
CS 232: Artificial Intelligence

Fall 2026

Prof. Carolyn Anderson
Wellesley College

Reminders

- ◆ Next week's reading is pages 28-52 of *AI Needs You*
- ◆ I have help hours today (W423) and Monday from 3-4 (W323)
- ◆ Tutor hours:
 - Aeka: Thursdays 7:30-8:30 in H103
 - Amber: Fridays 1:30-2:30 in Whiteboard Alley
 - Coco: Mondays 7-8 in Whiteboard Alley
 - Late Days



Building in the Age of AI: A Founder's Panel

September 24, 2026 · 131 Oliver St 3rd floor



Artem Lukoianov
PhD @ MIT CSAIL;
co-founder of Sundai Club



Rob Blaine
Founder & CEO @ TNT



Brian Devaney
Partner at Underscore VC

Hosted in Partnership with

Underscore VC

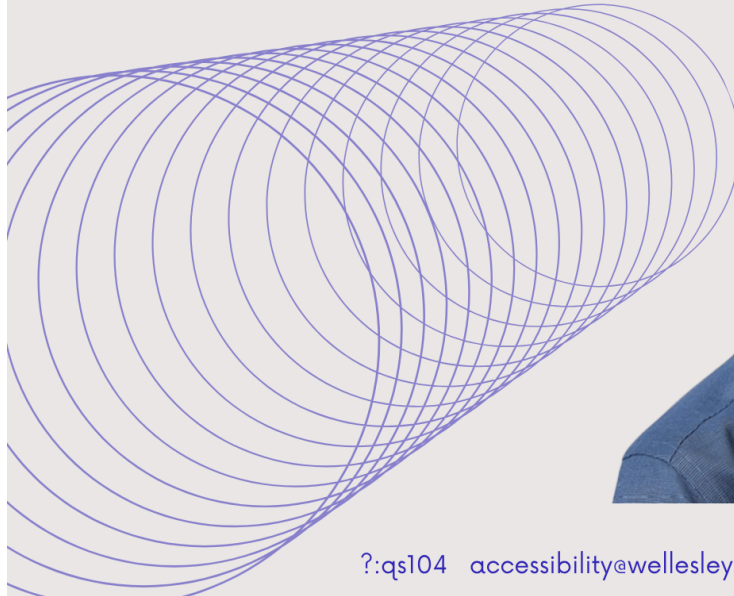
THURSDAY, OCTOBER 8
12:45PM TO 2PM
H105



<https://bit.ly/3V2EJLd>

CS Colloquium:

Tech Ethics and College Chaplaincy: Lessons from MIT Radius



Search

We've seen two kinds of search strategies so far:

- ♦ Uninformed search
 - Breadth-first search
 - Depth-first search
- ♦ Informed search
 - Uniform cost search
 - Greedy best first search
 - A* search

Uniform-Cost Search

- ✦ Like breadth-first search, but with a twist:
Expand the node with the **lowest path cost**
- ✦ Implementation:
Frontier = priority queue ordered by $g(n)$
- ✦ Key difference: when a new path to a node is discovered, if the cost is lower, we need to update the frontier!
- ✦ Also important: apply goal test only when a node is **visited**

A Better Idea

- ◆ Node expansion based on an **estimate** which includes distance to the goal.
- ◆ General approach of informed search:
 - ◆ Best-first search: node selected for expansion based on an **evaluation function $f(n)$**
 - ◆ $f(n)$ includes an **estimate** of the distance to the goal

A* Search

A* Search

$g(n)$: the actual cost so far to go
from start to current node n .

$h(n)$: estimated cost to get from node n
to the goal.

$f(n)$: estimated total cost of the
path start to node n to goal.

Key Idea: Rank nodes by $f(n) = g(n) + h(n)$

Implementation: priority queue ranked by $f(n)$ for frontier.

Admissible Heuristics

A^* is **guaranteed** to find the shortest path if the heuristic **never over-estimates** the cost to the goal.



