

Introduction to Artificial Intelligence

Lecture 17: Search and games I

November 3, 2025



Review: optimality of A^* search

- **Recall:** A^* search finds the next state based on cost function $f(n) = h(n) + g(n)$
 - $h(n)$ is a heuristic estimate of reaching goal state from state n
 - $g(n)$ is the cost from starting state to state n
- **Claim:** The solution found by A^* -search is optimal if the heuristic $h(n)$ is *admissible*, meaning that $\forall n, h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost from n
- **Assumptions**
 - $h(n)$ is never an overestimate of the true cost from node n to a goal node
 - Require $h(n) \geq 0$, so $h(G) = 0$ for any goal G



Proof: A^* search is optimal if $h(n)$ is admissible

1. At every iteration there is **at least one** node on the “frontier” that lies on **some** optimal path from the start to a goal
 - Base case: the starting node is on the shortest path
 - Induction step: suppose the frontier currently intersects with an optimal path at node n ; let n' denote the successor of n on the optimal path toward the goal state
 - If n is not popped out, then n is still in the “frontier”
 - Otherwise, n' must be inserted into “frontier” as it cannot be pruned since $f(n') \leq C^*$
2. Thus, the minimum f -value in the “frontier” must be at most C^* : take n in the “frontier,” we have

$$\min_{m \in \text{frontier}} f(m) = f(n) = g(n) + h(n) \leq g^*(n) + h^*(n) = C^*$$

where C^* is the cost on the minimum-cost path



Proof: A^* search is optimal if $h(n)$ is admissible

3. Finally, suppose the search pops out the goal state G from “frontier.”
 - We have that $h(G) = 0$, so $f(G) = g(G)$

$$g(G) = f(G) = \min_{m \in \text{frontier}} f(m) \leq C^*$$

No solution can cost less than C^* . As a result, the path that A^* search has found is optimal



Lecture plan

- **Search and games**
 - Example games
 - Minimax game
 - Speed up search: Evaluation functions; Pruning
- Survey of major AI techniques



AI and playing games

- **Playing games is perhaps the most successful application of AI techniques**
- **Historical Milestones**
 - 1994: **Chinook** defeats world checkers champion
 - 1997: **DeepBlue** beats Garry Kasparov at chess
 - 2016: **AlphaGo** defeats Lee Sedol at Go
 - 2019: **AlphaStar** masters StarCraft II



A simple game

- Example: game 1

You choose one of the three bins

I choose a number from that bin

Your goal is to maximize the chosen number
(Which one do you choose?)

