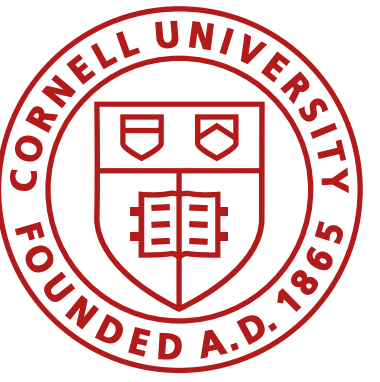


# Markov, Bayes Filter I

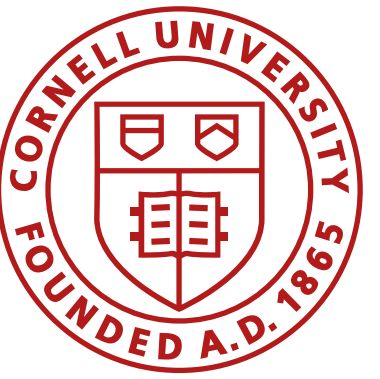
Fast Robots, ECE4160/5160, MAE 4190/5190

E. Farrell Helbling, 3/24/26

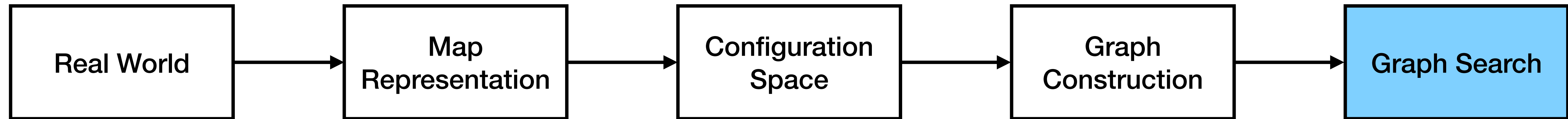


# Class Action Items

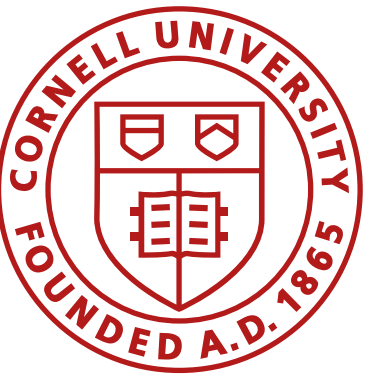
- Lab 8: Stunts!
  - We have two sticky mats set up in the lab for everyone who wants to do the flip.
- Regrade requests: please leave a comment on your grade in canvas! I do check for these and we are happy to review anything that you think needs to be revised.



# Modeling path planning as a graph search problem



- Uninformed Searches
  - Breadth First
  - Depth First
  - Dijkstra's (LCF)
- Informed Searches
  - Greedy
  - $A^*$

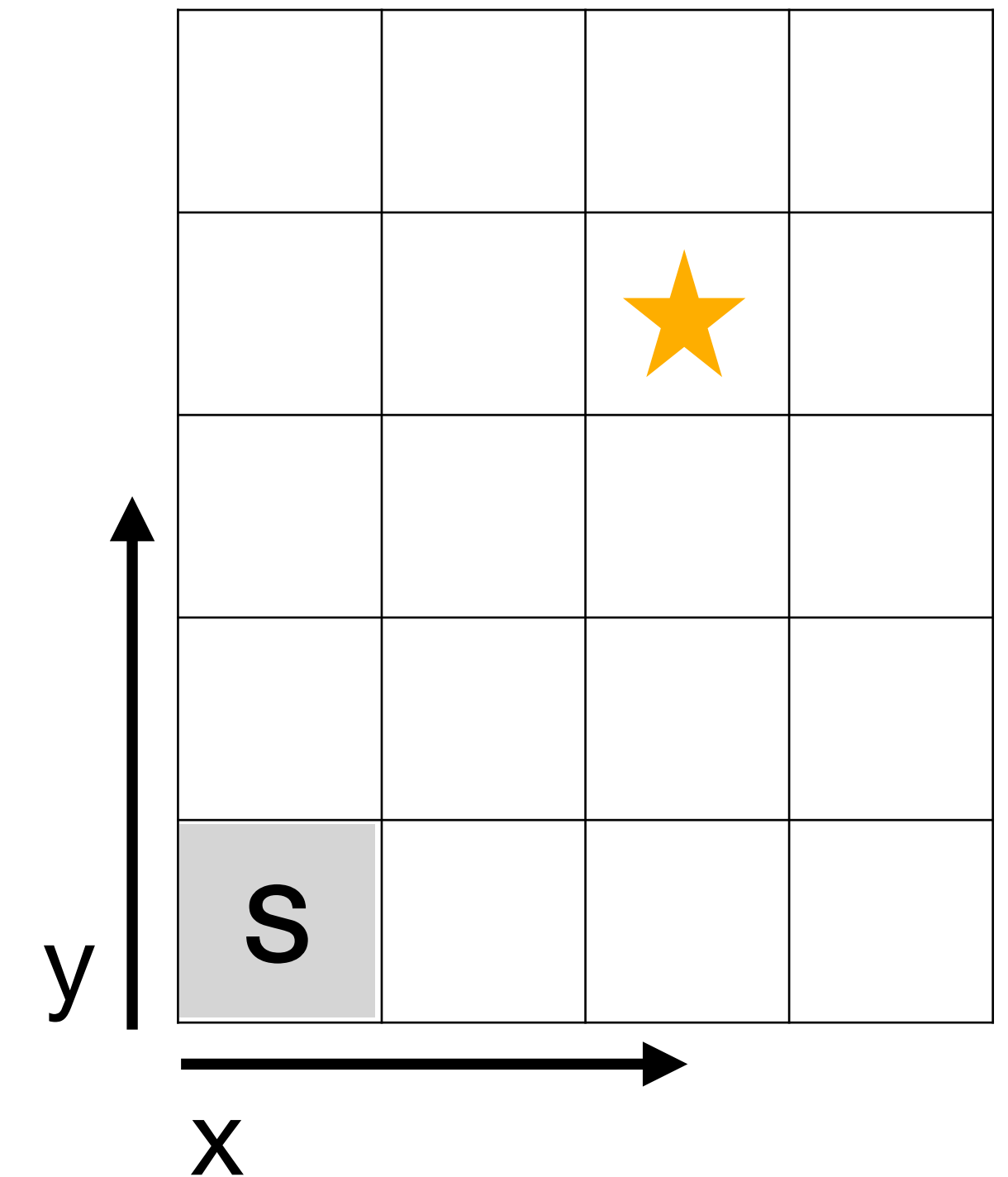
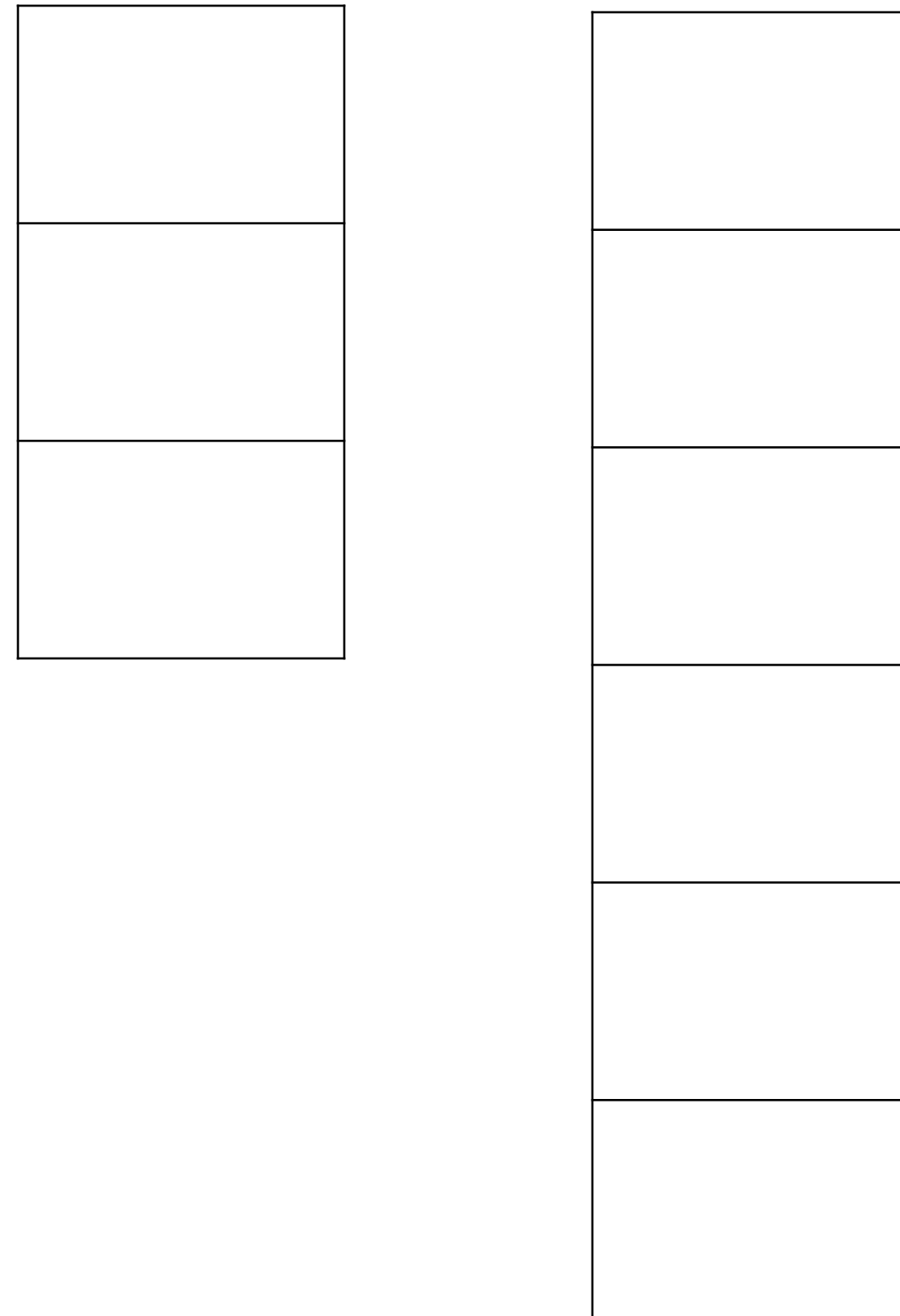


# Uninformed Algorithms, General

```

n = state(init)
frontier.append(n)
while (frontier not empty)
    n = pull state from frontier
    append n to visited
    if n = goal, return solution
    for all actions in n
        n' = a(n)
        if n' not visited
            append n' to frontier
  
```

frontier    visited



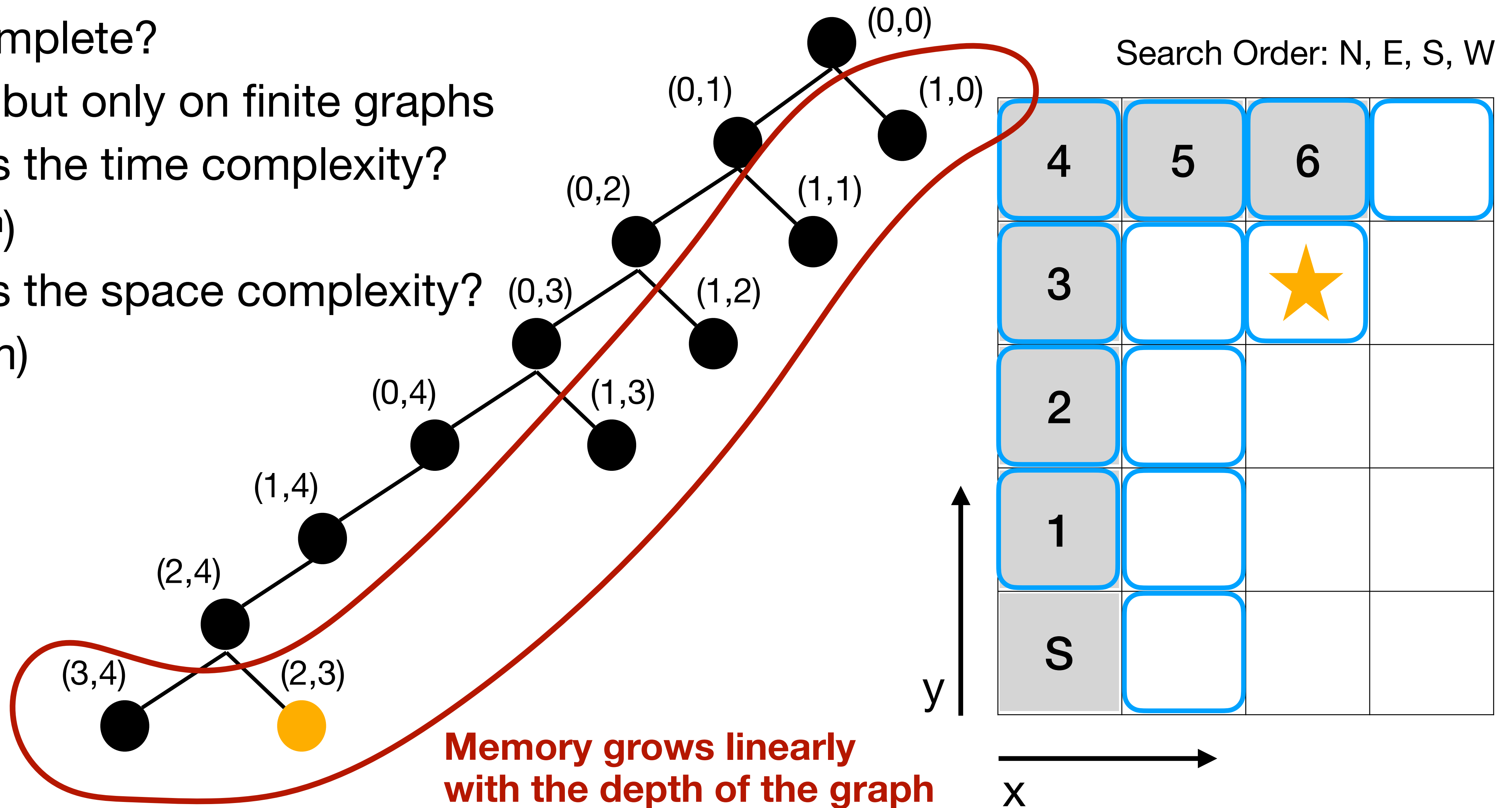
**DFS:** Last-In First-Out (LIFO)