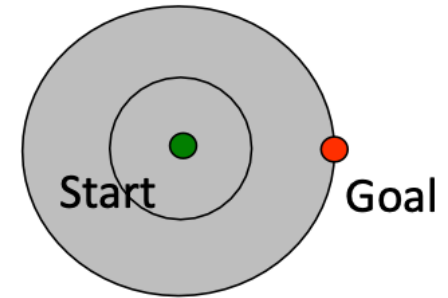
Inadmissible (pessimistic) heuristics break optimality by pushing good plans too far back on the frontier, where they may never be expanded.



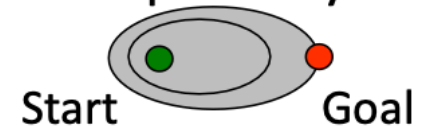
Admissible (optimistic) heuristics may slow down bad plans but never outweigh true costs. That means that the true best plan will always be expanded.

UCS vs A* Contours

Uniform-cost expands equally in all “directions”



A* expands mainly toward the goal, but does hedge its bets to ensure optimality



What is A* Used For?

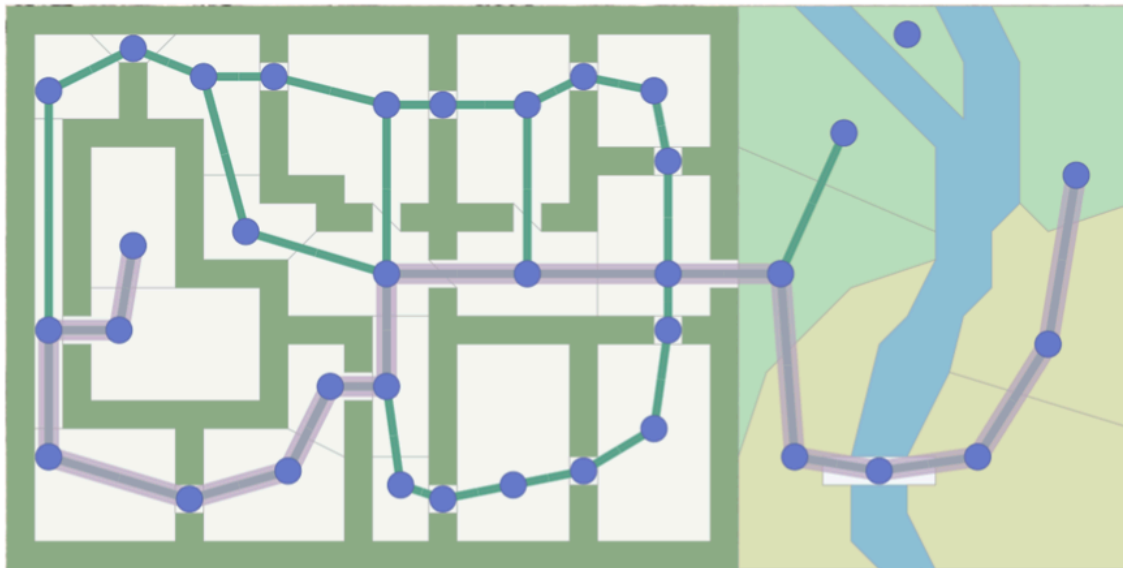
Pathing / routing problems (A* is in your GPS!)

Video games

Robot motion planning

Resource planning problems

...



Heuristics

Admissible Heuristics

A^* is **guaranteed** to find the shortest path if the heuristic **never over-estimates** the cost to the goal.



Inadmissible (pessimistic) heuristics break optimality by pushing good plans too far back on the frontier, where they may never be expanded.



Admissible (optimistic) heuristics may slow down bad plans but never outweigh true costs. That means that the true best plan will always be expanded.

Key Concept: Admissible Heuristics

A heuristic $h(n)$ is admissible only if it never overestimates the cost to reach the goal:

Formally, $\forall n$, n is a node:

$h(n) \leq h^*(n)$ where $h^*(n)$ is the true cost of reaching the goal from start passing through n .

