanded before $t$! ■



# 8-puzzle Heuristics

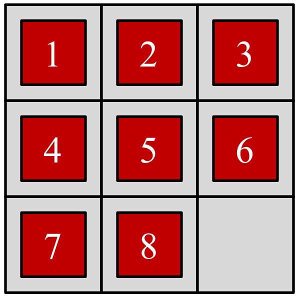
- *Defining a "good" heuristic it's not a trivial task …*
- $h_1$: #tiles in wrong position [$h_1(s) = 5$]
- $h_2$: sum of Manhattan distances of tiles from goal [$h_1(s) = 2+0+1+3+2+2+0+0=10$]



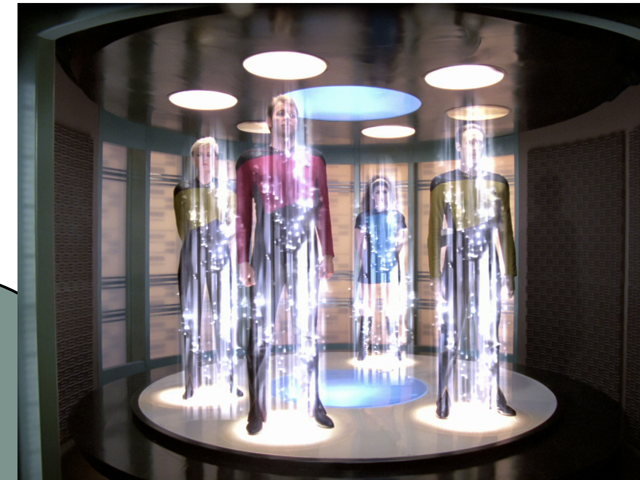
Example state

- Poll 2: Which heuristic is *admissible*?

1. Only $h_1$
2. Only $h_2$
3. Both $h_1$ and $h_2$
4. Neither one

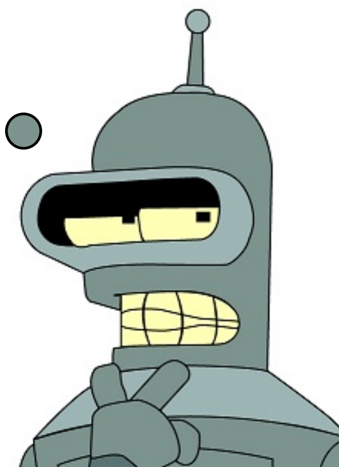


Goal state



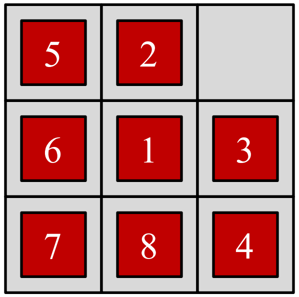
Heuristic for designing admissible heuristics: relax the problem!

*Relaxation:*
Remove functional / domain constraints
$\rightarrow$ Add "forbidden" moves



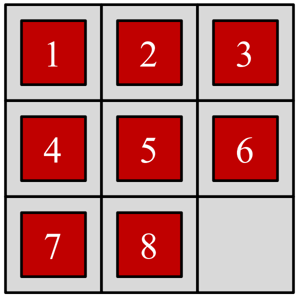
# Dominance

- $h_1$: #tiles in wrong position
- $h_2$: sum of Manhattan distances of tiles from goal
- $h$ *dominates* $h'$ iff $\forall x, h(x) \geq h'(x)$
  ($h$ is consistently a tighter bound wrt $h'$)



