Artificial Intelligence

CSC 665

tyler dae devlin

Search III

8.31.2023

Recap

Administrivia

- Late submissions open until Friday night for **homework 0**
- Aiming to get **grading** done by next class
- **Homework 1** due Monday 9/11
- **Office hours** Tuesdays before class, Thursdays after class

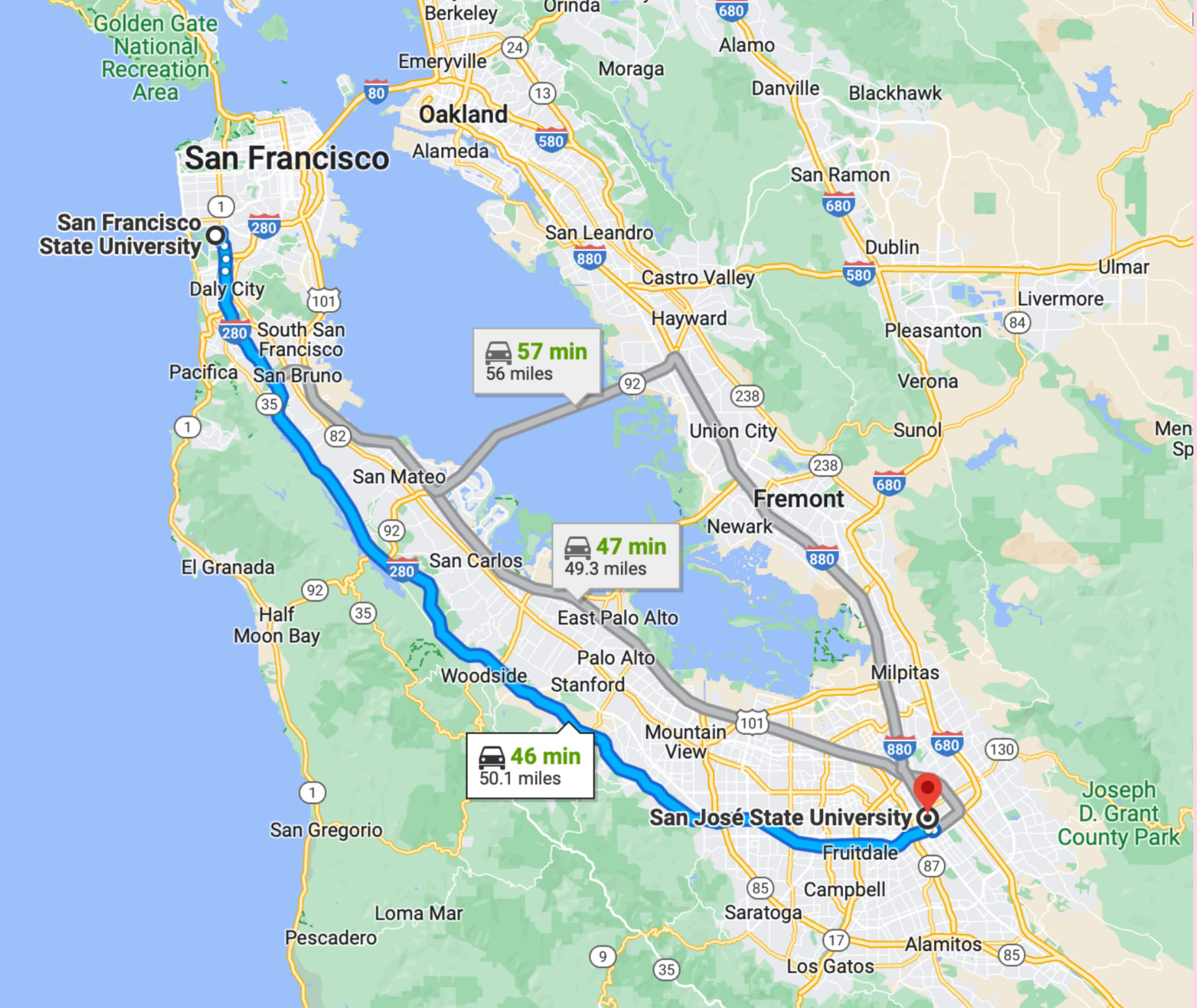
- **Search:** make decisions by looking ahead
- **Logic:** deduce new facts from existing facts
- **Constraints:** find a way to satisfy a given specification
- **Probability:** reason quantitatively about uncertainty
- **Learning:** make future predictions from past observations