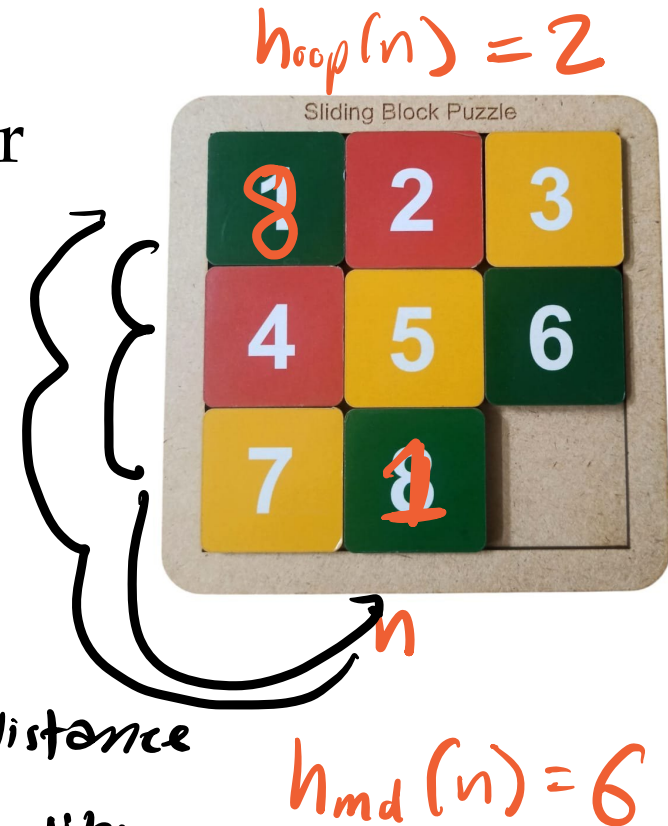
$$h(n) \geq 0$$

Heuristic Functions

What is an admissible heuristic for the 8 puzzle game?

✓ $h_{oop}(n) = \# \text{ of tiles out of place}$

$h_{md}(n) = \text{total Manhattan distance of out of place tiles}$



(Average solution cost is 22 steps; a good heuristic can greatly reduce this!)

Adversarial Search

Search

So far, we have only considered one-player games.
What happens when we add another player?

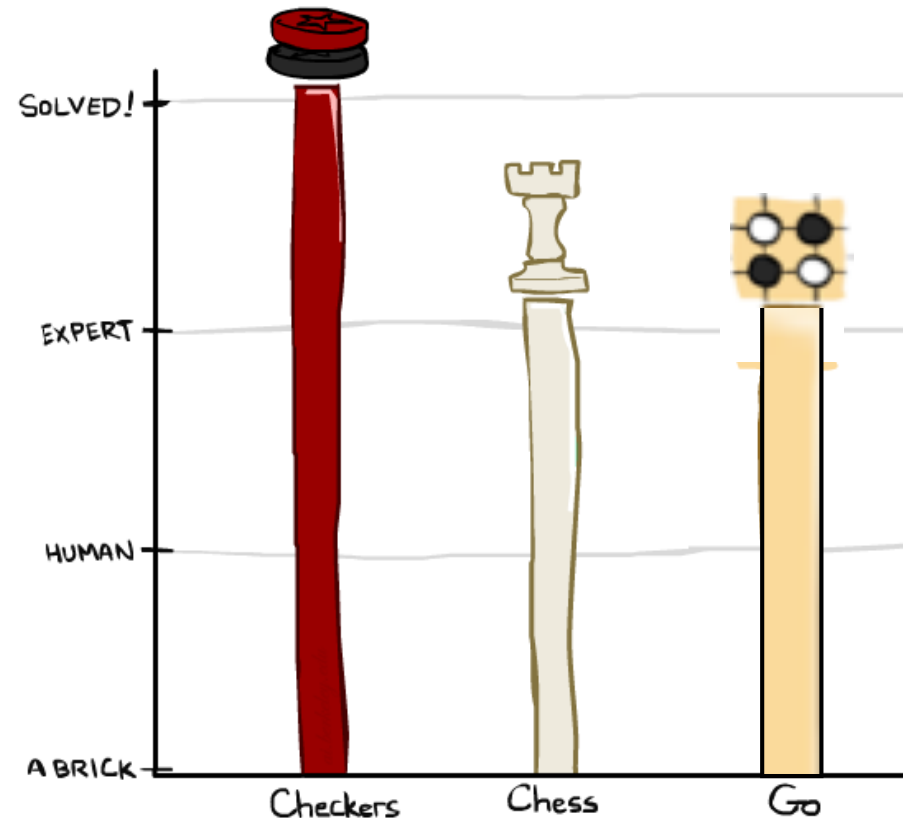
Multiplayer Games

In competitive multiplayer games, we have to consider our opponent's possible actions, as well as our own.