A

-50

50

B

1

3

C

-5

15



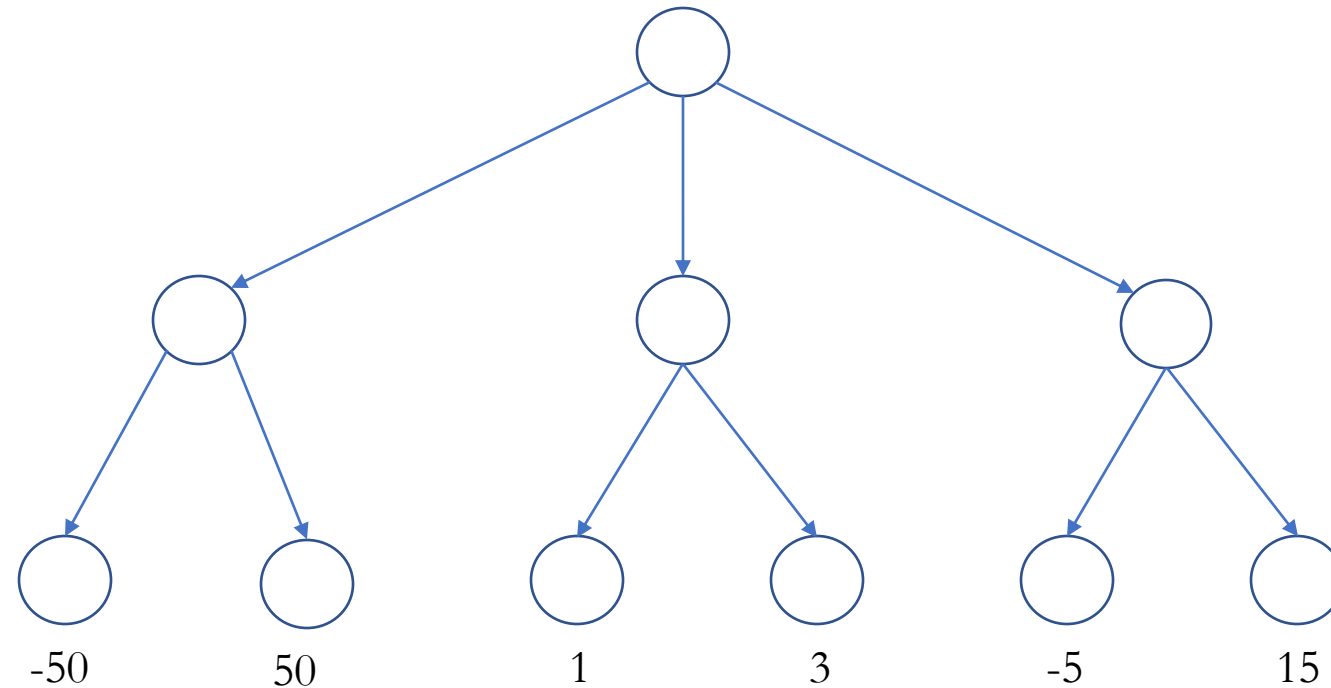
Best response strategy

- Which bin should you pick?
 - Note that this depends on your mental model of the other player (which is unknown)
- If this is a **cooperative game**, then you should pick A in hopes of getting 50
- If this is an **adversarial game**, then you should pick B as to guard against the worst case
- If this is a **stochastic game**, then pick C so that the expected reward is highest



Game tree

- Game tree representation
 - Each node is a decision point for a player
 - Each root-to-leaf path is a possible outcome of the game



Game tree

- Just as in search problems, we use a tree to describe the possibilities of the game. This lets us connect game trees with search techniques
- We can also think of a game graph to capture the fact that there are multiple ways to arrive at the same game state
 - However, our algorithms will operate on the tree rather than the graph since games generally have enormous state spaces



Two-player zero-sum games

- Players = {agent, opponent}
- **Definition: two-player zero-sum game**
 - s_{start} : starting state
 - Action(s): possible actions from state s
 - Successor(s, a): resulting state if choose action a in state s
 - IsEndState(s): whether s is an end state (game over)
 - Utility(s): agent's utility for end state s
 - Player(s) \in Players: player who controls state s (e.g., Chess/Go)



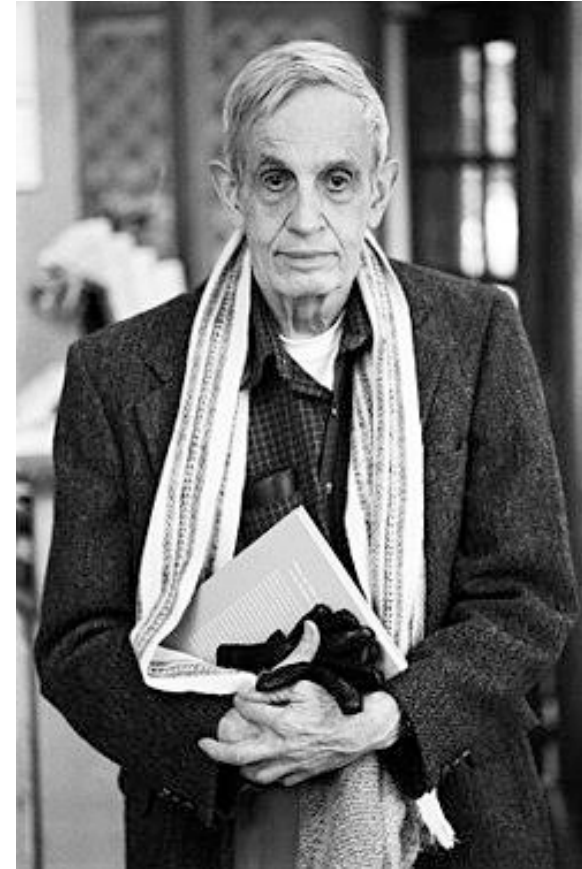
Two-player zero-sum games

- Focus on games in which people take turns and the state of the game is fully-observed
- Examples: chess
- Counter examples:
 - Rock-paper-scissors (in which people take turns)
 - Poker (where you don't know the other players' hands)