Example state

- 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



Goal state

# 8-PUZZLE HEURISTICS

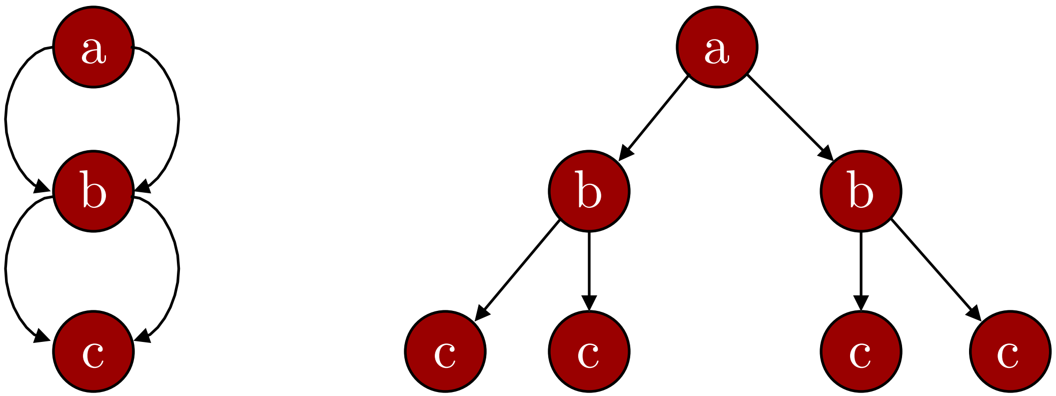
- The following table gives the search cost of A* with the two heuristics, averaged over random 8-puzzles, for various solution lengths

| 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!

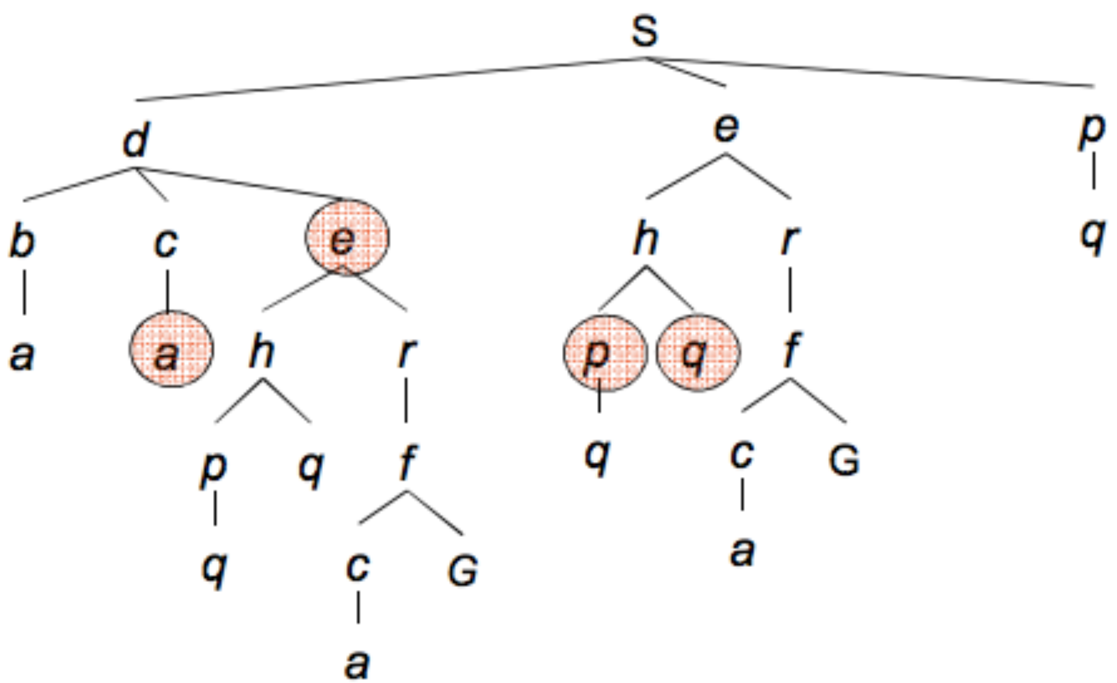
# Graph Search

- In tree search, expanding the same state multiple times can cause exponentially more work