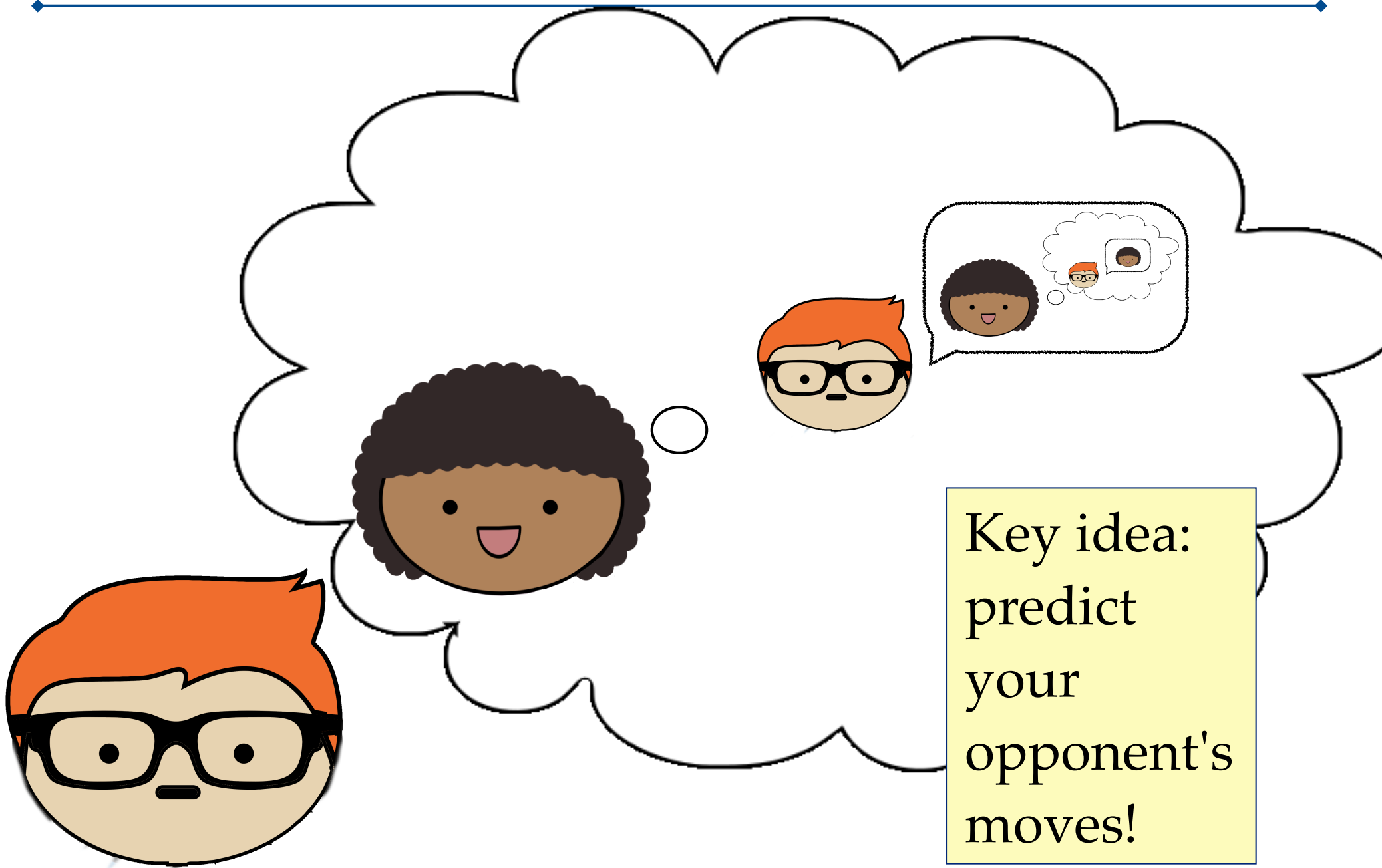
We call this **adversarial search**.

