Algorithm Design and Analysis

Game Theory (and applications to algorithm analysis!)

Welcome-back reminders

- **Midterm two is Tuesday 25th March (Week 10) at 7:00pm**
  - Conflict? We will post a form on Ed for you to apply for make-up exam
- Programming Homework 3 is coming out later today, due next week on Saturday
- Homework 6 (oral) is coming later today, oral presentations next week Wednesday - Friday

Mid-semester feedback feedback

- Most common criticism: **Style grading**
  - Style grading will still exist, however,
  - we will try to make it more lenient
- Homework timeline:
  - HW6 and Programming 3 will release early (today),
  - Programming 3 due slightly later

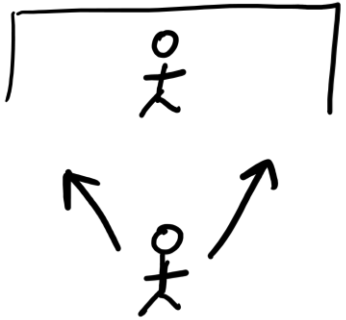
Roadmap for today

- Two-player zero sum games
- ***Minimax-optimal strategies*** for two-player zero-sum games
- Using game theory to analyze randomized algorithms!

Game theory

- Game theory is the study of models of ***strategic interactions*** between agents/players
- Each agent/player must ***choose an action***, and wants to figure out the "best" possible action ***without knowing which action the other players will make***
- The combined actions of the players gives each player a ***payoff***. Players want their own payoff to be as high as possible.

Game Theory: Example



Example (Soccer / "shooter-goalie game"):

- 2 players: The shooter and the goalie
- Shooter has two possible actions: kick left or kick right (say $\{L, R\}$)
- Goalie has two possible actions: dive left or dive right (say $\{L, R\}$)
- If the shooter and goalie both choose L or both choose R, the goalie is happy (they block the ball)
- If the shooter and goalie pick different directions, the shooter get the goal

Can encode the game as a *payoff matrix* ➡

shooter \ Goalie	L	R
L	(-1, 1)	(1, -1)
R	(1, -1)	(-1, 1)

The Payoff Matrix

- In a 2-player game, the payoff is a pair (r, c), the payoffs to the row player and column player respectively

Definition: If for every entry $r + c = 0$, the game is a ***zero-sum game***

- For zero-sum games, we often just write the row-payoff matrix R since we can infer the column payoffs C (since $R + C = 0$)

		goalie L	goalie R
shooter	L	(−1, 1)	(1, −1)
	R	(1, −1)	(−1, 1)

$$R = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \qquad C = -R$$

Strategies

- How should the players play if they want a high payoff?

- **Definition *(Pure strategy)***: Choose a single action deterministically.
  - Row player chooses an action i and column player chooses an action j
  - Payoff is $R_{i,j}$ for the row player and $C_{i,j}$ for the column player

- **Definition *(Mixed strategy)***: Choose an action randomly!
  - The row player has a probability distribution: $p_i \in [0,1]$ for each i
  - The column player has a probability distribution: $q_j \in [0,1]$ for each j

Expected payoff

- When playing a mixed strategy, we get an ***expected payoff***
- Define $V_R(\mathbf{p}, \mathbf{q})$ as the expected row payoff
- Define $V_C(\mathbf{p}, \mathbf{q})$ as the expected column payoff

$$V_R(p, q) = \sum_{i,j} p_i q_j \cdot R_{ij}$$

$$V_C(p, q) = \sum_{i,j} p_i q_j \cdot C_{ij}$$

Expected payoff example

- Shooter-goalie game with mixed strategies $\mathbf{p} = \left(\frac{1}{2}, \frac{1}{2}\right)$ and $\mathbf{q} = \left(\frac{1}{2}, \frac{1}{2}\right)$

		goalie	
		L	R
shooter	L	$(-1, 1)$	$(1, -1)$
	R	$(1, -1)$	$(-1, 1)$

$$\frac{1}{2}\frac{1}{2}\cdot(-1) + \frac{1}{2}\cdot\frac{1}{2}\cdot(1) + \frac{1}{2}\frac{1}{2}(-1) + \frac{1}{2}\frac{1}{2}(1) = 0$$

- Shooter-goalie game with mixed strategies $\mathbf{p} = \left(\frac{3}{4}, \frac{1}{4}\right)$ and $\mathbf{q} = \left(\frac{3}{5}, \frac{2}{5}\right)$

$$\frac{3}{4}\cdot\frac{3}{5}(-1) + \frac{3}{4}\frac{2}{5}(1) + \frac{1}{4}\cdot\frac{3}{5}(1) + \frac{1}{4}\cdot\frac{2}{5}(-1) = -\frac{1}{10}$$