History of game theory

- Much of the concepts emerged post world war II



- **Game theory remains a central topic in microeconomics today**



Example: chess



- Players = {white, black}
- State s : (position of all pieces, whose turn it is)
- Actions(s): legal chess moves that Player(s) can make
- IsEnd(s): whether s is checkmate or draw
- Utility(s): $+\infty$ if white wins, 0 if draw, $-\infty$ if black wins



Example: chess

- Chess is a canonical example of a two-player zero-sum game
 - In chess, the state must represent the position of all pieces, and importantly, whose turn it is (white or black)
- Hence, we assume white is the agent and black is the opponent
- Characteristics of games
 - All the utility is at the end state
 - Different players in control at different states



The halving game

- **Problem: halving game**

- Start with a number N
- Players take turns either decrementing N or replacing it with $\text{int}\left(\frac{N}{2}\right)$
- The player that is left with 0 wins



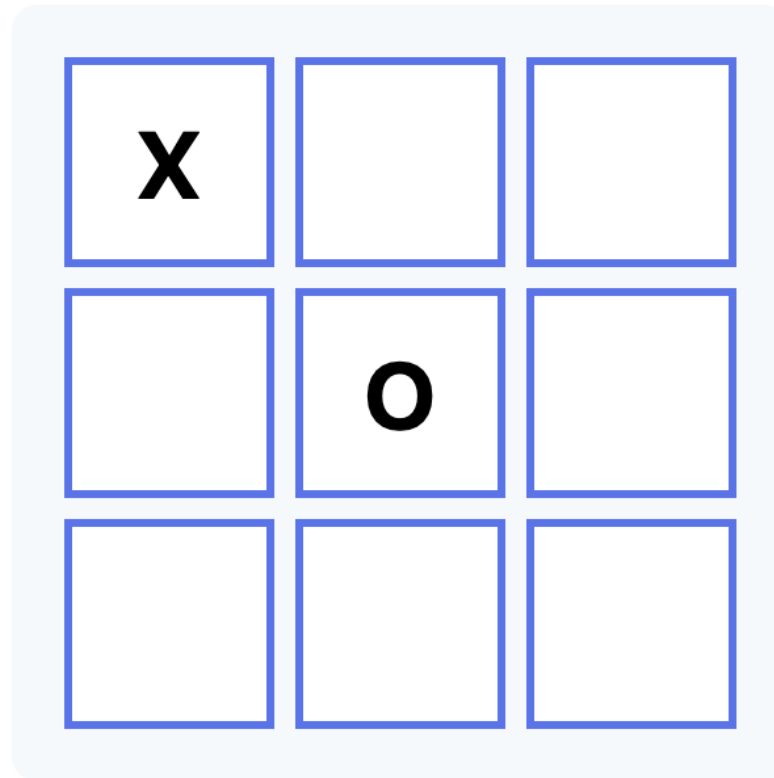
Policies

- **Deterministic policies:** $\pi_p(s) \in \text{Action}(s)$
 - Action that player p takes in state s
- **Stochastic policies** $\pi_p(s, a) \in [0,1]$
 - Probability of player p taking action a in state s