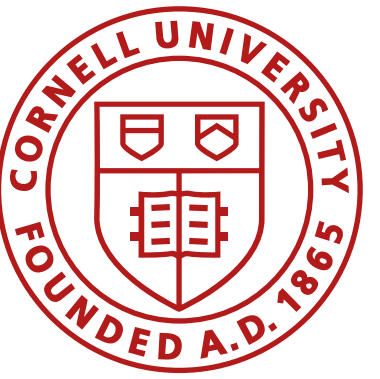
**BFS:** First-In First-Out (FIFO)

**LCFS:** Prioritize cost

# Depth First Search

- Is it complete?
  - Yes, but only on finite graphs
- What is the time complexity?
  - $O(b^m)$
- What is the space complexity?
  - $O(bm)$

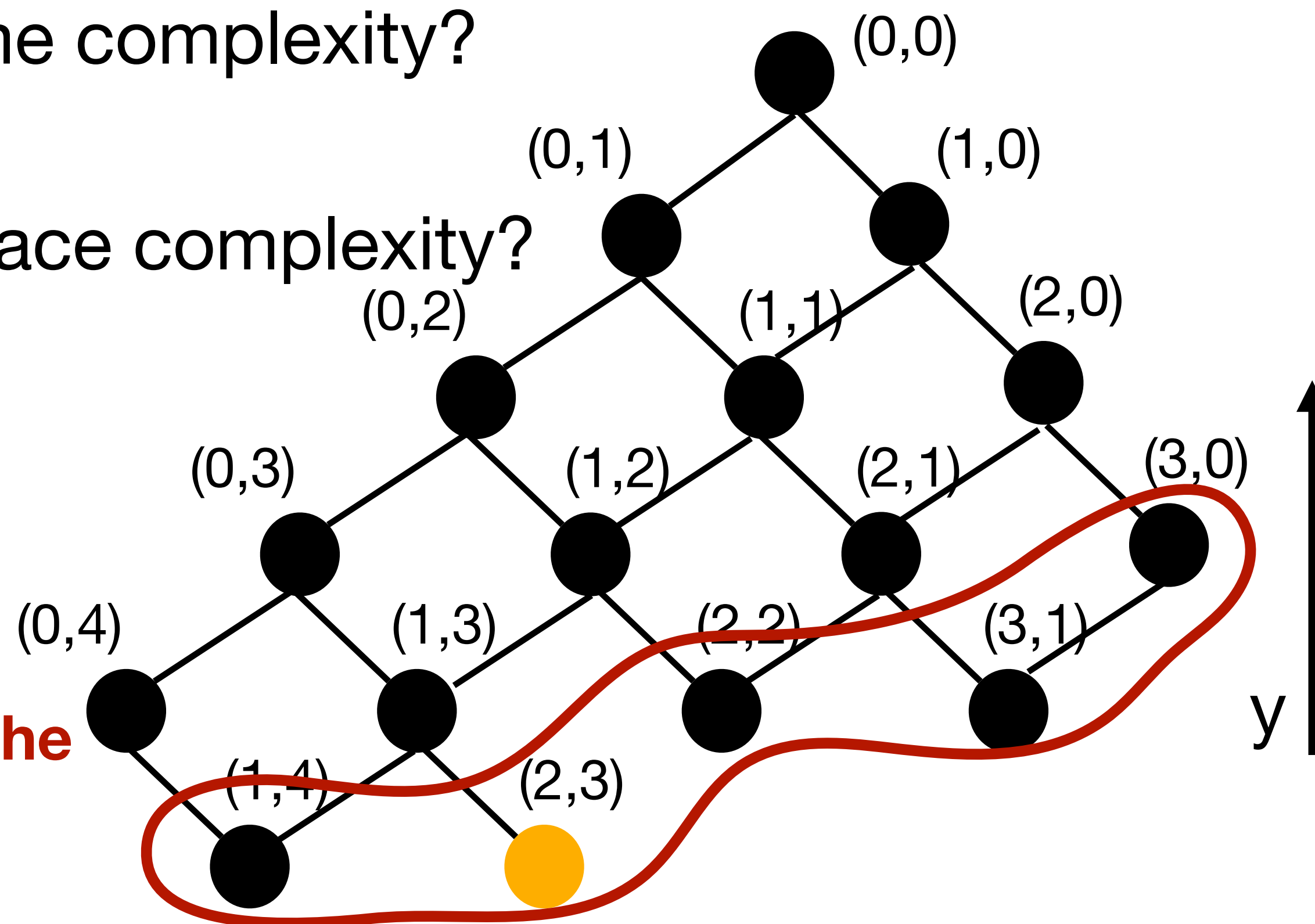




# Breadth First Search

- Is it complete?
  - Yes, as long as  $b$  is finite
- Is it optimal?
  - Yes
- What is the time complexity?
  - $O(b^m)$
- What is the space complexity?
  - $O(b^m)$

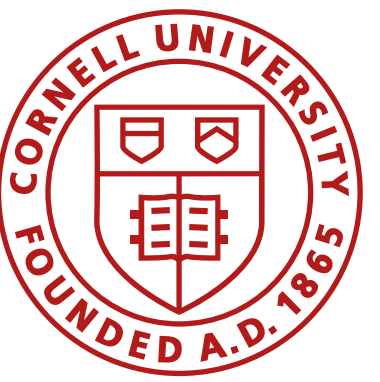
**Memory grows exponentially with the depth of the graph**



Search Order: N, E, S, W



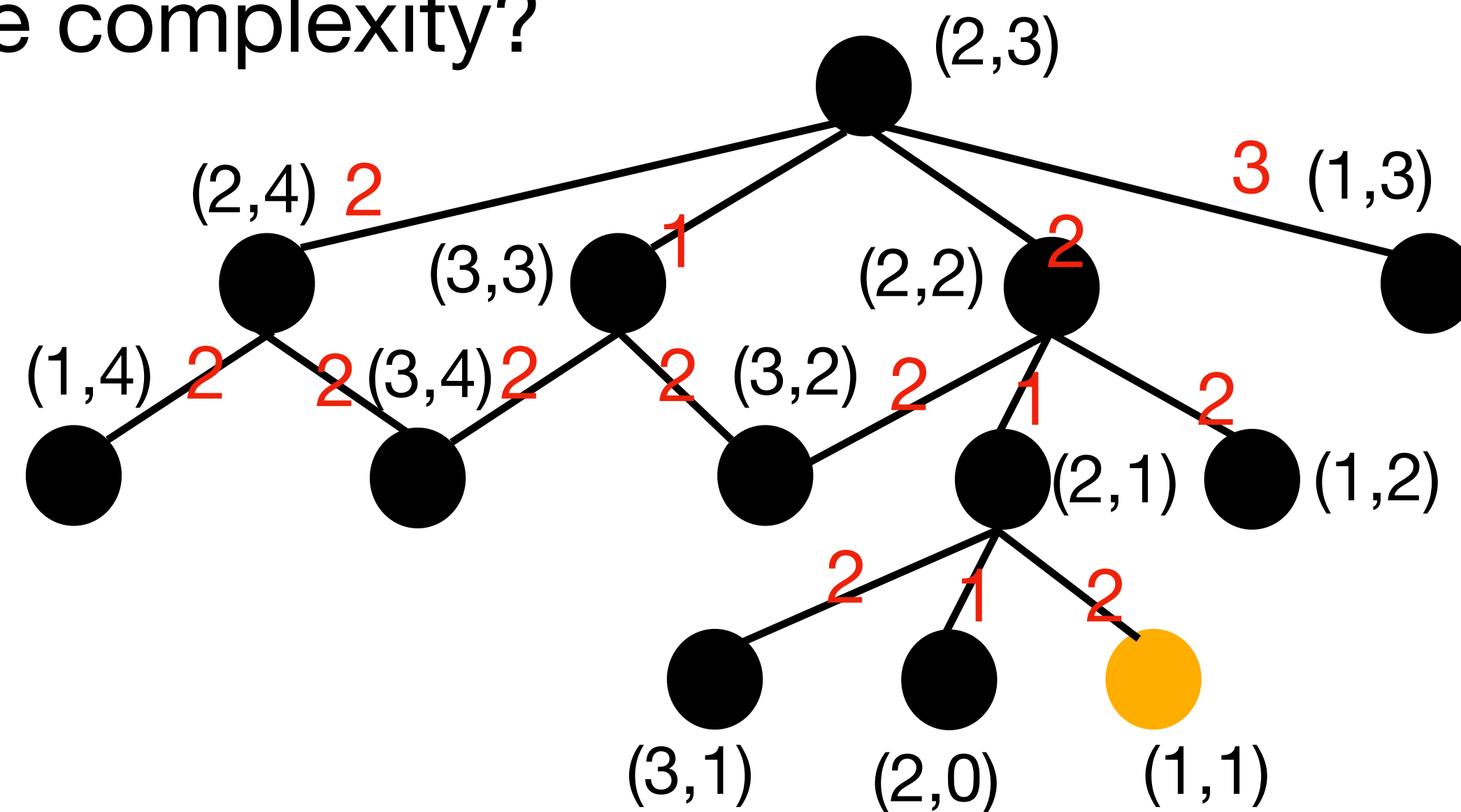
x



# Lowest-Cost First Search (LCFS)

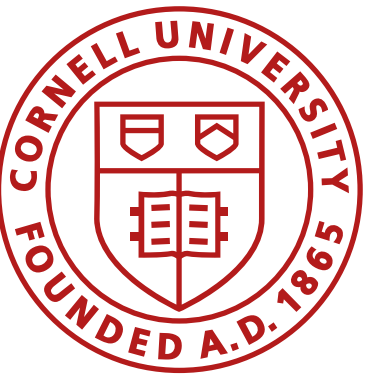
## Consider parent cost!