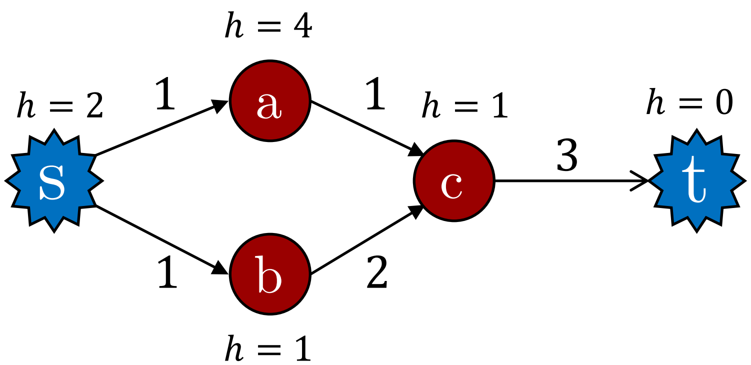
# Graph Search

- Same as tree search, but never *expand* a node twice
- E.g., in BFS expanding the circled nodes is not necessary
- Set of already expanded nodes has to be stored in *memory*

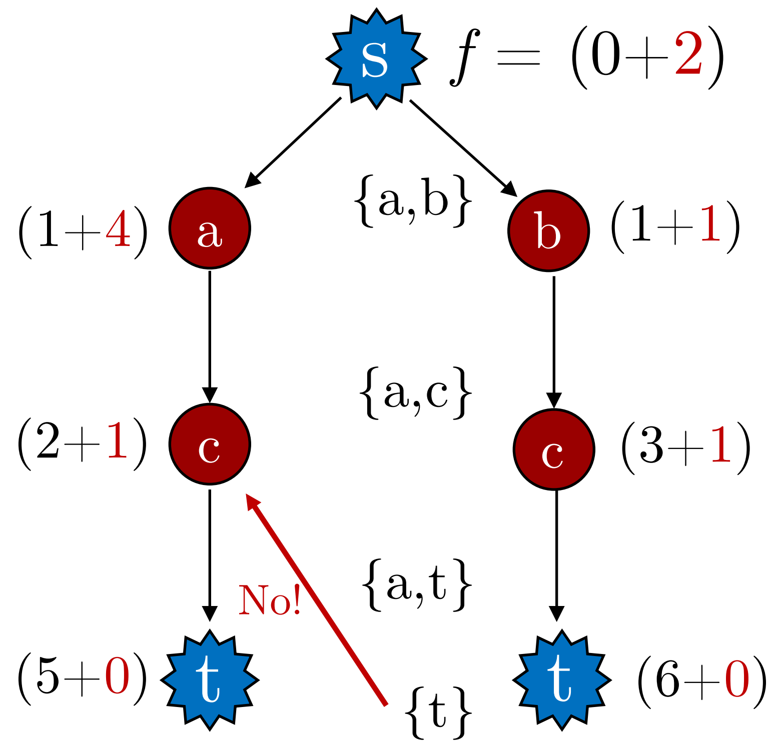



# Graph Search and Optimality

- Is *optimality* of A* under admissible heuristics preserved? No!