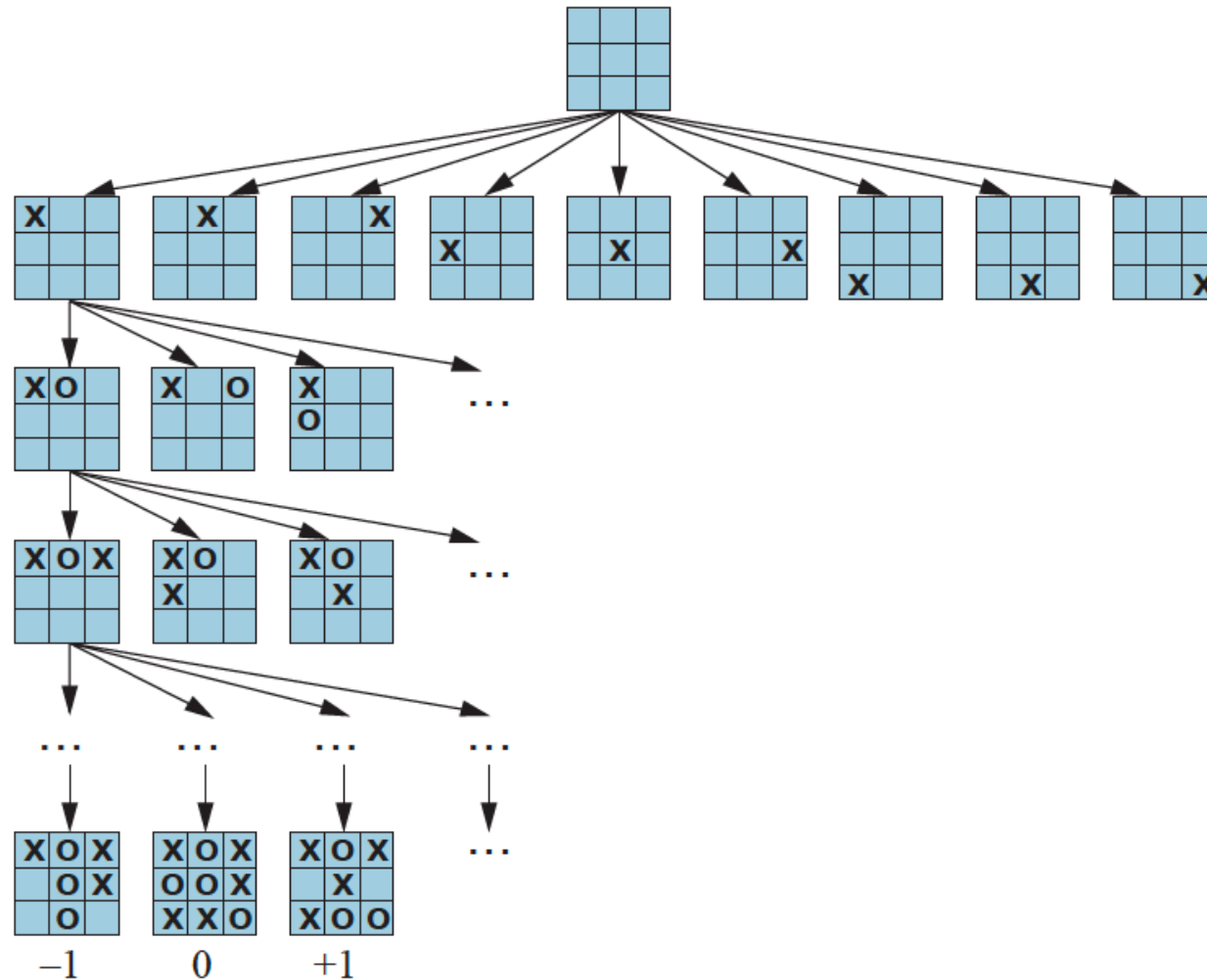
Tic-tac-toe

- Two players who take turns marking the spaces in a three-by-three grid, one with Xs and the other with Os
- A player wins when they mark all three spaces of a row, column, or diagonal of the grid



Game tree illustration for tic-tac-toe

- For tic-tac-toe the game tree is relatively small (but for chess the space is very large)