Search

Modeling: start state, actions, cost, transition model, goal test

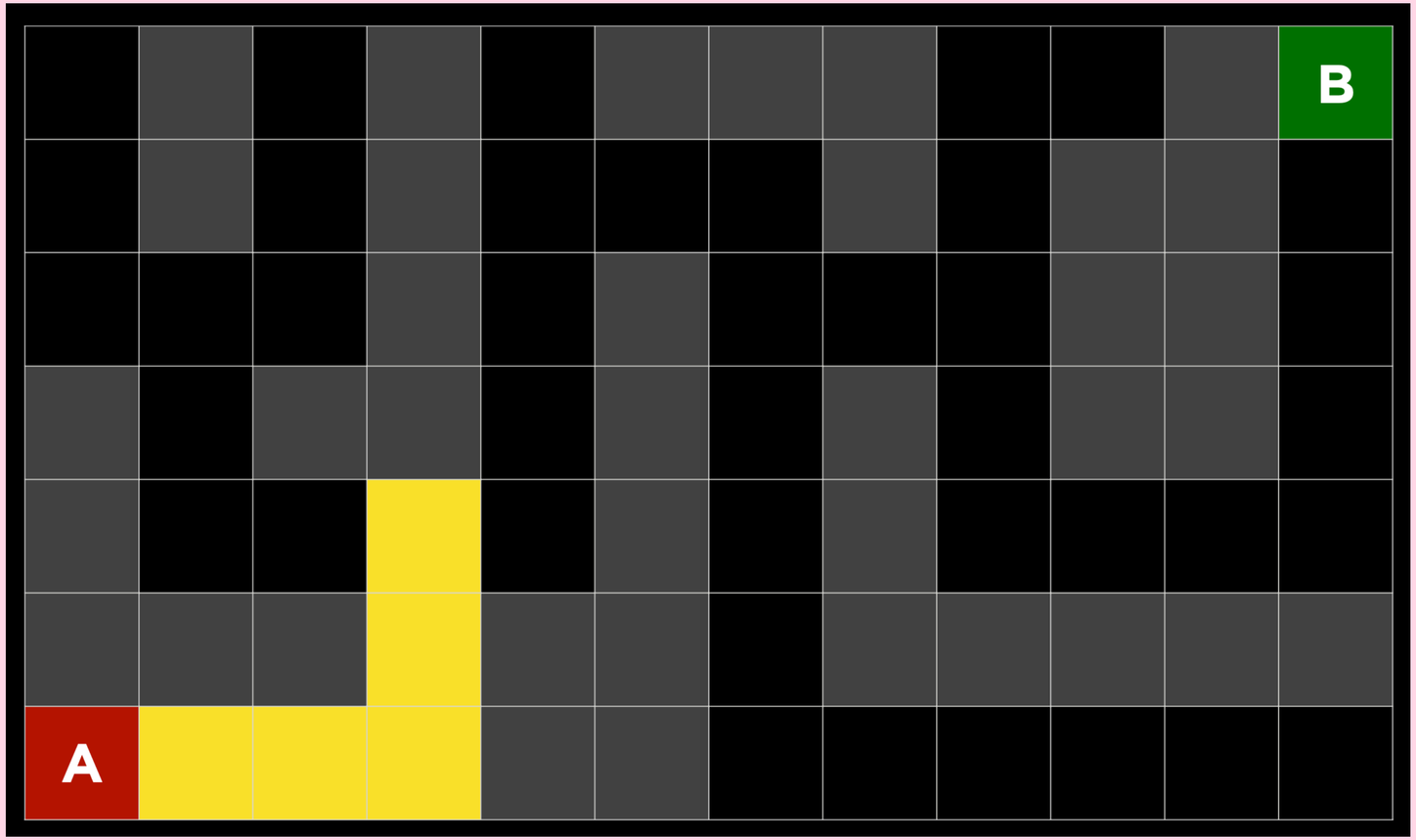
Inference: backtracking, DFS, BFS, UCS — *all uninformed search algorithms*