Search tree

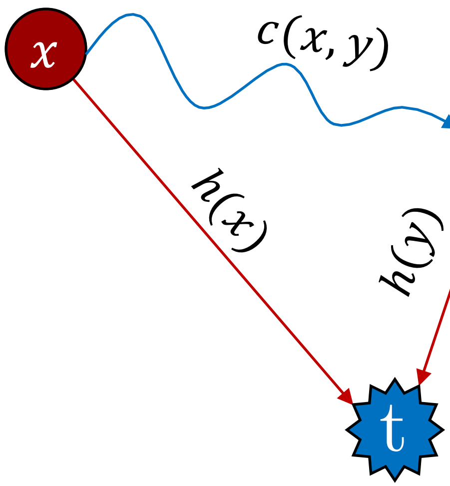


# Consistent Heuristics

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

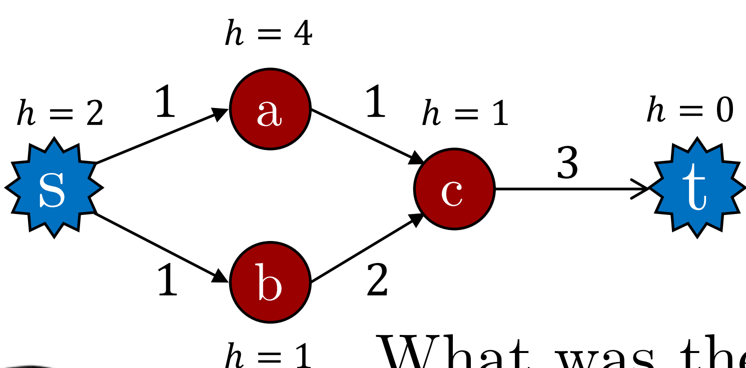
$$h(x) \leq c(x, y) + h(y)$$

- *Triangle inequality*
- Necessary for graph search optimality

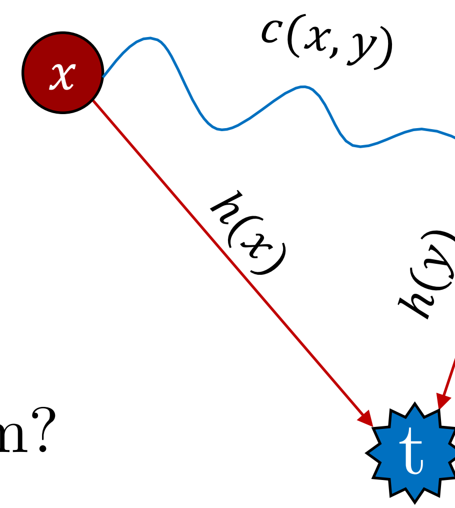



# CONSISTENT HEURISTICS

- *"Consistency"*: The estimated distance to the goal from $x$ cannot be reduced by moving to a different state $y$ and adding the estimate of the distance to the goal from $y$ to the cost of reaching $y$ from $x$
- $c(x,y) \geq h(x) - h(y)$ → The real cost is higher than the cost implied by the heuristics




What was the problem?



# Consistent Heuristics

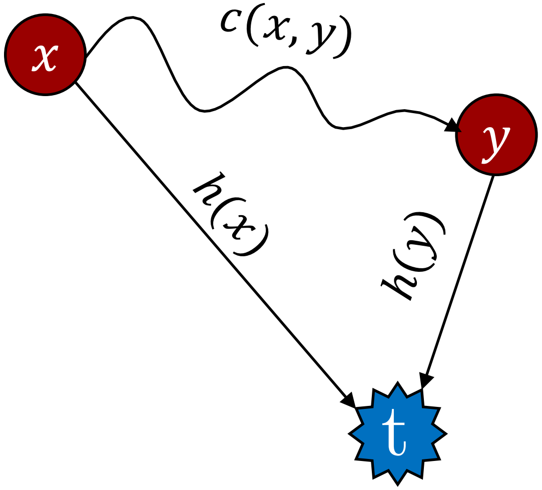
- $h$ is consistent if for every two nodes $x, y$,

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

- Poll 4: What is the relation between admissibility and consistency?

Assuming $h(t) = 0$ at goals $t$

1. Admissible $\Rightarrow$ consistent
2. Consistent $\Rightarrow$ admissible
3. They are equivalent
4. They are incomparable



# Consistency → Monotonicity

- Lemma: If $h(x)$ is consistent, then the values of the cost function $f(x)$ along any path are *nondecreasing*
- Proof:
  - If $y$ is a successor of $x$: $g(y) = g(x) + c(x,y)$
  - By consistency: $f(y) = g(y) + h(y)$

$$= g(x) + c(x,y) + h(y)$$

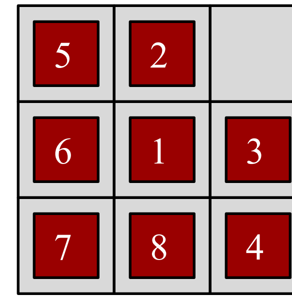
$$\geq g(x) + h(x) = f(x) \blacksquare$$

In moving from a state to its neighbor, (a consistent) $h$ must not decrease more than the cost of the edge that connects them.