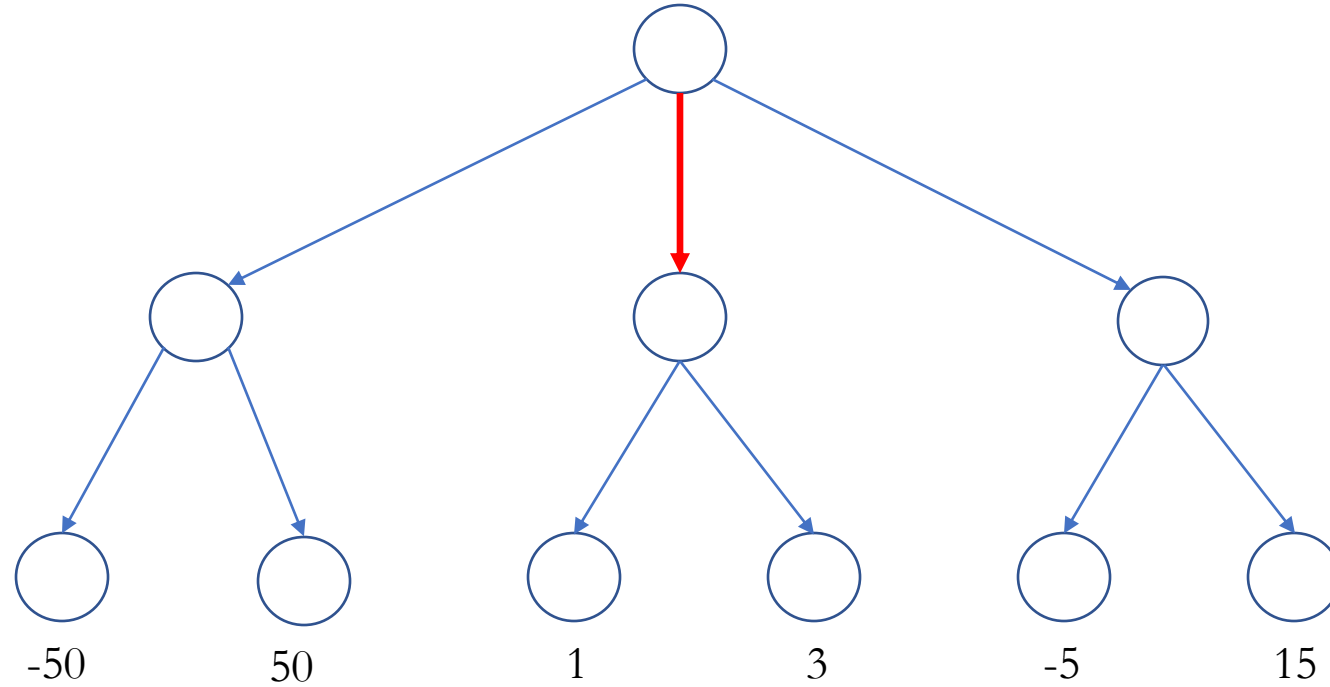
Minimax game

- **Problem:** don't know opponent's policy
- **Approach:** assume the worst case



Minimax example

- Extracting minimax policies



Minimax policies

- Derived from a recurrence relation
- $V_{minimax}(s) =$
 - Utility(s), if s is the terminal state
 - $\max_{a \in Actions(s)} V_{minimax}(Succ(s, a))$, if $Player(s) = agent$
 - $\min_{a \in Actions(s)} V_{minimax}(Succ(s, a))$, if $Player(s) = opponent$



Bottleneck of minimax policies

- In general, there are still too many possibilities
- Two plausible solutions:
 - **Evaluation functions:** use domain-specific knowledge, compute approximate answer
 - **Alpha-beta pruning:** general-purpose, compute exact answer



Evaluation functions

- Estimate how good a position is without playing to the end!
- **Definition:** An evaluation function $Eval(s, p)$ is a (possibly very weak) estimate of the expected utility of the value $V_{minimax}(s)$
- We can construct evaluation functions from domain knowledge



Example of evaluation functions



- $\text{Eval}(s) = \text{material} + \text{mobility} + \text{king-safety} + \text{center-control}$
 - Material: weighted function of king, queen, rooks, bishops, knights, pawns, etc
 - Mobility: number of legal moves

Using evaluation functions

- By using an evaluation function, we can cut down the search space by pruning search directions that have a high cost



- On the other hand, there is no guarantee (unlike A^*) on the error from approximation



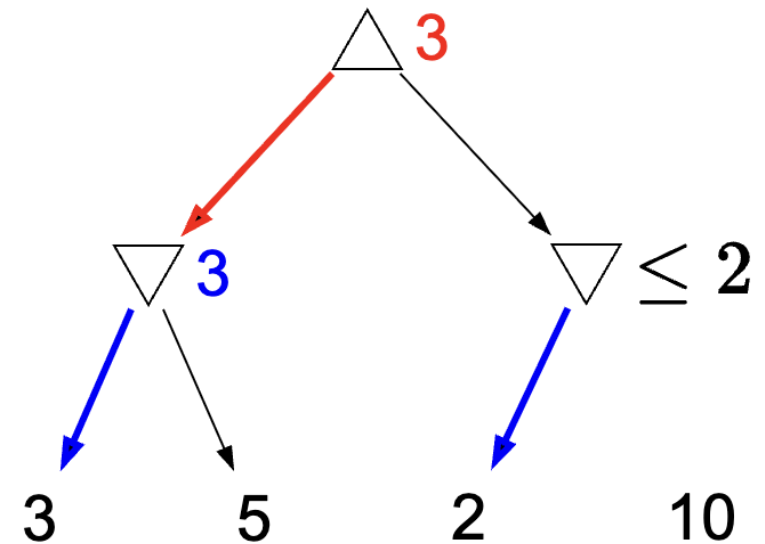
Pruning

- Ideas: Some branches can't possibly affect the final decision
 - Maintain lower and upper bounds on values
 - If intervals don't overlap non-trivially, then can choose optimally without further work