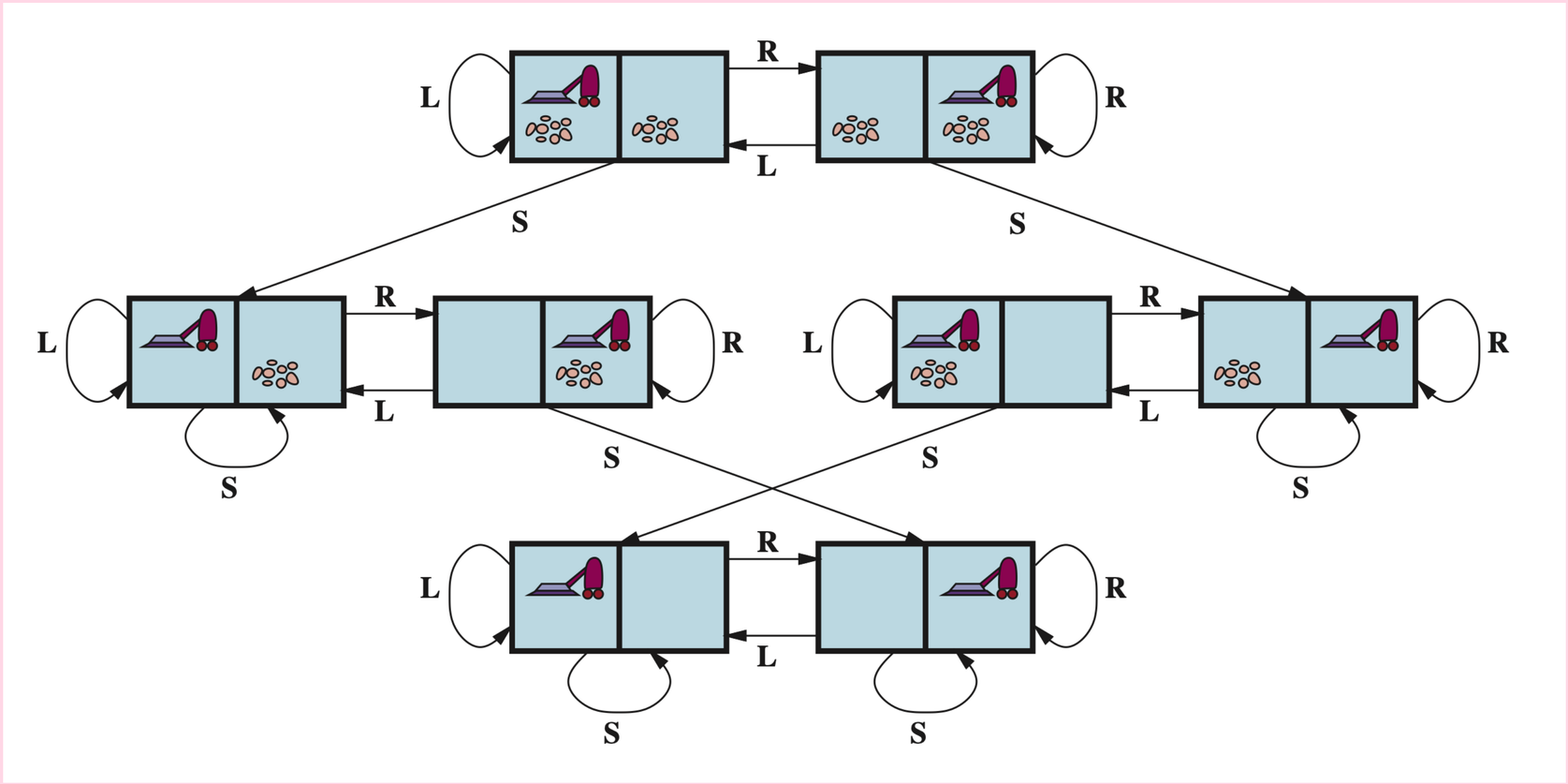
Informed search



Probably not good to start driving toward Marin

Probably not good
to turn left