- Is it complete?
  - Yes, as long as path costs are positive
- What is the time complexity?
  - $O(b^{1+C/c})$
- What is the space complexity?
  - $O(b^{1+C/c})$



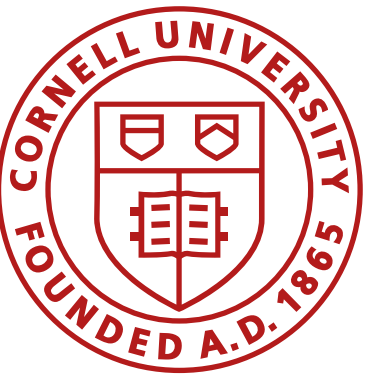
- Go straight, **cost 1**
- Turn one quadrant, **cost 1**

	(1,4)	(2,4)	(3,4)
	(1,3)	<b>R</b> →	(3,3)
	(1,2)	(2,2)	(3,2)
	<b>G</b> ←	(2,1)	(3,1)
		(2,0)	



# Uninformed Search Algorithms

Criterion	BFS	DFS	LCFS
<b>Complete</b>	Yes (finite)	No (finite)	Yes (positive cost)
<b>Time</b>	$O(b^m)$	$O(b^m)$	$O(b^{1+C/c})$
<b>Space</b>	$O(b^m)$	$O(bm)$	$O(b^{1+C/c})$
<b>Optimal</b>	Yes (identical cost)	No	Yes
<b>When to use</b>	<ul style="list-style-type: none"> <li>• Memory is a nonissue</li> <li>• Shallow solutions</li> <li>• Minimal branching factors</li> <li>• Shortest path needed</li> </ul>	<ul style="list-style-type: none"> <li>• Memory is restricted</li> <li>• Deep solutions</li> </ul>	<ul style="list-style-type: none"> <li>• Care about cost over length of path</li> </ul>



# Informed Search

## Greedy Search

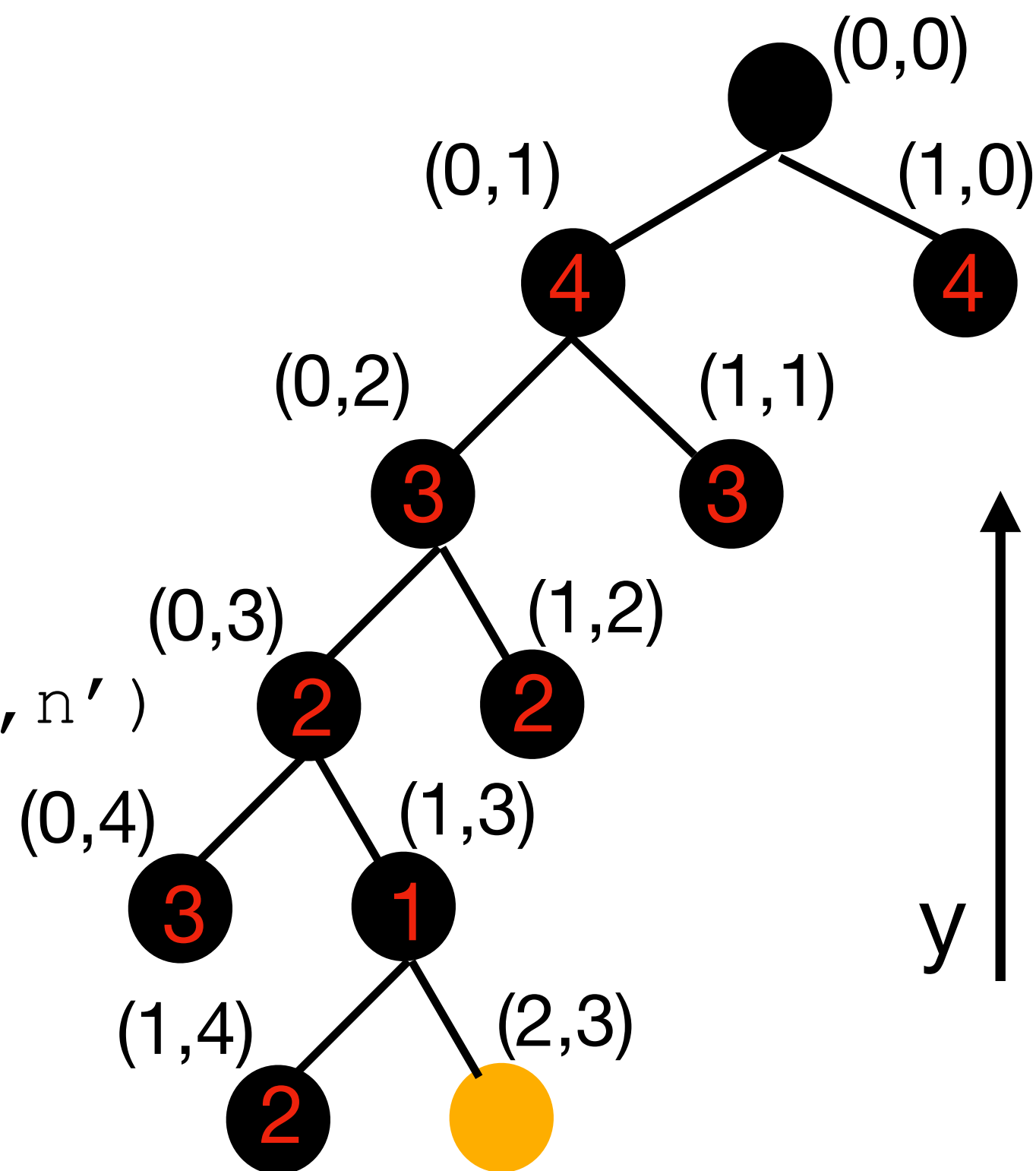
```

n = state(init)
frontier.append(n)
while (frontier not empty)
    n = pull state from frontier
    visited.append(n)
    if n = goal, return solution
    for all actions in n
        n' = a(n)
        if n' not visited
            priority = heuristic(goal, n')
            frontier.append(priority)

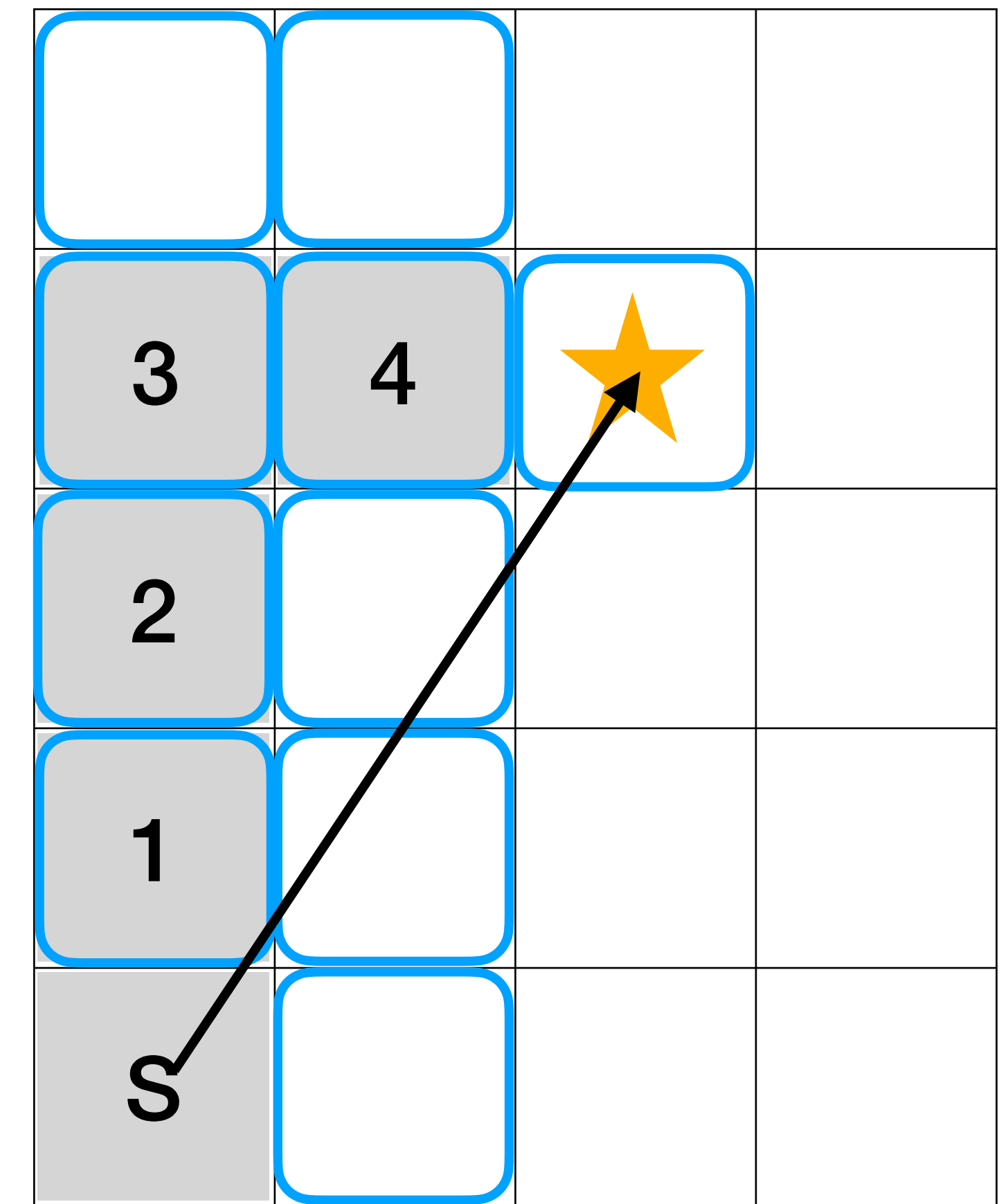
```

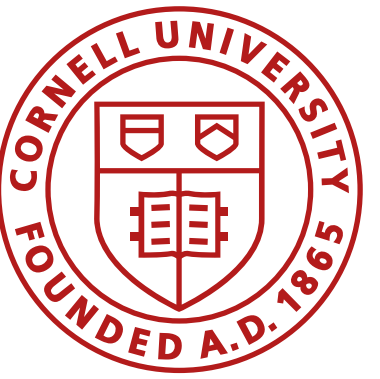
Define a heuristic to target:

- Manhattan Distance
- $\text{abs}(x_s - x_g) + \text{abs}(y_s - y_g)$



Search Order: N, E, S, W



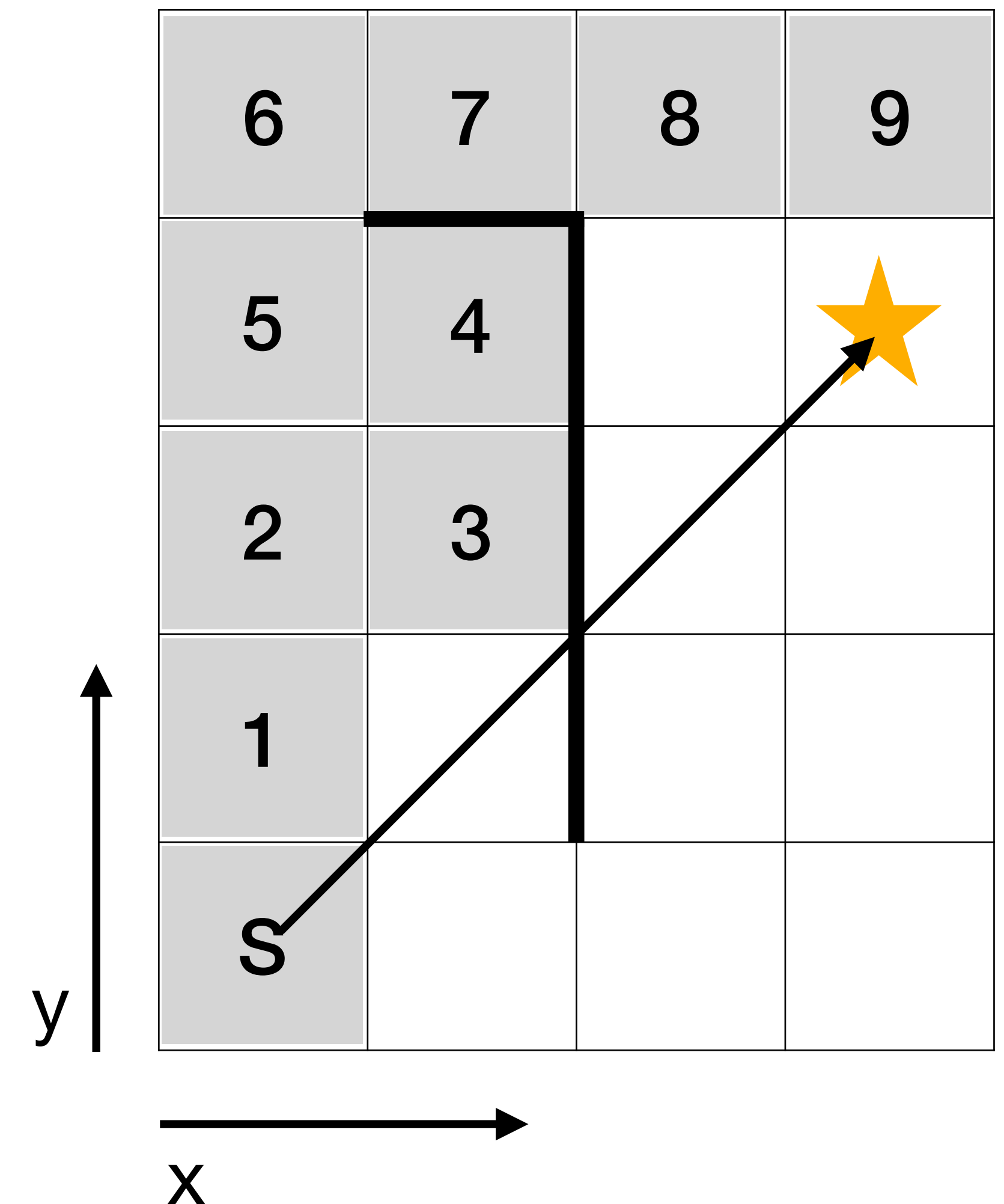


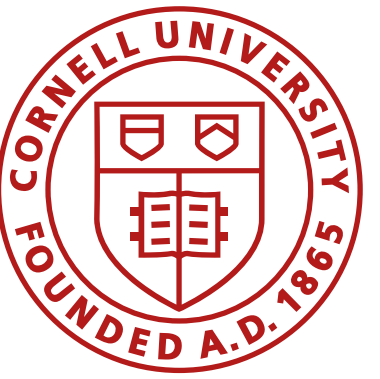
# Informed Search

## Greedy Search

- Is it complete?
  - No
- What is the time complexity?
  - $O(b^m)$
- What is the space complexity?
  - $O(b^m)$
- Optimal?
  - No...