An example of pruning game trees

- Once see 2, we know that value of right node must be ≤ 2
- Root computes $\max(3, \leq 2) = 3$
- Since branch doesn't affect root value, we can safely prune



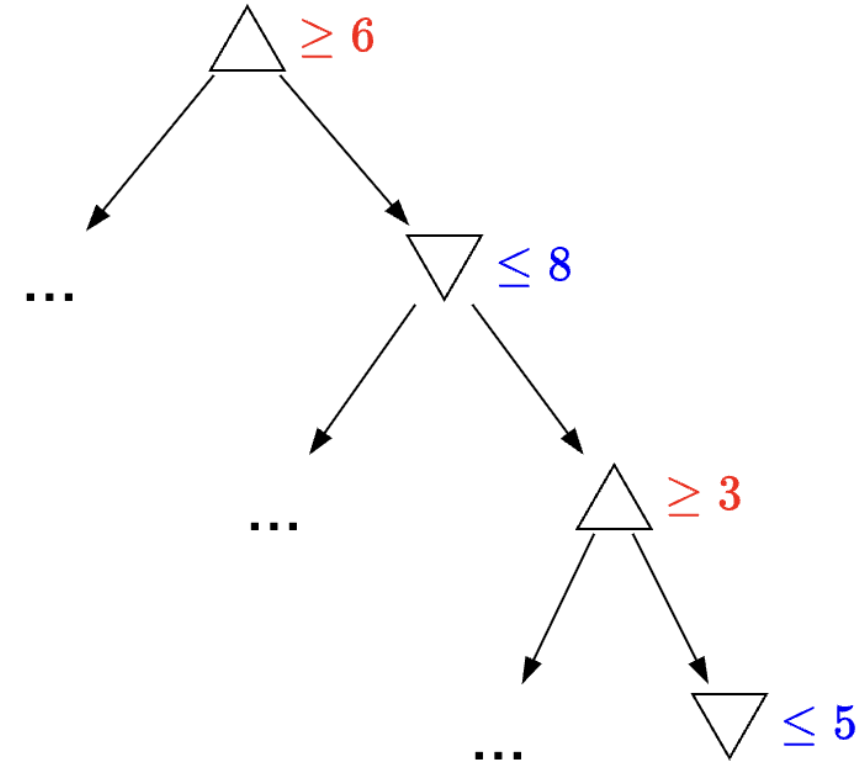
Another example of pruning in shortest path search

- Imagine traversal a graph $G = (V, E)$
- Starting node is s
- We store a search frontier from s , which includes the best path and its cost to each node in the frontier
 - Frontier = $\{(v_1, c_1), (v_2, c_2), \dots, (v_k, c_k)\}$
- During every search step
 - Expand from one node in the frontier: if the found path is worse than current node, **prune the search**



Alpha-beta pruning

- Alpha-beta pruning instantiates this idea of search in minimax games
- Maintain a lower bound on value of max node
- Main an upper bound on value of min node



Example of alpha-beta pruning

- If the interval of the current node does not non-trivially overlap the interval of every one of its ancestors, then we can prune the current node
- In the example, we've determined the root's node must be at least 6
- Once we get to the node on at level 4 and determine that node is at most 5, we can prune the rest of its children since it is impossible that this node will be on the optimal path