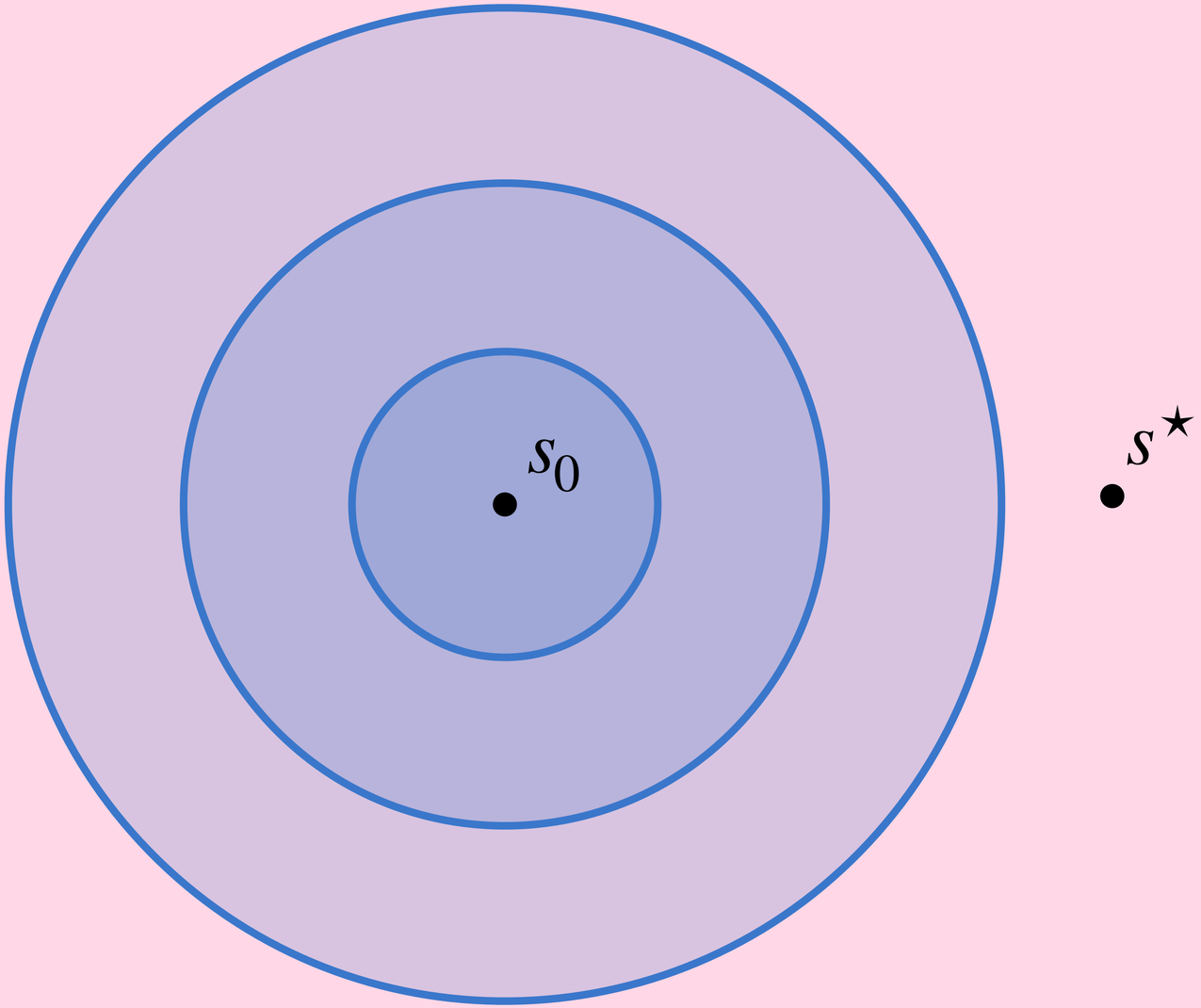


Starting in the upper left state, probably not good to move right before sucking

How do we know?