Consistency is a property of $h(x)$, monotonicity is a property of $f(x)$

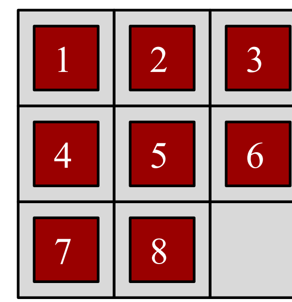
# 8-PUZZLE HEURISTICS, REVISITED

- $h_1$: #tiles in wrong position
- $h_2$: sum of Manhattan distances of tiles from goal



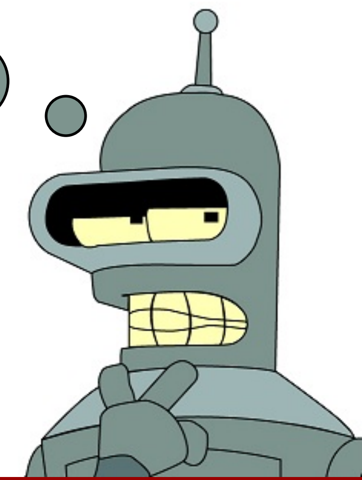

Example state

- Poll 5: Which heuristic is consistent?
  1. Only $h_1$
  2. Only $h_2$
  3. Both $h_1$ and $h_2$
  4. Neither one



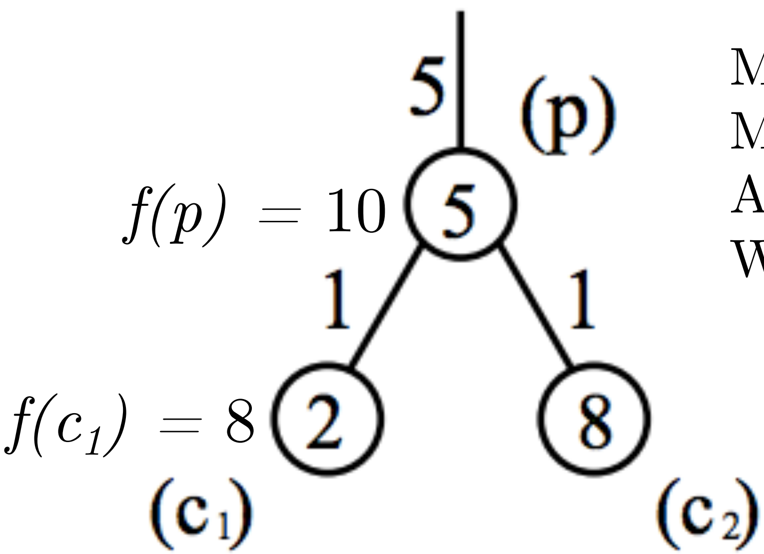

Goal state

Heuristic for designing consistent heuristics: design an admissible heuristic!

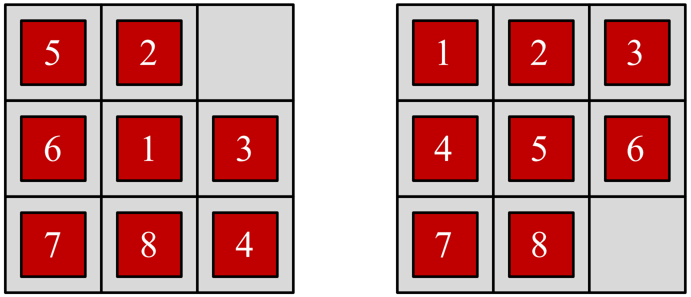


# Admissible but Inconsistent Heuristics?

- Keep the LB property, but violate monotonicity
- Inconsistent for at least one pair of states

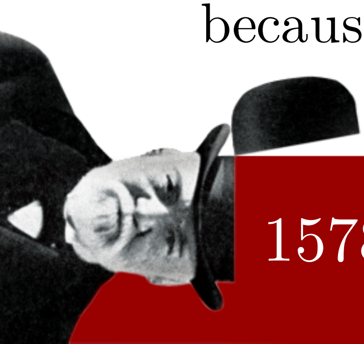



Manhattan distance for set {1,2,3,4}
Manhattan distance for set {5,6,7,8}
At each step, choose at random which set
What is the relation between heuristic estimates?