Lecture plan

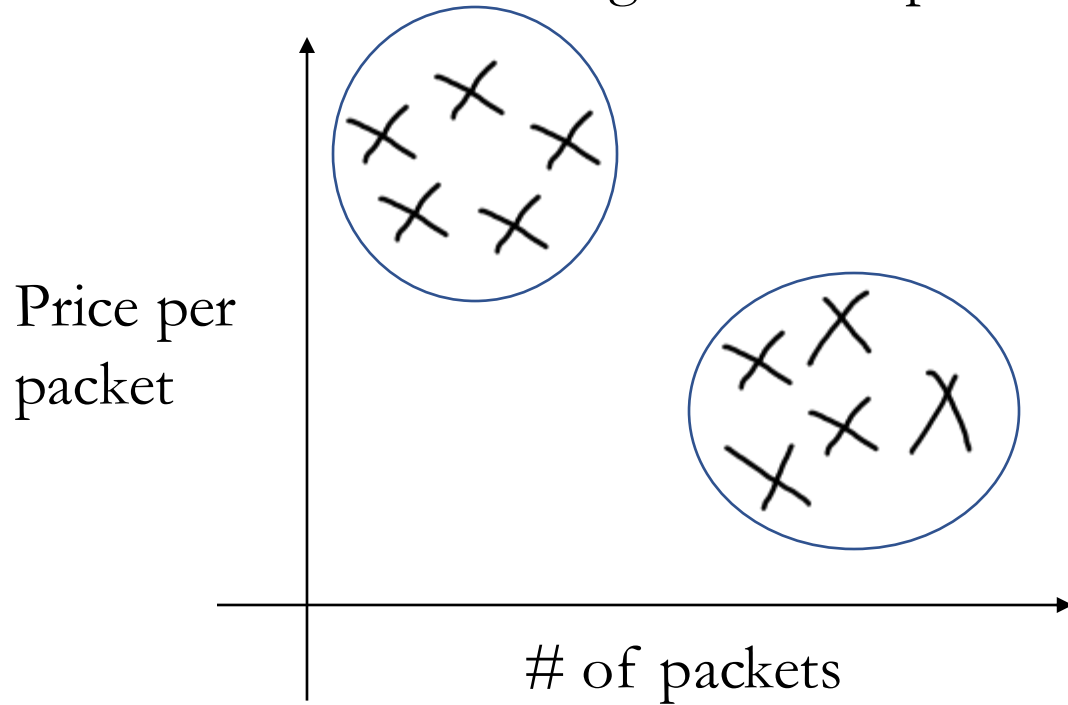
- Adversarial search and games
- **Survey of major AI techniques**



Survey of AI techniques: Unsupervised learning

Given data (without any specific desired output labels), find something interesting about the data