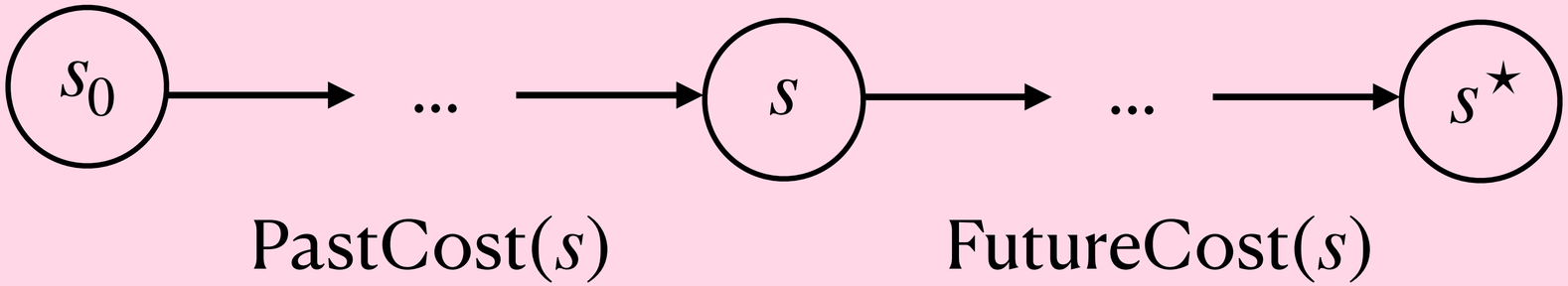
Uniform cost frontier is a good idea.

But why bother searching in this direction?



Heuristic functions

Consider getting from s_0 to $s^\star$ on a path through s.



UCS and BFS work by maintaining a frontier of **uniform PastCost**.

FutureCost is unknown, otherwise we could immediately find an optimal solution.

But we can **estimate** FutureCost(s) with a simple **heuristic** $h(s)$.

Naïve Idea

- If we had access to FutureCost, then an optimal algorithm is to always expand the node that minimizes FutureCost.
- If all we have is an estimate h, then why not pick the node that minimizes h?
- This is called **greedy search**.

[greedy search examples]

A* Search

UCS

- Maintains a frontier of uniform PastCost
- **Correct** but **slow**.

Greedy search

- Chooses the node that minimizes h
- **Incorrect** but **potentially fast**.

A*

- Maintains a frontier of uniform PastCost + h
- **Sometimes correct** and **potentially fast**.