Search Order: N, E, S, W





# Search Algorithms, general

- Breadth First Search
  - Complete and optimal
  - ...but searches everything

**Can we do better? A\***

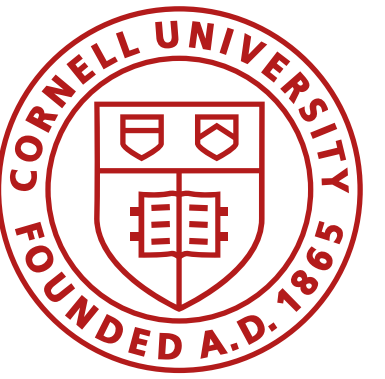
- Lowest-Cost First Algorithm
  - Complete and optimal
  - ... but it wastes time exploring in directions that aren't promising

**Considers parent cost**

- Greedy Search
  - Complete (in most cases)
  - ... only explores promising directions

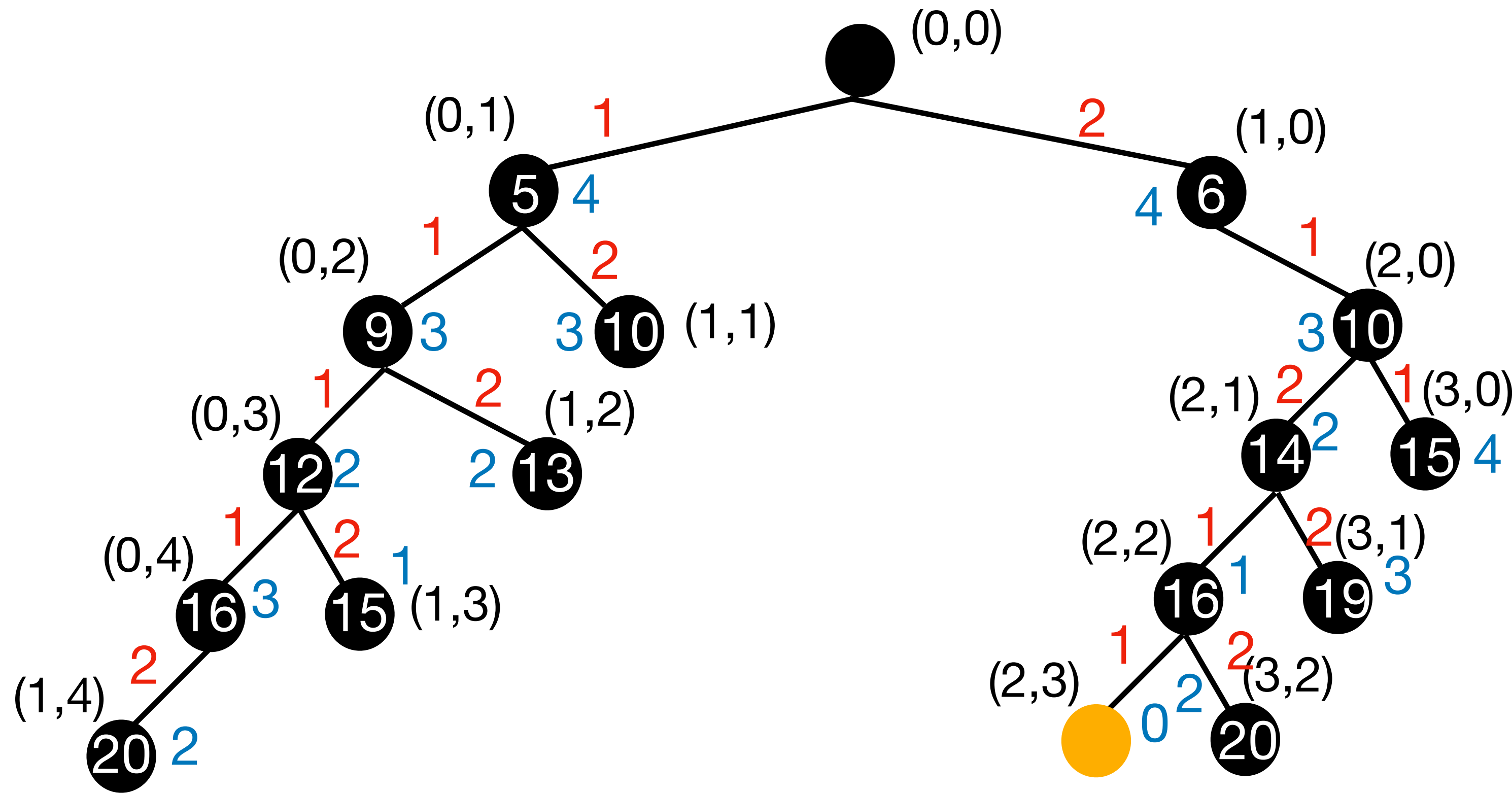
**Considers goal**



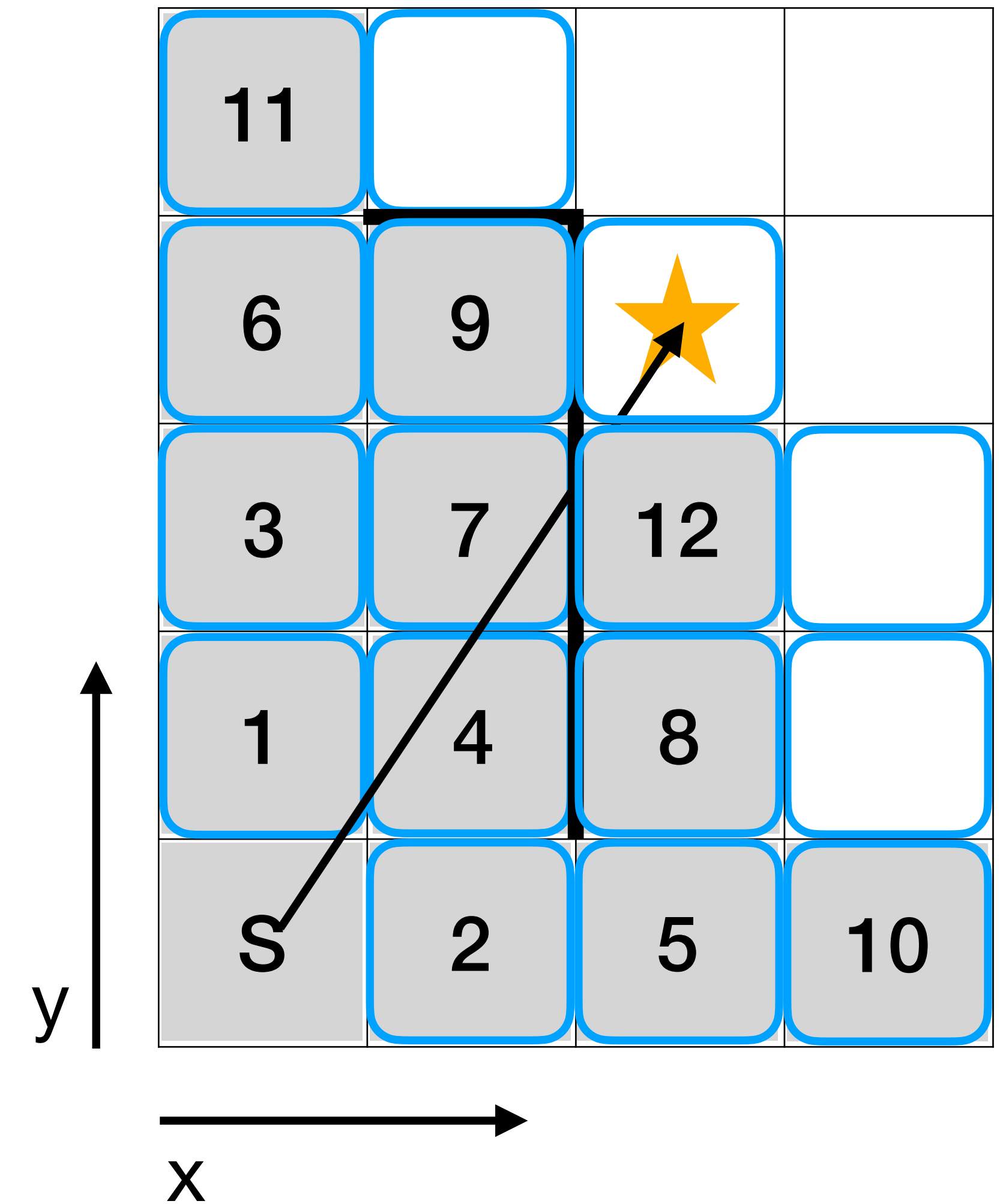


# Informed Search

## A\* (A-star)



Search Order: N, E, S, W

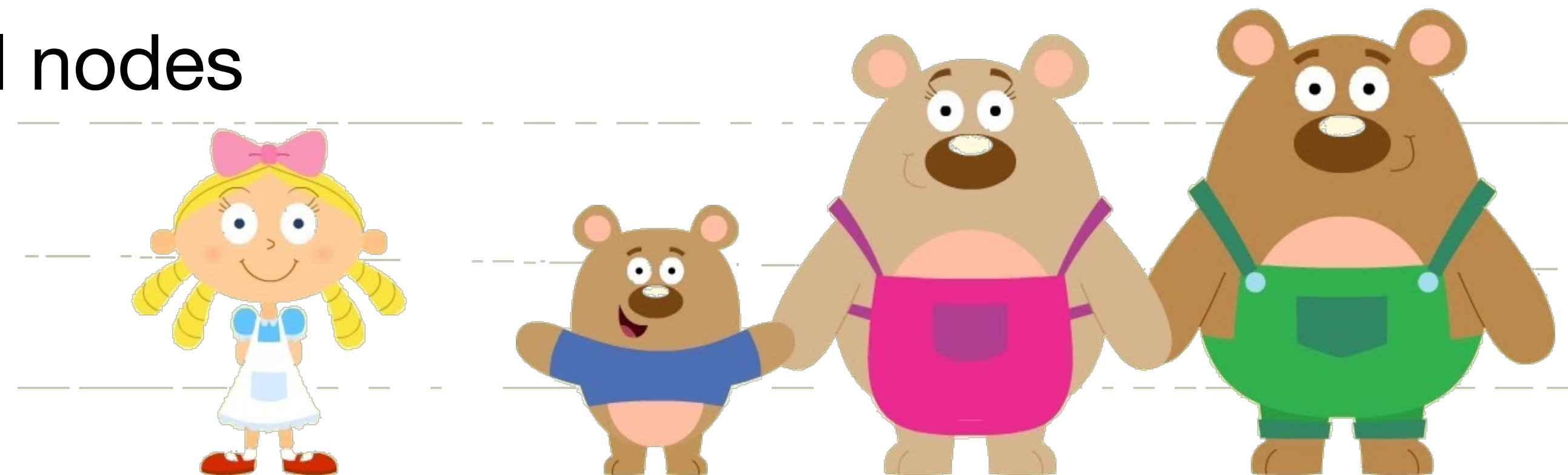


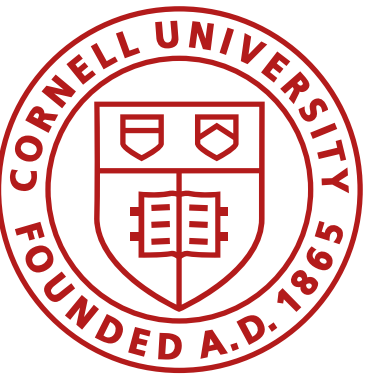
# A\* Search

- What if the heuristic is too optimistic?
  - Estimated cost to goal  $<$  true cost to goal
- What if the heuristic is too pessimistic?
  - Estimated cost to goal  $>$  true cost to goal
  - No longer guaranteed to be optimal
- What if the heuristic is just right?
  - Pre-compute the cost between all nodes
  - Feasible for you?