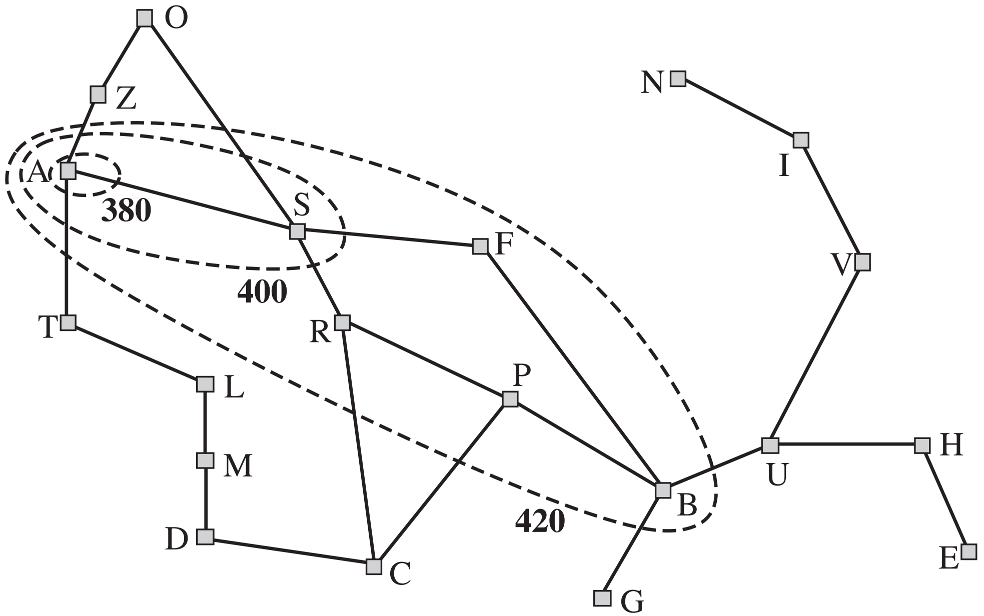
# OPTIMALITY OF A*, REVISITED

- Theorem: A* graph search with a consistent heuristic returns an optimal solution
- Proof:
  - Whenever A* selects a state $x$ for expansion, the optimal path to $x$ has been found. Otherwise, there would a frontier node $y$ (*separation property*) on the optimal path from start to $x$ that should be expanded first because $f$ is non decreasing along any path (monotonicity)
  - *The first goal state* $x^*$ selected for expansion must be optimal, because $f(x^*)$ is the true (optimal) cost for goal nodes ($h(x^*) = 0$), and any other later goal node would be at least as expensive because of $f$ monotonicity

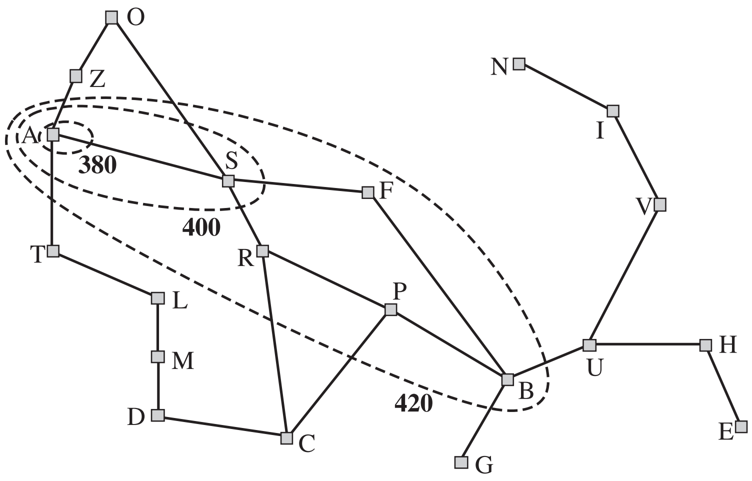


# Contours in the State Space

- *f*-costs are nondecreasing along any path from the start → Contours (isolines) in the state-space, like in topographic maps
- A* search fans out adding nodes in circoncentric bands of increasing *f*-cost
- The tight the LB bounds are, the more the bands will stretch toward the goal state
- Bands using UCS?