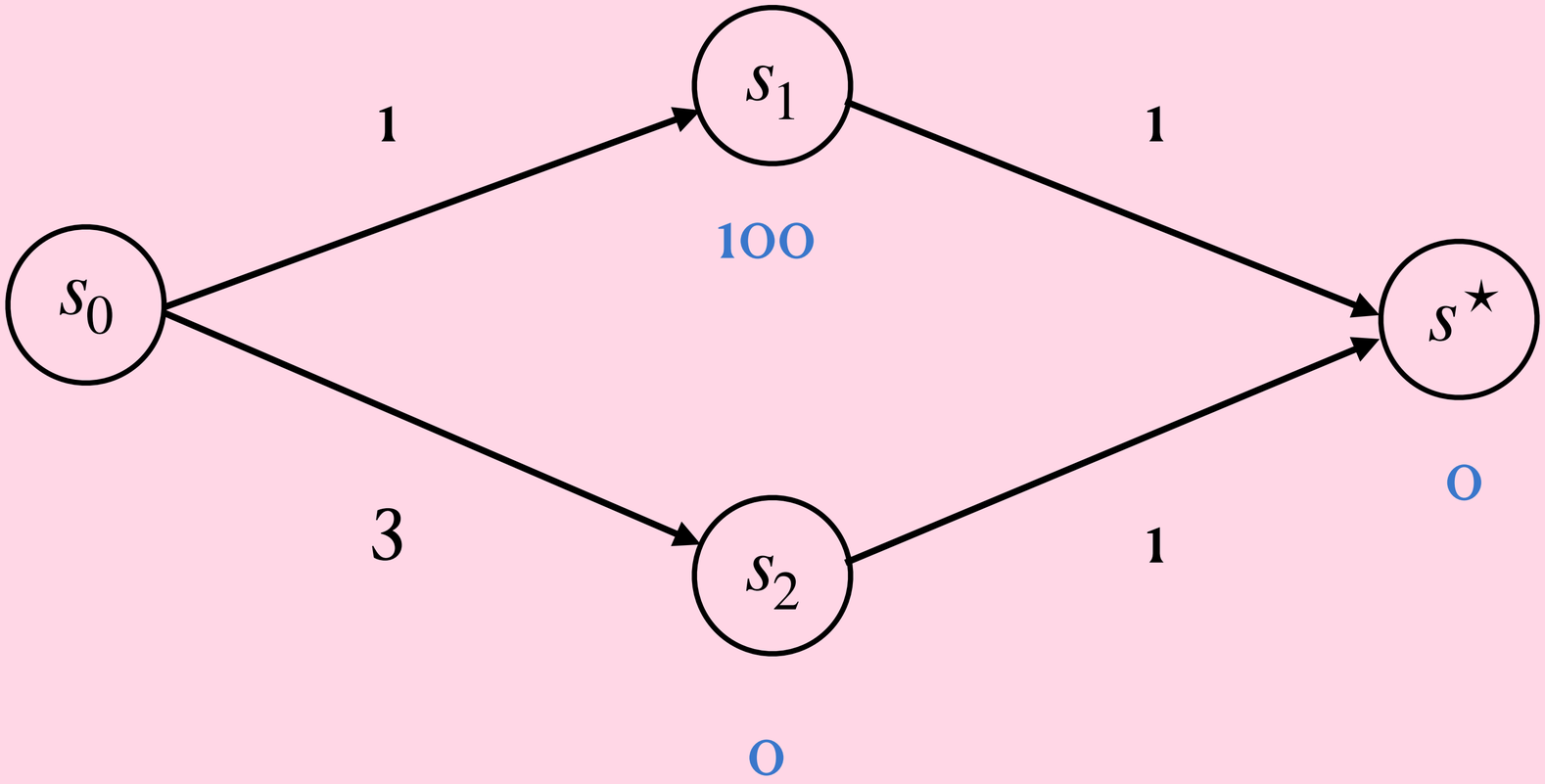
A* vs. Greedy

Problem: short-term greediness can get you into long-term trouble (true for all greedy algorithms in computer science and in life).

Key insight: computing PastCost is easy (just accumulate edge costs), and helps us realize when a prior greedy decision has led us astray.

[A maze example]*

A* can be wrong



Action costs

FutureCost

When is A* correct?

Definition: A heuristic is *admissible* if it **never overestimates** the cost to the goal. That is, $h(s) \leq \text{FutureCost}(s)$ for every $s \in S$.

Theorem: A* with heuristic function h is correct if h is admissible.

When is A* correct?

Proof: For contradiction, assume A* returns a path with cost C, but the optimal path has cost $C^\star < C$. Then there is a node s on the optimal path that was not expanded by A*. Focusing on this node,

$$\begin{aligned} C^\star &< \text{PastCost}(s) + h(s) \\ &\leq \text{PastCost}(s) + \text{FutureCost}(s) \\ &= C^\star \end{aligned}$$

This is a contradiction. Thus, A* returns an optimal path.

How fast is A*?

Theorem: A* explores all states s satisfying $\text{PastCost}(s) \leq \text{PastCost}(s^\star) - h(s)$.

Proof: A* explores all states s satisfying $\text{PastCost}(s) + h(s) \leq \text{PastCost}(s^\star)$

Takeaway: Want h to be as large as possible, because this means we explore fewer states. But can't be too large or we lose admissibility (and thus correctness)!

Problem Relaxation

- How to choose h?
- Create a **"relaxed"** version of the problem by **removing constraints**.
- Set the **estimate** h in the original problem to be the **exact** FutureCost in the relaxed problem.
- *Such a heuristic is guaranteed to be **admissible**.*
- **Example:** for mazes, remove the constraint that you can't travel through walls. Then FutureCost(s) is simply the Manhattan distance from s to $s^\star$.
- What is the relaxation for Google maps? for the Roomba?