Game Playing State-of-the-Art

- Checkers: 1950: First computer player. 1994: First computer champion: Chinook ended 40-year-reign of human champion Marion Tinsley using complete 8-piece endgame. 2007: Checkers solved!
- Chess: 1997: Deep Blue defeats human champion Gary Kasparov in a six-game match. Deep Blue examined 200M positions per second, used very sophisticated **evaluation** and undisclosed methods for extending some lines of search up to 40 ply. Current programs are even better, if less historic.
- Go: 2016: Alpha GO defeats human champion. Uses Monte Carlo Tree Search + neural network to learn evaluation function.
- Go + Chess + Shogi: 2017: Alpha Zero learns all 3 games using **reinforcement learning** to play against itself.



Deterministic Games



Deterministic Games

States : S

Players : $P = \{1 \dots N\}$

Actions : A

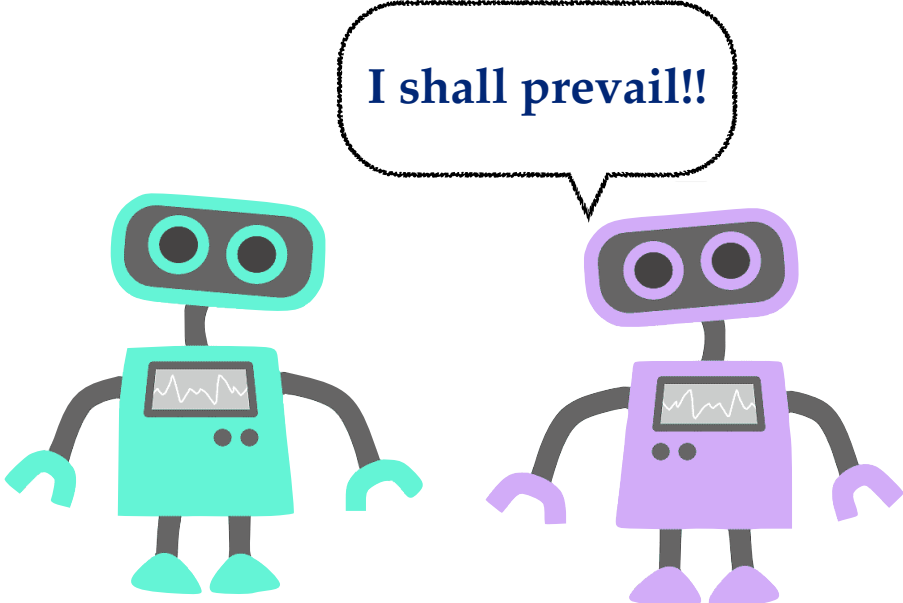
Transition : $T(s, a) \rightarrow s'$

Goal Test / Terminal State : $\text{Terminal}(s) \rightarrow \{\text{True}, \text{False}\}$

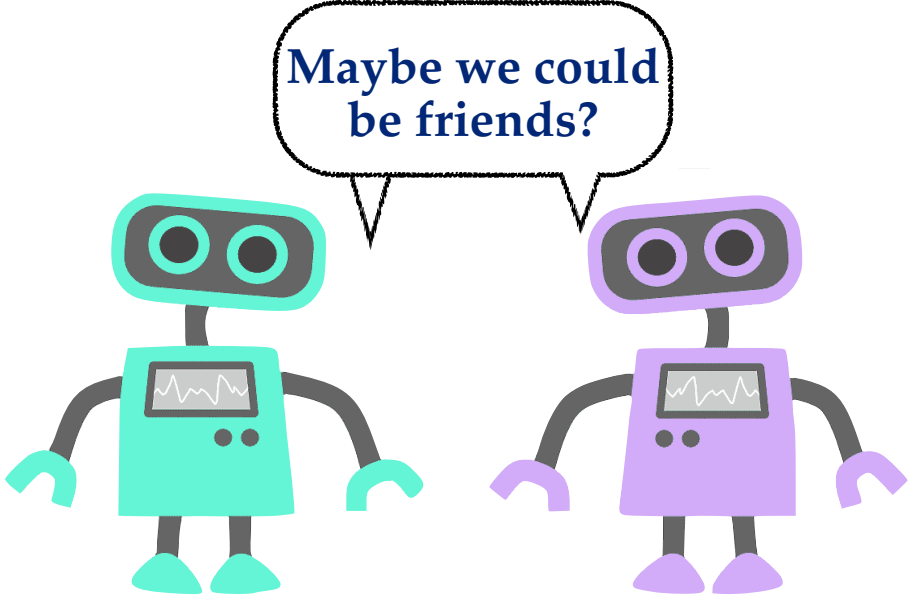
Utility Function : $U(s, p) \rightarrow r$