**Admissible heuristic**

**Inadmissible heuristic**



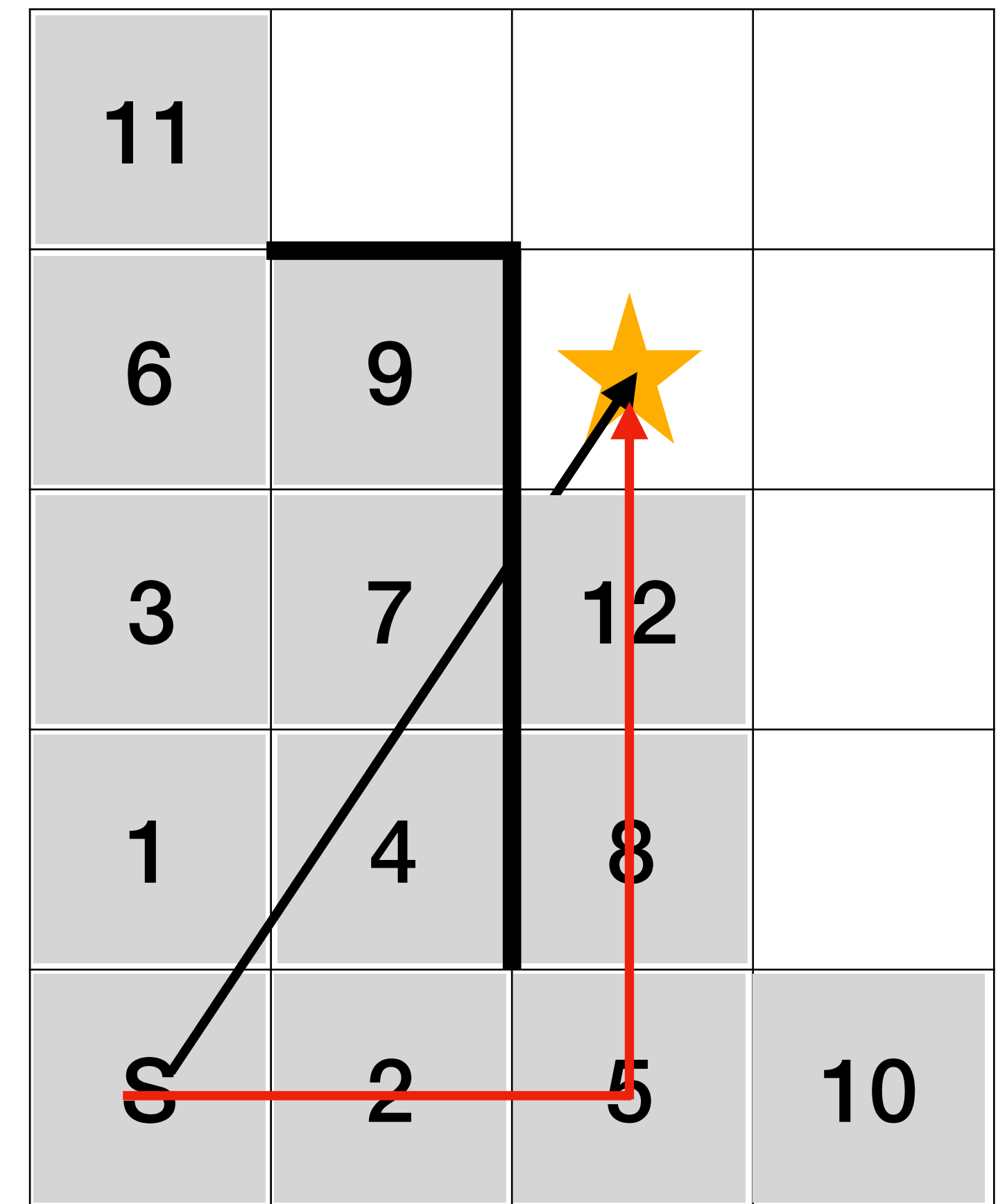


# Informed Search

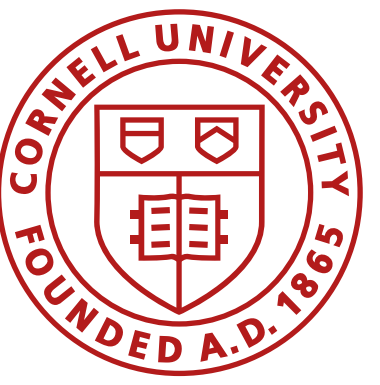
## A\* (A-star)

- Is it complete?
  - Yes!
- What is the time complexity?
  - $O(b^m)$
- What is the space complexity?
  - $O(b^m)$
- Optimal?
  - Yes, if the heuristic is admissible!

A\* minimum path & efficient







# Bayes Filter

**Bayes Theorem**

**+**

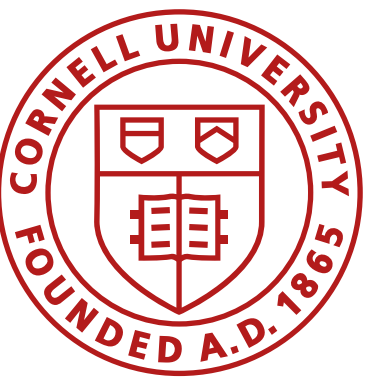
**Robot-Environment Model**

**+**

**Markov Assumption**

**=**

**Bayes Filter**

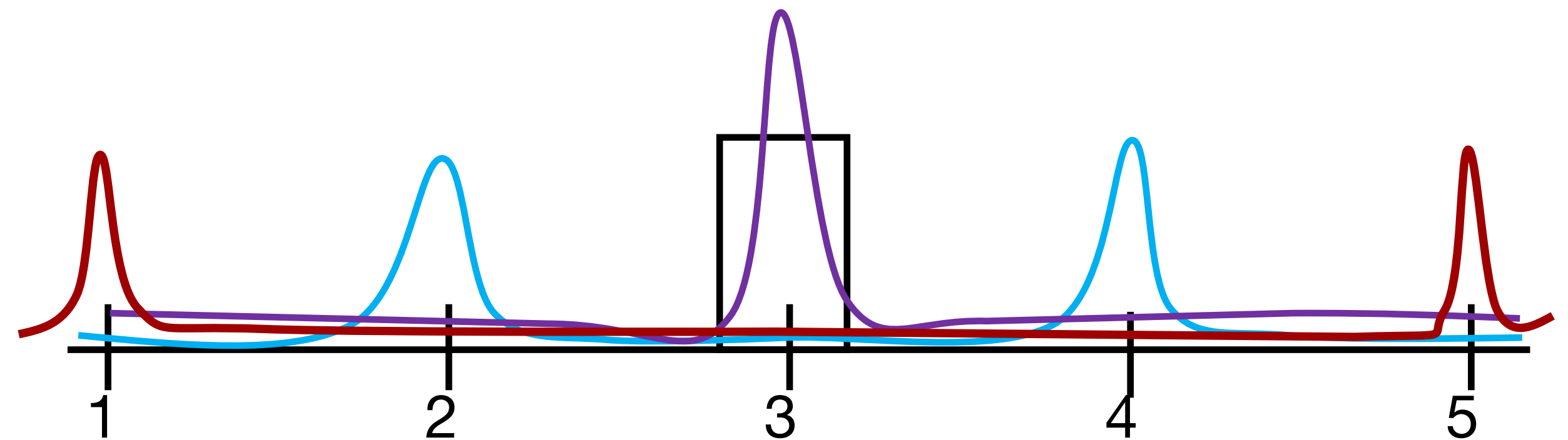


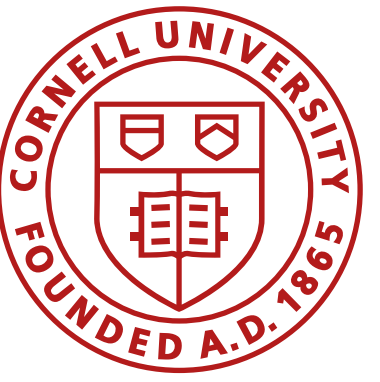
# Bayesian Inference

$$\begin{array}{c} \text{Posterior} \\ \boxed{P(x | z)} \end{array} = \frac{\begin{array}{c} \text{Likelihood} \\ \boxed{P(z | x)} \end{array} \begin{array}{c} \text{Prior} \\ \boxed{P(x)} \end{array}}{\begin{array}{c} \boxed{P(z)} \\ \text{Marginal Likelihood} \\ \text{(constant)} \end{array}}$$

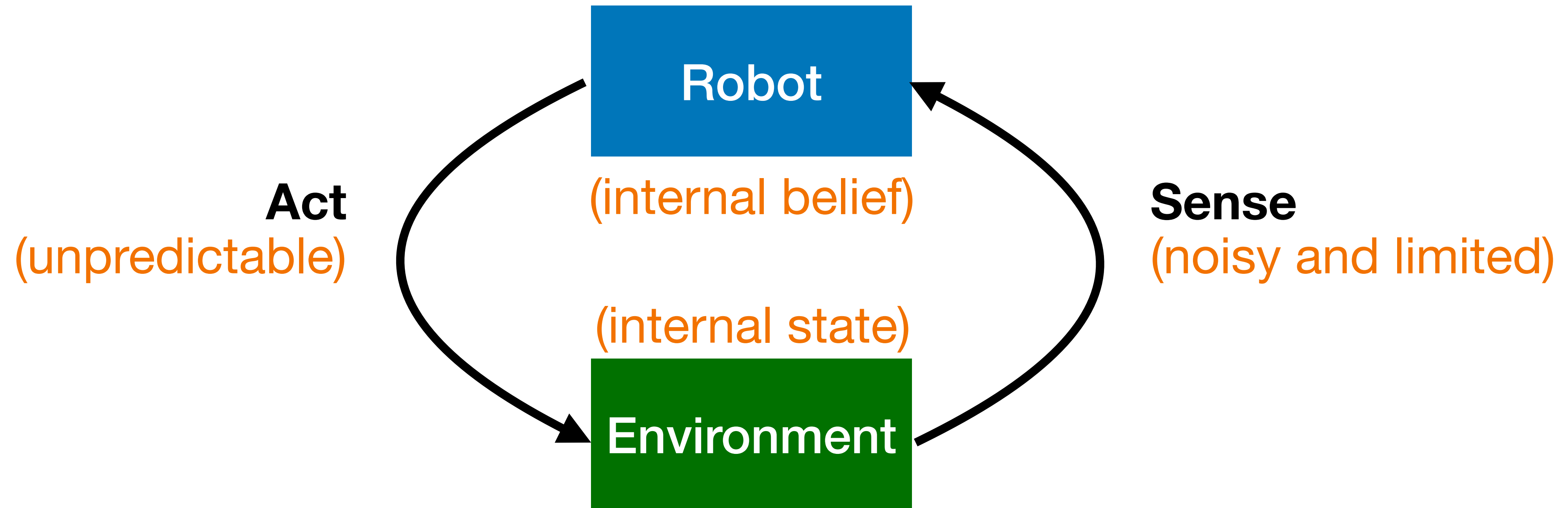
$x$ : robot pose  
 $z$ : sensor data

- Lost robot example
  - Sensor measures distance to the door
  - $p(X_0 = 1 \text{ or } 2 \text{ or } 3 \text{ or } 4 \text{ or } 5) = 1/5$
  - $p(x | z)$  can be hard to compute
  - What is  $p(z | x)$ ?
  - If  $Z = 1$ , where are you most likely to be?
  - If  $Z = 0$ , where are you most likely to be?
  - If  $Z = 2$ , where are you most likely to be?