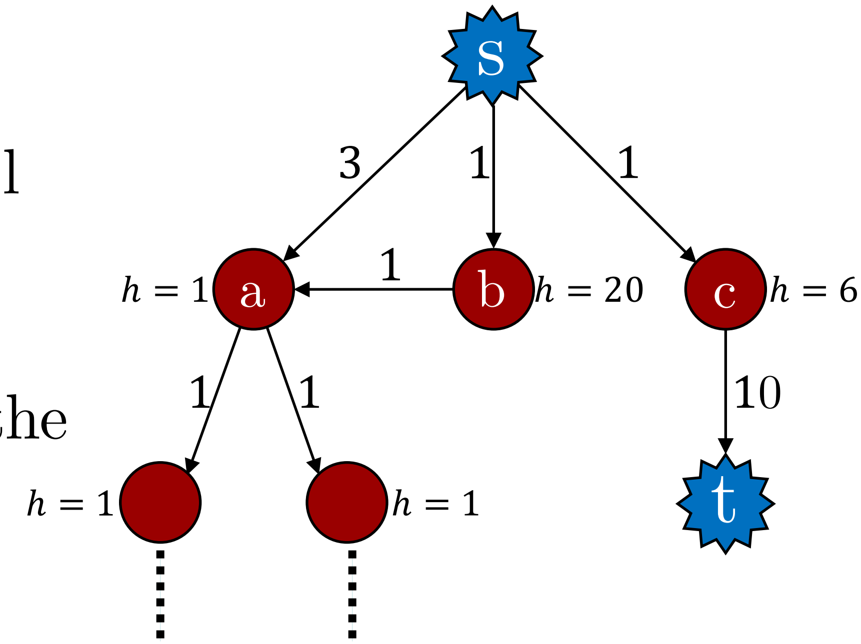
# Pruning and Completeness




- If $C^*$ is the cost of the optimal path, $A^*$ expands *all nodes* with $f(x) < C^*$, and no nodes with $f(x) > C^*$ *(automatic pruning)*
- $A^*$ might expand some of the nodes on the goal contour, where $f(x) = C^*$, before selecting the goal node
- **Completeness?** Yes, if only a finitely many nodes with cost less or equal to C* are present ($b$ is finite and all step costs are $\varepsilon > 0$)

# A* IS OPTIMALLY EFFICIENT

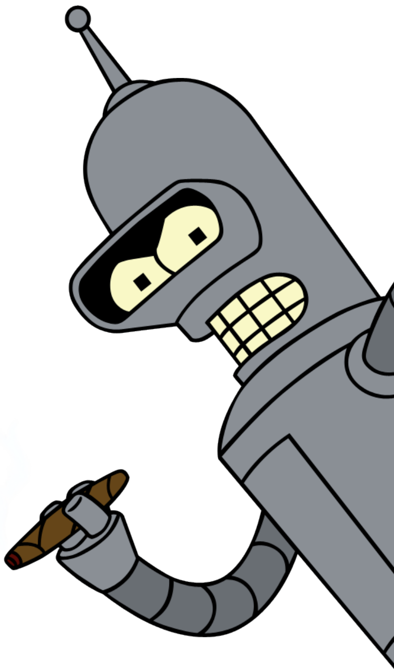
- Theorem: Any algorithm that returns the optimal solution given a consistent heuristic will expand all nodes surely expanded by A*
- But this is not the case when the heuristic is only admissible




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, best-first search, uniform cost search, greedy, A*
  - Admissible and consistent heuristics
- Big ideas:
  - Properties of the heuristic ⇒ A* optimality
  - Don't be too pessimistic!
  - Be consistent!



# Projects

- Proposals to be submitted by October 24
- 1-2 pages stating:
  - Motivation
  - Goals
  - Work plan
- 40 hours of work
- Can be done in pairs
- Poster presentation only, at the end of the semester