Clustering Potato chip sales



Finding cats from unlabeled YouTube videos



Transfer learning

Car detection



100,000 images

Golf cart detection

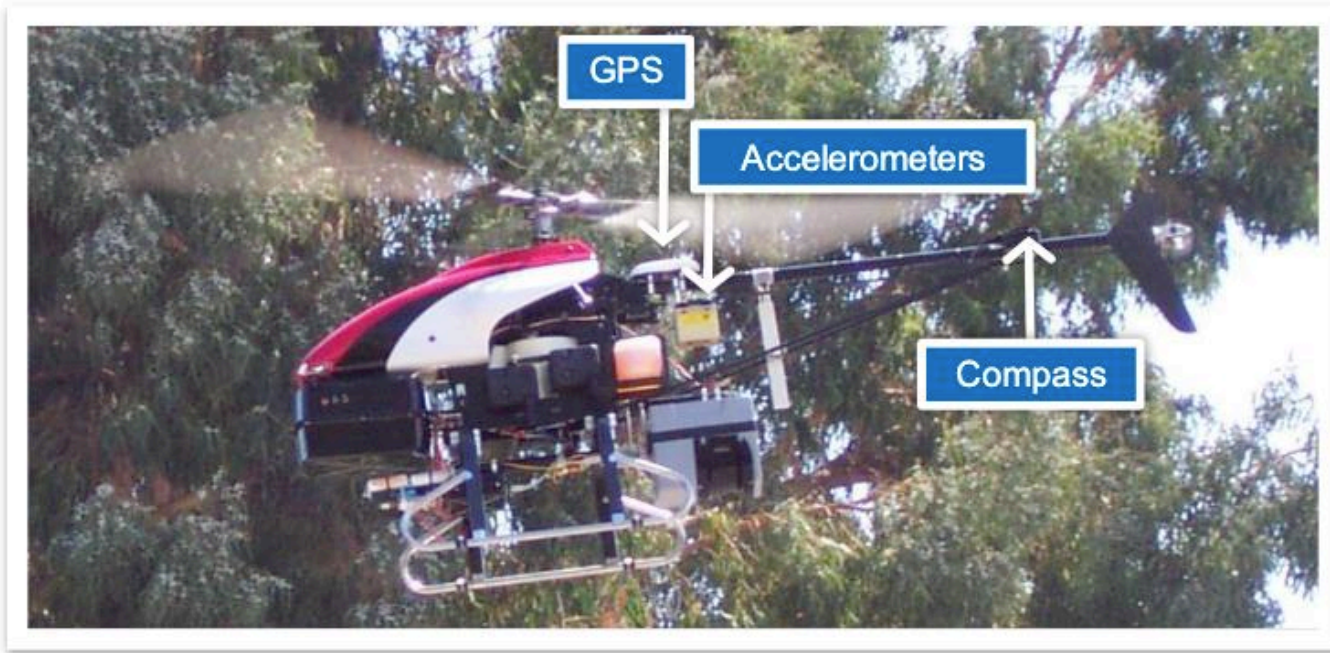


100 images

Learn from task A, and use knowledge to help on task B



Reinforcement learning

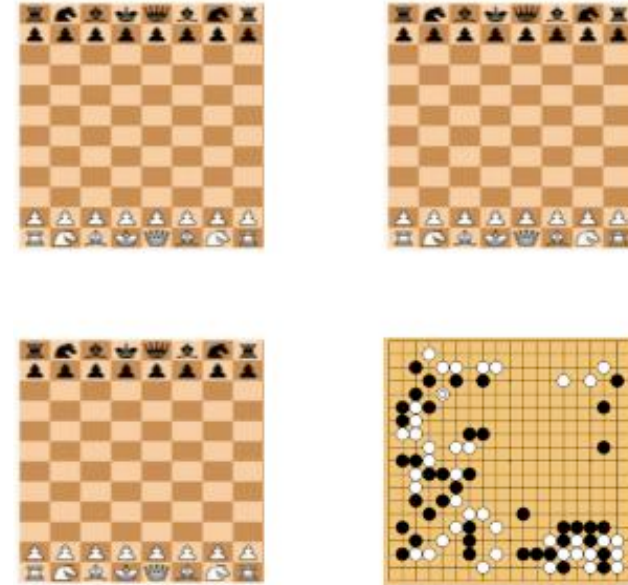


Use a “reward signal” to tell the AI when it is doing well or poorly. It automatically learns to maximize its rewards

Reinforcement learning



Auto pilot



Playing games

Use a “reward signal” to tell the AI when it is doing well or poorly. It automatically learns to maximize its rewards

GANs (Generative Adversarial Network)

- Synthesize new images from scratch



[Source: Karras et al. (2018). Progressive Growing of GANs for Improved Quality, Stability, and Variation]



Knowledge graph

The screenshot shows a Google search for 'ada lovelace'. The search results include:

- Ada Lovelace - Wikipedia**: Auguste Ada King, Countess of Lovelace was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical ...
Resting place: Church of St. Mary Magdalene. Spouse(s): William King-Noel, 1st Earl of Lovelace
Known for: Mathematics, computing
Charles Babbage - Analytical Engine - William King-Noel, 1st Earl of - Lady Byron
- Ada Lovelace: Founder of Scientific Computing**: <https://www.edsc.edu/ScienceWomen/lovelace.html>
ADA BYRON, COUNTESS OF LOVELACE ... Ada Byron was the daughter of a brief marriage between the Romantic poet Lord Byron and Anne Isabella ...
- People also ask**:
 - What is Ada Lovelace famous for?
 - What did Ada Lovelace invent and what impact it had?
 - When did Ada Lovelace invent the computer?
 - What is Ada Lovelace Day?
- Ada Lovelace | Biography & Facts | Britannica.com**: <https://www.britannica.com/biography/Ada-Lovelace>
Jan 3, 2019 - Ada Lovelace, in full Ada King, countess of Lovelace, original name Augusta Ada Byron, ...

The knowledge panel on the right displays:

- Ada Lovelace** (Mathematician)
- Augusta Ada King, Countess of Lovelace was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical general-purpose computer, the Analytical Engine. [Wikipedia](#)
- Born**: December 10, 1815, London, United Kingdom
- Died**: November 27, 1852, Marylebone, United Kingdom
- Spouse**: William King-Noel, 1st Earl of Lovelace (m. 1835-1852)
- Children**: Anne Blunt, 15th Baroness Wentworth, [MORE](#)
- Parents**: Lord Byron, Lady Byron
- Known for**: Mathematics, Computing
- People also search for**: [View 15+ more](#)

Ada Lovelace	
Born	Dec 10, 1815
Died	Nov 27, 1852
Bio	English mathematician and writer ...