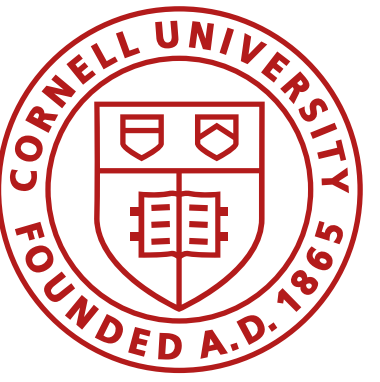




# Robot-Environment Interaction

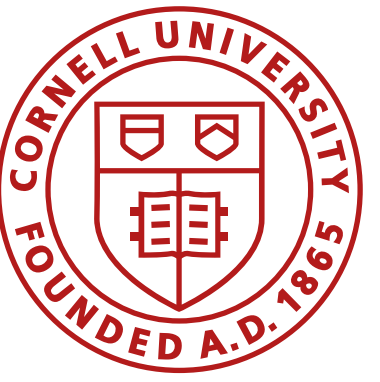


- Two fundamental types of interaction between a robot and its environment:
  - Sensor measurements/ observations
  - Control actions



# Robot-Environment Model

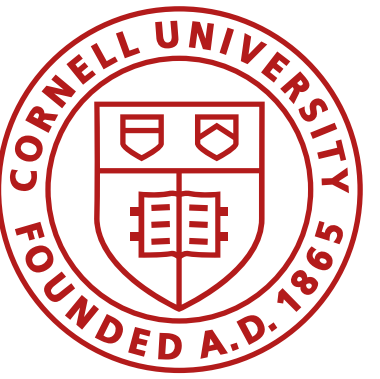
- Helps us express a robot-environment interaction using probability
  - Typically modeled as a discrete time system
    - The **state** at time  $t$  will be denoted as  $x_t$
    - A **sensor measurement** at time  $t$  will be denoted as  $z_t$
    - A **control action** will be denoted as  $u_t$ 
      - Induces a transition from  $x_{t-1}$  to  $x_t$



# Robot-Environment Model

## Assumptions (arbitrary)

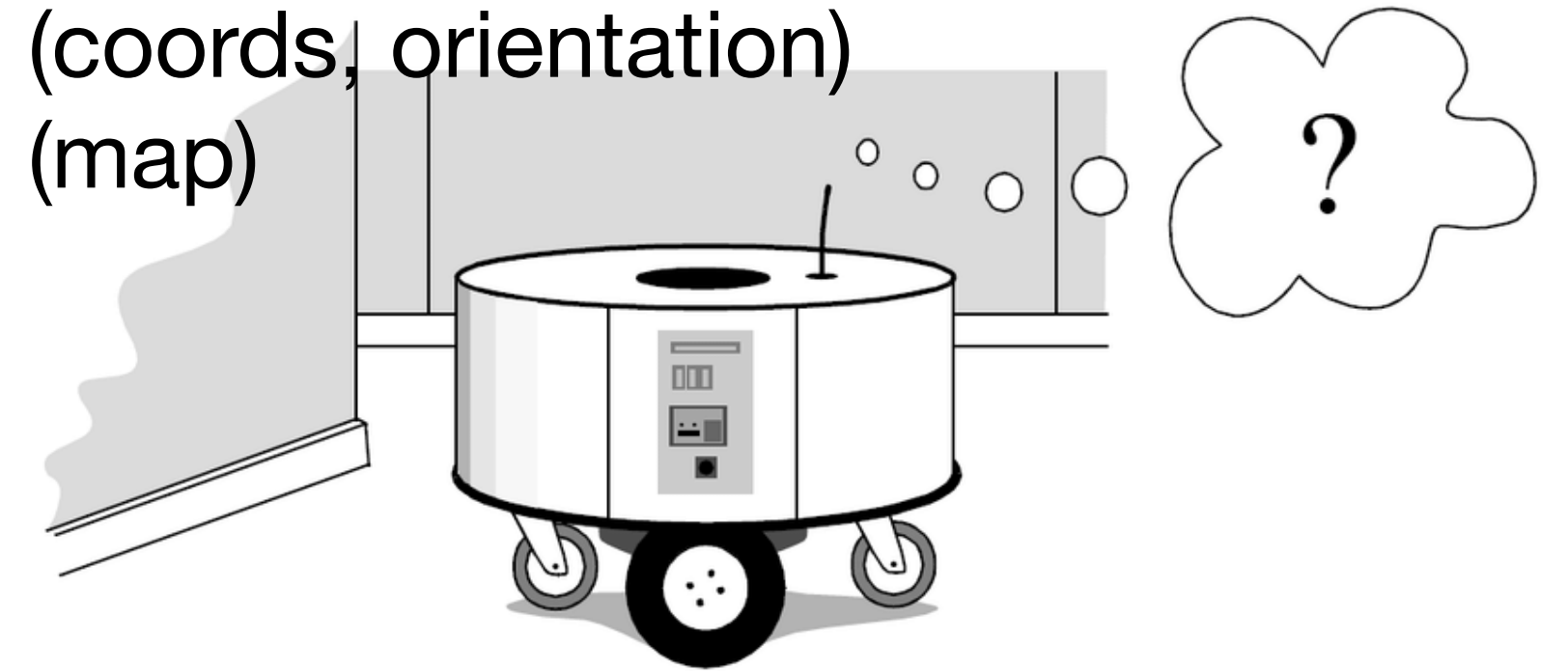
- The robot executes a control action  $u_t$  first and then takes a measurement  $z_t$
- There is one control action  $u$  per time step  $t$ 
  - This includes the legal action “do-nothing”
- There is only one measurement  $z$  per time step  $t$
- Shorthand notation:  $x_{t1:t2} = x_{t1}, x_{t1+1}, x_{t1+2}, \dots, x_{t2}$



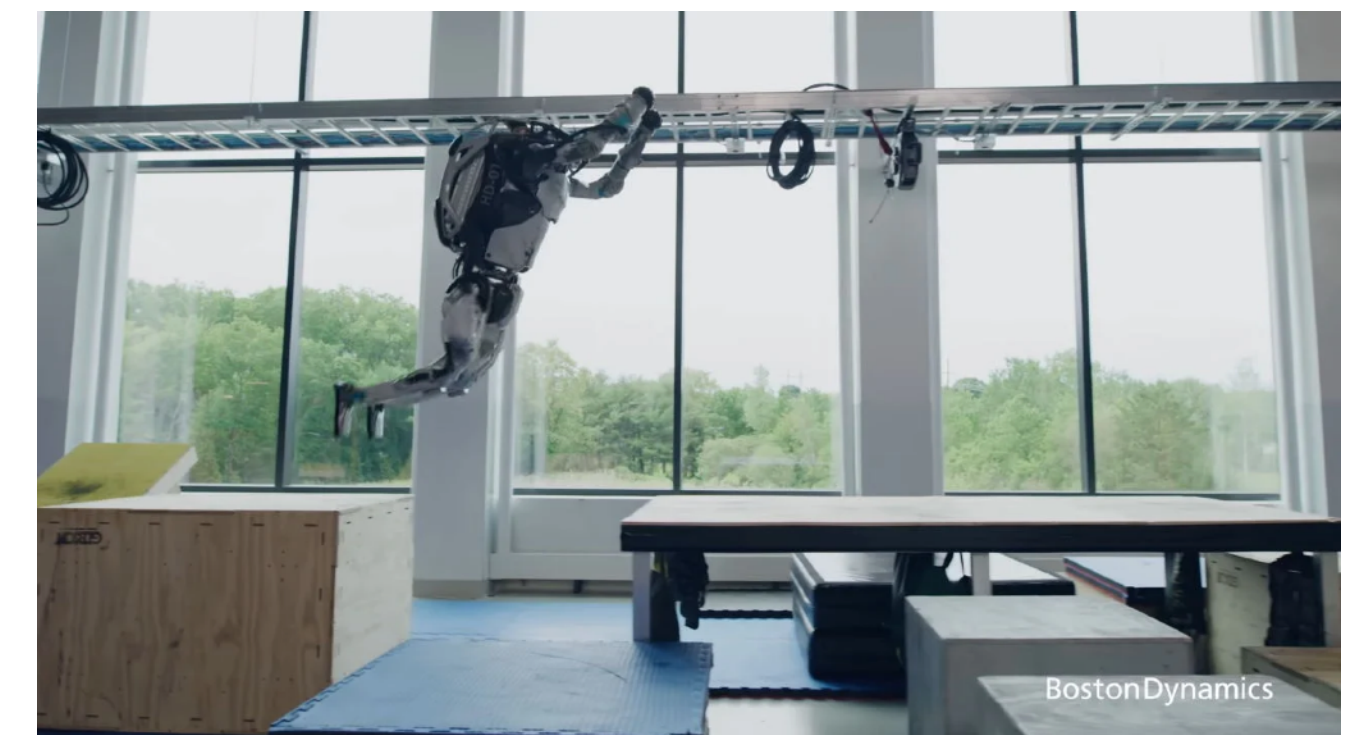
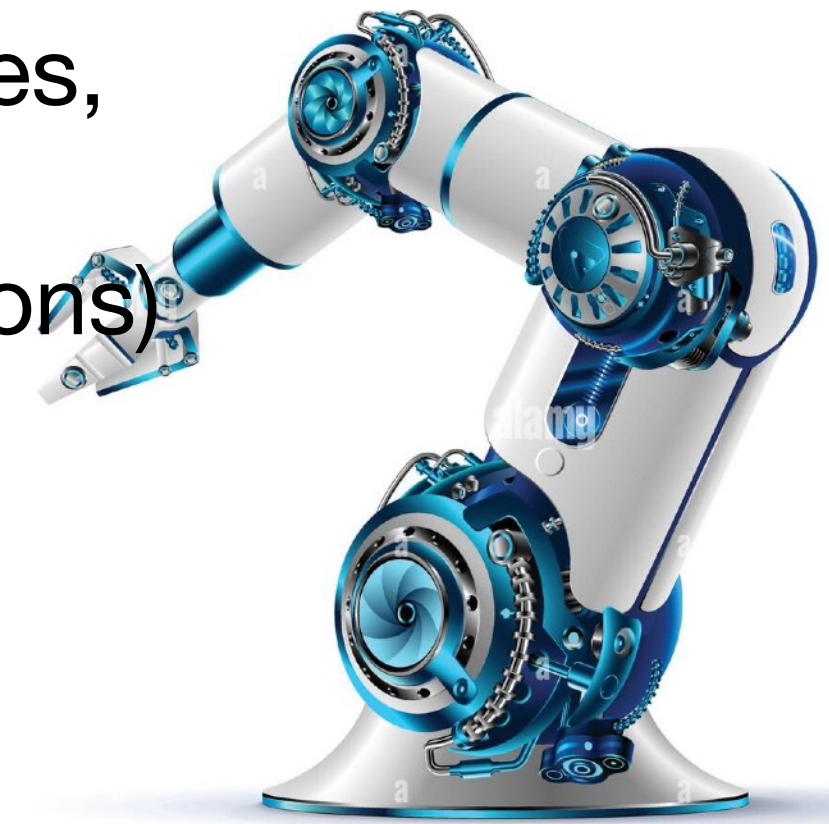
# Robot State

- Robot-specific:
  - Pose, velocity, sensor status, etc.
- Environment-specific:
  - Static variables: locations of walls, etc.
  - Dynamic variables: people, etc.
- ... context specific

(coords, orientation)  
(map)



(joint angles,  
velocities,  
accelerations)  
(objects,  
texture)



# Sensor Measurements/ Observations

- Tend to increase the robot's knowledge

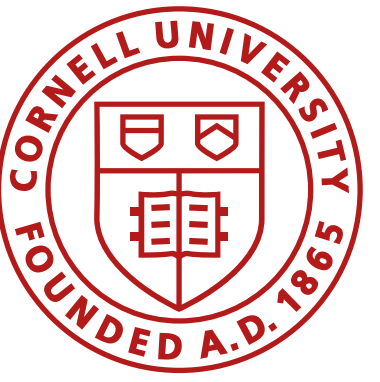
balance



# Control Actions

- ... change the state of the world
- Carry information about the change of robot state in the time interval  $(t - 1 : t]$
- Tends to induce loss of knowledge





# Probabilistic Generative Laws

- The evolution of state and measurements is governed by probabilistic laws
  - State: How is  $x_t$  generated stochastically?
  - Measurements: How is  $z_t$  generated stochastically?
- State generation
  - $x_t$  depends on  $x_{0:t-1}$ ,  $z_{1:t-1}$ , and  $u_{1:t}$
  - $p(x_t | x_{0:t-1}, z_{1:t-1}, u_{1:t})$  **...intractable!**

**Bayes Theorem**

**+**

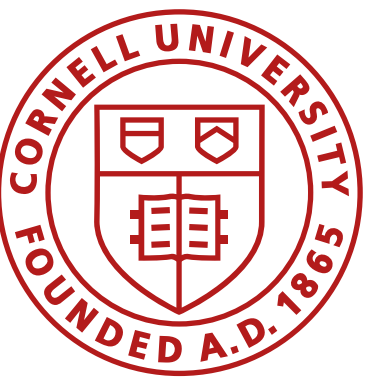
**Robot-Environment Model**

**+**

**Markov Assumption**

**=**

**Bayes Filter**



# Markov Assumption

*The Markov assumption postulates that past and future data are independent if one knows the current state*