Solution for a player is a policy :
an action to take in each state
 $\text{Policy}(s) \rightarrow a$

Competing with Adversaries



I shall prevail!!



Maybe we could be friends?

Zero-Sum Games

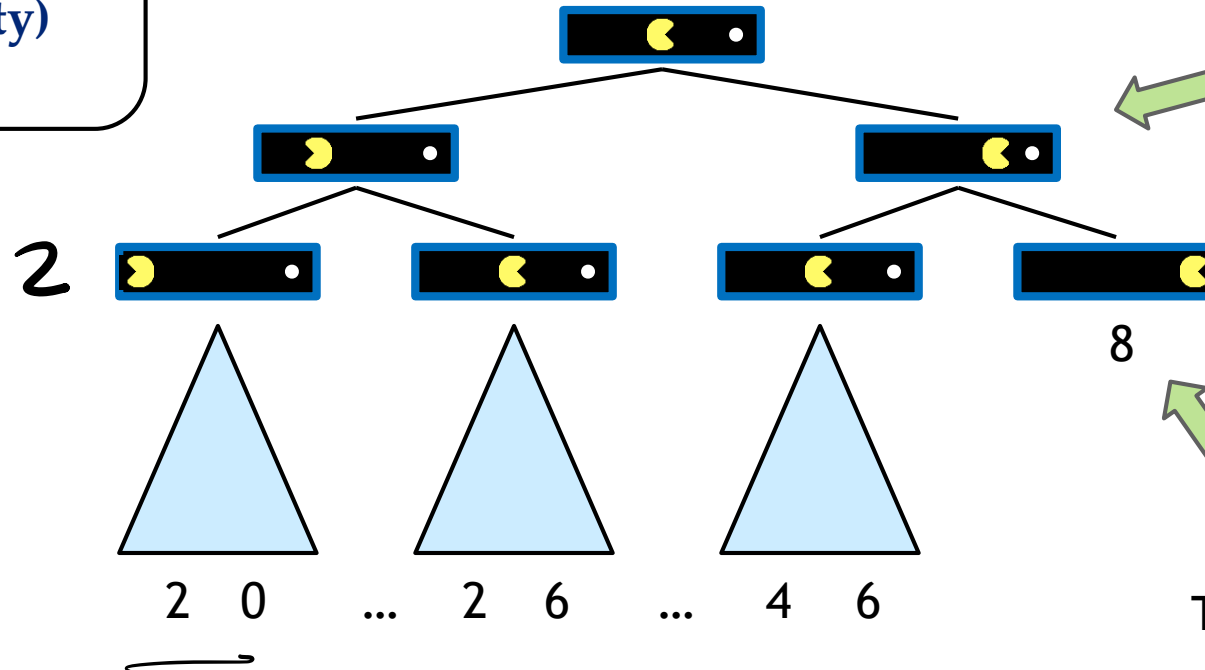
- Agents have opposite utilities (values on outcomes)
- A single score that one maximizes and the other minimizes
- Adversarial, pure competition

General Games

- Agents have independent utilities (values on outcomes)
- Cooperation, indifference, competition, and more are all possible

Value of a State

Value of a state:
The best achievable
outcome (utility)
from that state



Non-Terminal States:

$$V(s) = \max_{s' \in \text{children}(s)} V(s')$$

Terminal States:

$$V(s) = \text{known}$$

Adversarial Game Trees