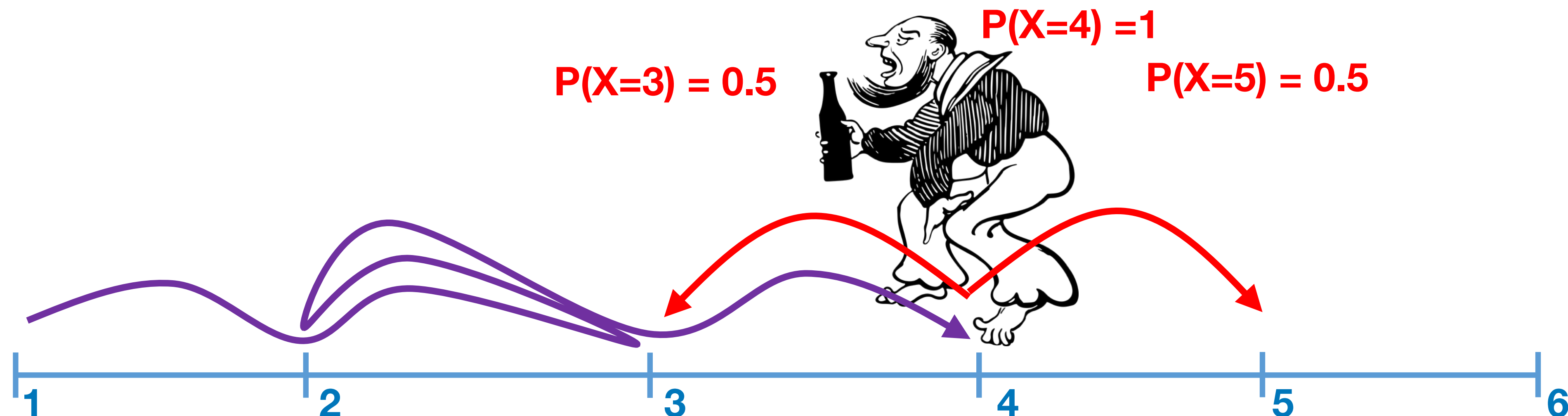
- A stochastic model/ process that obeys the Markov assumption is a Markov model
  - This does not mean that  $x_t$  is deterministic based on  $x_{t-1}$
- **If** we can model our robot as a Markov process...
  - We can recursively estimate  $x_t$  using:
    - $x_{t-1}$ ,  $z_t$ , and  $u_t$
    - **But not**  $x_{0:t-1}$ ,  $z_{1:t-1}$ , and  $u_{1:t}$ !
    - Tractable!

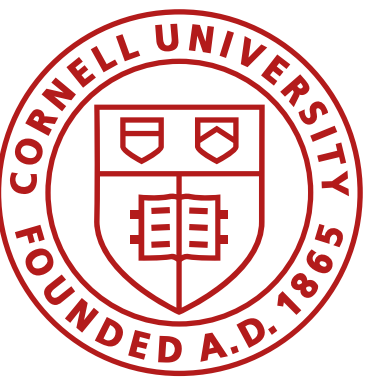


Andrey Markov (1856–1922) was a Russian mathematician best known for his work on stochastic processes

# Drunkards Walk

- Random walk on the number line
  - At each step, the position may change by +1 or -1 with equal probability
- The transition probabilities depend only on the current position, not on the manner in which the position was reached
- This is a Markov process!



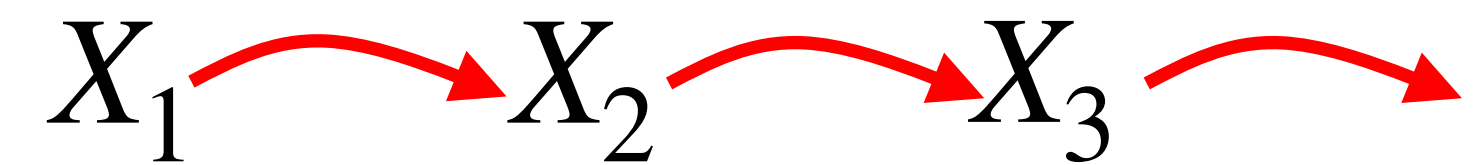


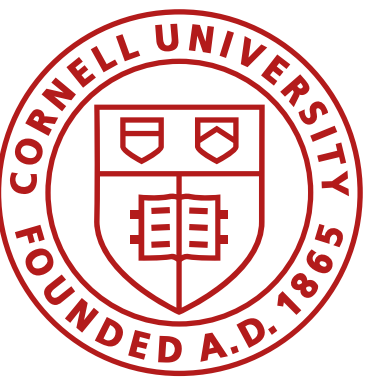
# Coin purse

- Contents
  - 5 quarters (25¢)
  - 5 dimes (10¢)
  - 5 nickels (5¢)
- Draw coins randomly, one at a time and place them on a table
- $X_n$  is the total value of coins on the table after  $n$  draws

- Example:

- First, I draw a nickel, what is  $X_1 =$
- Next I draw a dime, what is  $X_2 =$





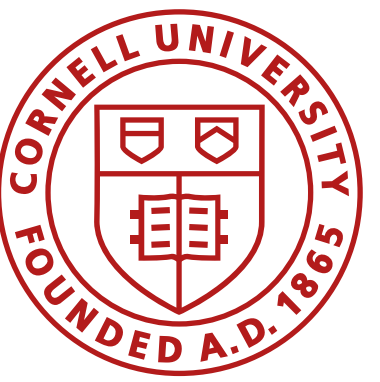
# Coin purse

- Suppose...
  - In the first six draws, you pick all 5 nickels and 1 quarter,  $X_6 = 50\text{¢}$
  - What can we say about  $X_7$ ?

- Contents
  - 5 quarters (25¢)
  - 5 dimes (10¢)
  - 5 nickels (5¢)

- Is this a Markov Model?





# Coin purse

- Markov model
  - $X_n = \{ \text{quarters, dimes, nickels} \}$  remaining
  - $V_n = 25(5 - q) + 10(5 - d) + 5(5 - k)$
  - First, you pick a nickel
    - $X_1 = \{5, 5, 4\}$
    - $X_6 = \{4, 5, 0\}$
  - Now, what can you say about  $V_7$ ?
    - $P(V_7 \geq 60) = 1$
  - State space has  $6^3$  possible states
  - Transition probabilities are independent of the number of draws!

- Contents
  - 5 quarters (25¢)
  - 5 dimes (10¢)
  - 5 nickels (5¢)



**Bayes Theorem**

**+**

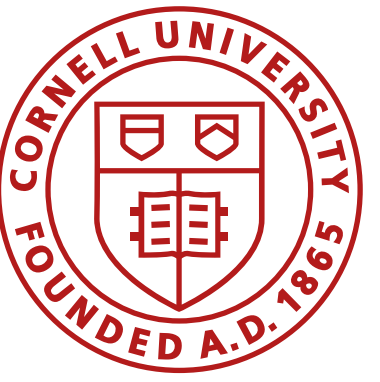
**Robot-Environment Model**

**+**

**Markov Assumption**

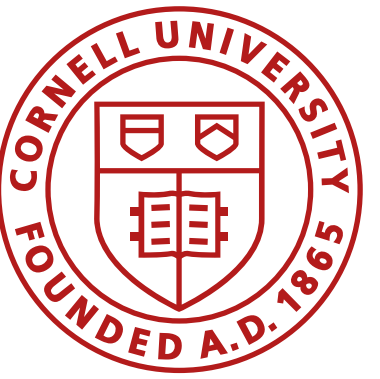
**=**

**Bayes Filter**



# State Generative Model

- $x_t$  is generated stochastically from the state  $x_{t-1}$
- $x_t$  depends on  $x_{0:t-1}$ ,  $z_{1:t-1}$ , and  $u_{1:t}$ , and  $p(x_t | x_{0:t-1}, z_{1:t-1}, u_{1:t}) \dots$  intractable!
- If state  $x_t$  is modeled under the **Markov Assumption**, then
  - $p(x_t | x_{0:t-1}, z_{1:t-1}, u_{1:t}) = p(x_t | x_{t-1}, u_t)$
  - Knowledge of only the previous state  $x_{t-1}$  and control  $u_t$  is sufficient to predict  $x_t$



# Measurement Generative Model

- Similarly, the process by which measurements are generated are of importance
  - $p(z_t | x_{0:t-1}, z_{1:t-1}, u_{1:t})$
- If state  $x_t$  conforms to the **Markov Assumption**, then
  - $p(z_t | x_{0:t-1}, z_{1:t-1}, u_{1:t}) = p(z_t | x_t)$
  - The state  $x_t$  is sufficient to predict the (potentially noisy) measurements
  - Knowledge of any other variable, such as past measurements, controls, or even past states, is irrelevant under the Markov Assumption

**Bayes Theorem**

**+**

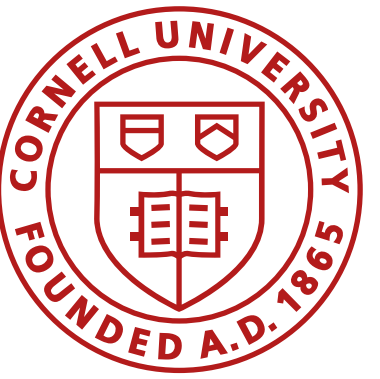
**Robot-Environment Model**

**+**

**Markov Assumption**

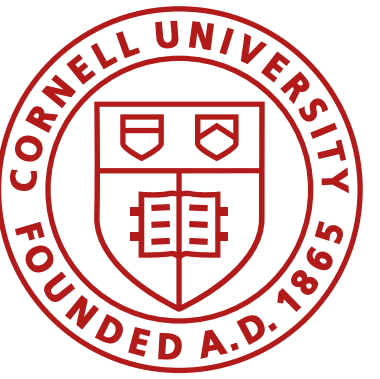
**=**

**Bayes Filter**



# Robot Belief

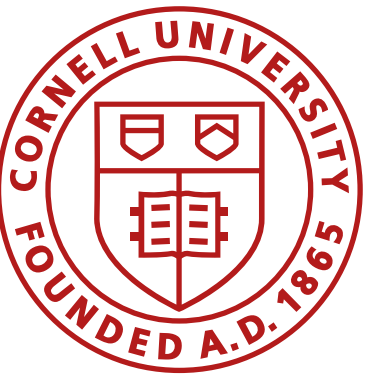
- Probabilistic robotics represents beliefs through *posterior conditional probability distributions*
  - Probability distributions over state variables conditioned on available data
  - The belief of a robot is the posterior distribution over the state of the environment, given all past sensor measurements and all past controls
    - Belief over a state variable  $x_t$  is denoted by  $bel(x_t)$ :
      - $bel(x_t) = p(x_t | z_{1:t}, u_{1:t})$
    - The (prior) belief is the belief before incorporating the latest measurement  $z_t$ :
      - $\overline{bel}(x_t) = p(x_t | z_{1:t-1}, u_{1:t})$



# Bayes Filter

- A recursive algorithm that calculates the belief distribution from measurements and control data

```
1. Algorithm Bayes_Filter ( $bel(x_{t-1}), u_t, z_t$ ) :  
2.   for all  $x_t$  do  
3.      $\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1})$   
4.      $bel(x_t) = \eta p(z_t | x_t) \overline{bel}(x_t)$   
5.   end for  
6. return  $bel(x_t)$ 
```



# Bayes Filter

- A recursive algorithm that calculates the belief distribution from measurements and control data

1. **Algorithm Bayes\_Filter** ( $bel(x_{t-1}), u_t, z_t$ ) :

2. **for all**  $x_t$  **do**

3.  $\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1})$

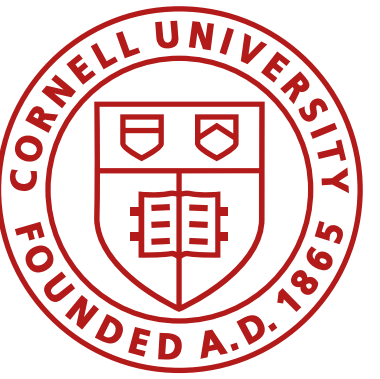
Transition probability/ action model

**(Prediction step)**

4.  $bel(x_t) = \eta p(z_t | x_t) \overline{bel}(x_t)$

5. **end for**

6. **return**  $bel(x_t)$



# Bayes Filter

- A recursive algorithm that calculates the belief distribution from measurements and control data

1. **Algorithm Bayes\_Filter** ( $bel(x_{t-1}), u_t, z_t$ ) :

2. **for all**  $x_t$  **do**

3.  $\bar{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1})$  **(Prediction step)**

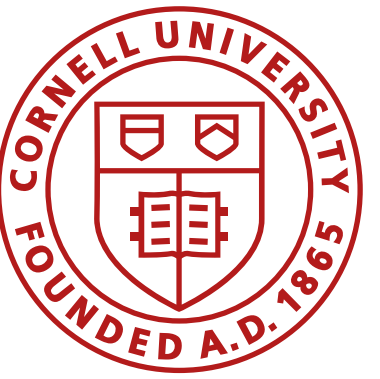
4.  $bel(x_t) = \eta p(z_t | x_t) \bar{bel}(x_t)$  **(Update/measurement step)**

5. **end for**

6. **return**  $bel(x_t)$

Transition probability/ action model

Measurement probability/ sensor model



# Kalman Filter Implementation

Kalman Filter ( $\mu(t-1)$ ,  $\Sigma(t-1)$ ,  $u(t)$ ,  $z(t)$ )

$$1. \mu_p(t) = A\mu(t-1) + Bu(t)$$

**prediction**

$$2. \Sigma_p(t) = A\Sigma(t-1)A^T + \Sigma_u$$

$$3. K_{KF} = \Sigma_p(t)C^T(C\Sigma_p(t)C^T + \Sigma_z)^{-1}$$

$$4. \mu(t) = \mu_p(t) + K_{KF}(z(t) - C\mu_p(t))$$

**update**

$$5. \Sigma(t) = (I - K_{KF}C)\Sigma_p(t)$$

6. Return  $\mu(t)$  and  $\Sigma(t)$

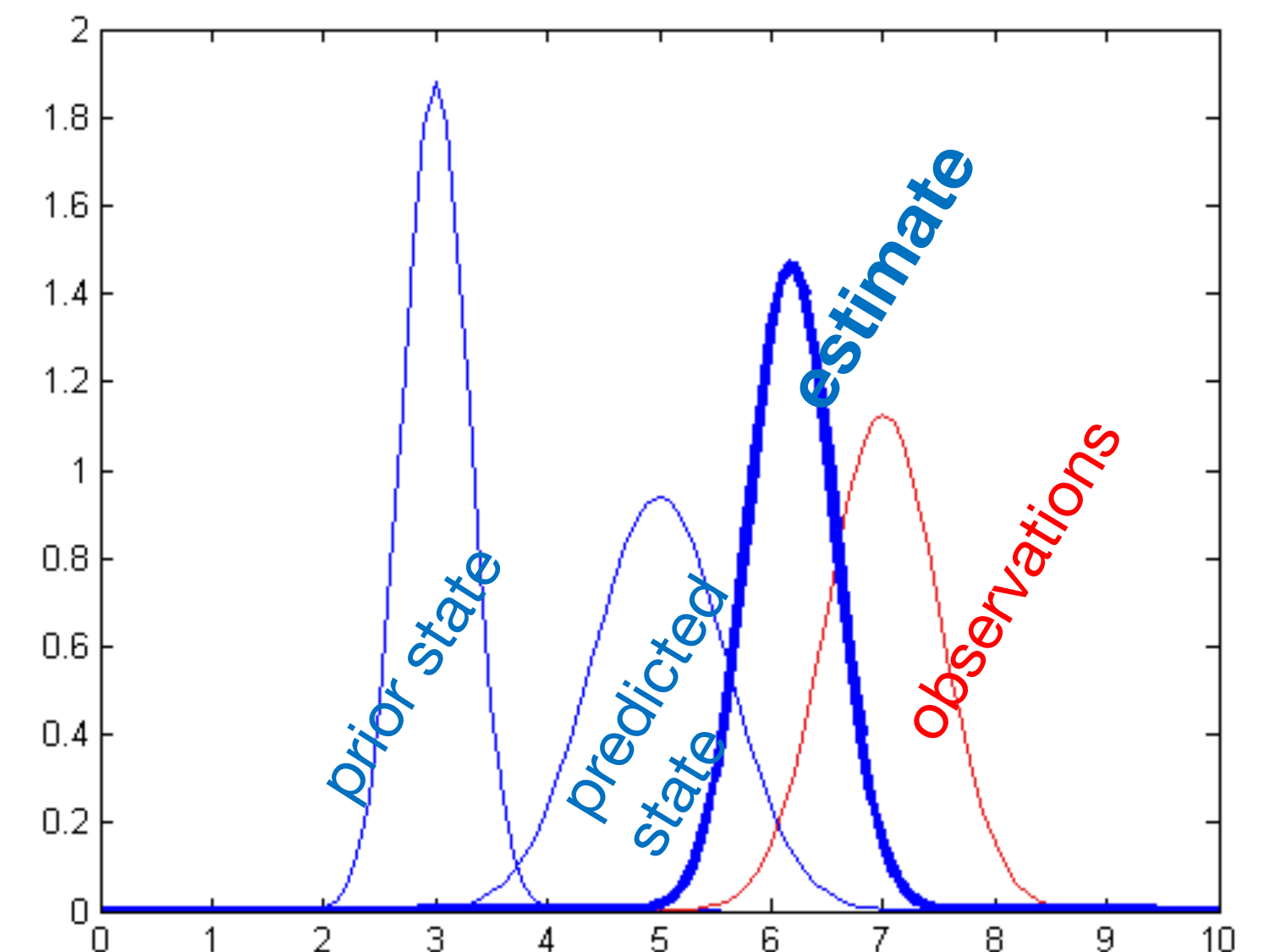
State estimate:  $\mu(t)$

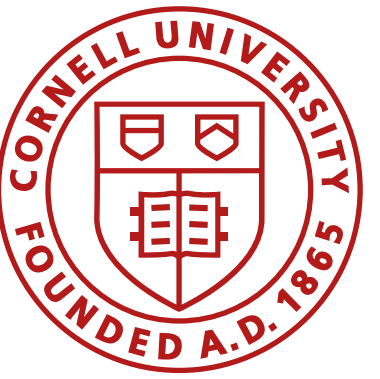
State uncertainty:  $\Sigma(t)$

Process noise:  $\Sigma_u$

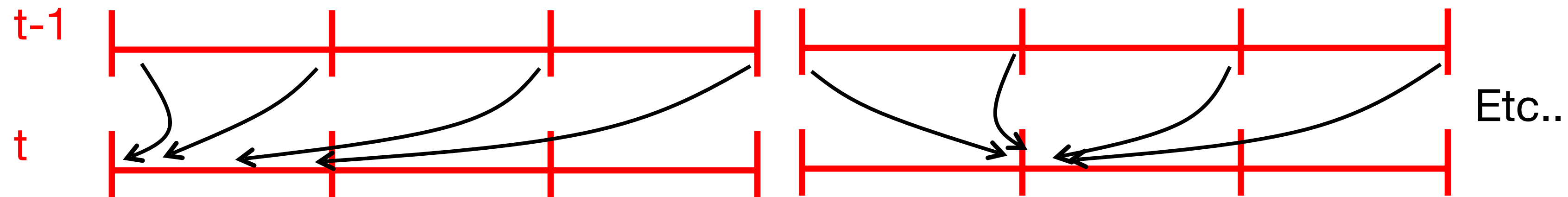
Kalman filter gain:  $K_{KF}$

Measurement noise:  $\Sigma_z$





# Bayes Filter



1. **Algorithm Bayes\_Filter** ( $bel(x_{t-1}), u_t, z_t$ ) :

2. **for** all  $x_t$  **do**

3.  $\bar{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1})$  **(Prediction step)**

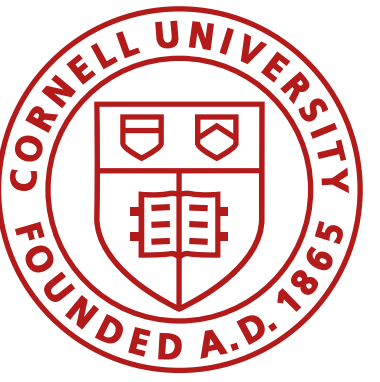
4.  $bel(x_t) = \eta p(z_t | x_t) \bar{bel}(x_t)$  **(Update/measurement step)**

5. **end for**

6. **return**  $bel(x_t)$

Transition probability/ action model

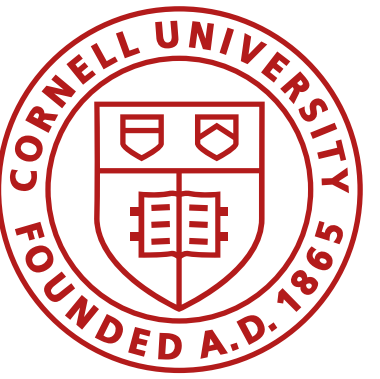
Measurement probability/ sensor model



# Bayes Filter

## Dynamic stochastic model

- $p(x_t | x_{t-1}, u_t)$  is the **state transition probability**
  - How the robot state  $x_t$  evolves over time as a function of the control  $u_t$
- $p(z_t | x_t)$  is the **measurement probability**
  - How measurements are generated from the robot state  $x_t$
  - Informally, you can think of measurements as noisy projections of  $x_t$
- Remember that these prediction are *stochastic and not deterministic*

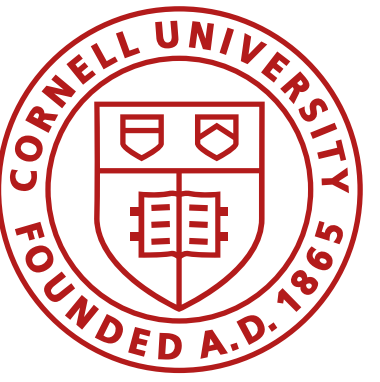


# Bayes Filter

## Initial conditions

- To compute the posterior belief recursively, the algorithm requires an initial belief  $bel(x_0)$  at time  $t = 0$

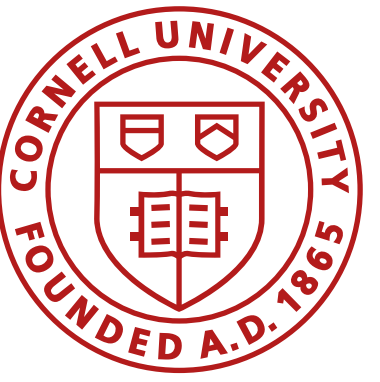
```
1. Algorithm Bayes_Filter ( $bel(x_{t-1}), u_t, z_t$ ) :  
2.   for all  $x_t$  do  
3.      $\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1})$            (Prediction step)  
4.      $bel(x_t) = \eta p(z_t | x_t) \overline{bel}(x_t)$            (Update/measurement step)  
5.   end for  
6. return  $bel(x_t)$ 
```



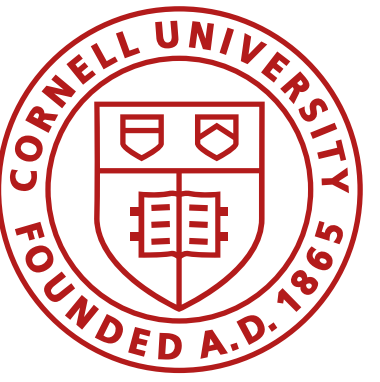
# Bayes Filter

## Initial conditions

- To compute the posterior belief recursively, the algorithm requires an initial belief  $bel(x_0)$  at time  $t = 0$
- If we know the initial state with absolute certainty, we can initialize a point mass distribution that centers all probability mass on the correct value of  $x_0$  and assign zero everywhere else
- If we are entirely ignorant of the initial state, we can initialize it with a uniform probability distribution over all the possible states



**See you guys on Thursday!**



# References

1. Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.
2. <http://gki.informatik.uni-freiburg.de/teaching/ws0607/advanced/recordings/BayesFiltering.pdf>
3. [http://home.deib.polimi.it/restelli/MyWebSite/pdf/E05\\_h.pdf](http://home.deib.polimi.it/restelli/MyWebSite/pdf/E05_h.pdf)
4. [https://en.wikipedia.org/wiki/Markov\\_chain](https://en.wikipedia.org/wiki/Markov_chain)
5. [https://en.wikipedia.org/wiki/Markov\\_model](https://en.wikipedia.org/wiki/Markov_model)
6. Image Sources:
  - a. <https://blog.robotiq.com/a-brief-history-of-robots-in-manufacturing>
  - b. [https://northwesturbanist.files.wordpress.com/2015/01/20151104\\_082523.jpg](https://northwesturbanist.files.wordpress.com/2015/01/20151104_082523.jpg)  
<https://expresswriters.com/is-the-dress-blue-black-white-or-gold-how-it-went-viral/>  
[https://en.wikipedia.org/wiki/Andrey\\_Markov](https://en.wikipedia.org/wiki/Andrey_Markov)