Lower bound strategies

- The row player wants to pick their strategy $\mathbf{p}^*$ to maximize the expected payoff **over all possible strategies** q of the column player.
- Given a strategy $\mathbf{p}$ for the row player, define the ***lower bound*** as

$$\mathrm{lb}(\mathbf{p}) := \min_{\mathbf{q}} V_R(\mathbf{p}, \mathbf{q})$$

- The row player can guarantee this expected payoff regardless of the column player's strategy, so the lower bound to the row player is:

$$\mathrm{lb}^* := \max_{\mathbf{p}} \mathrm{lb}(\mathbf{p}) = \max_{\mathbf{p}} \min_{\mathbf{q}} V_{\mathrm{R}}(\mathbf{p}, \mathbf{q})$$

Upper bound strategies

- Similarly, given a strategy $\mathbf{q}$ for the column player, there is some highest-payoff response from the row player

$$\mathrm{ub}(\mathbf{q}) \coloneqq \max_{\mathbf{p}} V_R(\mathbf{p}, \mathbf{q})$$

- This is an **upper bound** on the payoff that the row player can achieve.

- The worst possible upper bound for the row player is therefore

$$\mathrm{ub}^* \coloneqq \min_{\mathbf{q}} \mathrm{ub}(\mathbf{q}) = \min_{\mathbf{q}} \max_{\mathbf{p}} V_R(\mathbf{p}, \mathbf{q})$$

Usefulness of lower/upper bounds

- We know that $\mathrm{lb}(\mathbf{p}) \leq \mathrm{ub}(\mathbf{q})$ by definition
- This property can be used to **prove that a strategy is optimal**

- Like flows/cuts! If we find a flow with value F and a cut of capacity C, since $F \leq C$ for all F and C this proves that F was a max flow.
- That is, if we can find a $\mathbf{p}$ and a $\mathbf{q}$ such that $\mathrm{lb}(\mathbf{p}) = \mathrm{ub}(\mathbf{q})$ then this is a proof that these strategies are optimal!

Important lemma: pure response

- Suppose we consider a fixed row strategy $\mathbf{p}$ and want to compute the lower bound, i.e., the best counter-strategy $\mathbf{q}$ from the column player.

Theorem: To evaluate $\mathrm{lb}(\mathbf{p})$, we can assume the column player plays a **pure strategy**

$$\mathrm{lb}(\mathbf{p}) \coloneqq \min_{\mathbf{q}} V_R(\mathbf{p},\mathbf{q}) = \min_j \sum_i p_i R_{i,j}$$

$$\min_q V_R(p,q) = \min_q \sum_{i,j} p_i q_j R_{ij} = \min \sum_j q_j \left(\sum p_i R_{ij}\right)$$

constant

$$\min \underset{q_1}{\underline{\quad}} \times \text{const} + \underset{q_2}{\underline{1}} \times \text{const} + \ldots$$

1/10   -1/5   2/3

Example: Shooter-Goalie game

- Suppose we (row player, i.e., shooter) play $\mathbf{p} = \left(\frac{1}{2}, \frac{1}{2}\right)$.
  - If Goalie plays L: payoff = $\frac{1}{2}(1) + \frac{1}{2}(-1) = 0$
  - If Goalie plays R: payoff = $\frac{1}{2}(-1) + \frac{1}{2}(1) = 0$
- So, $\mathrm{lb}(\mathbf{p}) =$ $\min(0, 0) = 0$

Example: Shooter-Goalie game

- Suppose the column player (goalie) plays $\mathbf{q} = \left(\frac{1}{2}, \frac{1}{2}\right)$
  - If Shooter plays L: payoff = $\frac{1}{2}(1) + \frac{1}{2}(-1) = 0$
  - If Shooter plays R: payoff = $\frac{1}{2}(-1) + \frac{1}{2}(1) = 0$
- So, $\mathrm{ub}(\mathbf{q}) = \max(0, 0) = 0$

$$lb(p) = ub(q) = 0$$

Von-Neumann's Minimax Theorem

- Coincidentally, we had $\mathrm{lb}^* = \mathrm{ub}^*$
- Not a coincidence!!

Theorem: Given a finite two-player zero-sum game with payoff matrices $R = -C$,

$$\mathrm{lb}^* = \max_{\mathbf{p}} \mathrm{lb}(\mathbf{p}) = \max_{\mathbf{p}} \min_{\mathbf{q}} V_R(\mathbf{p}, \mathbf{q}) = \min_{\mathbf{q}} \max_{\mathbf{p}} V_R(\mathbf{p}, \mathbf{q}) = \min_{\mathbf{q}} \mathrm{ub}(\mathbf{q}) = \mathrm{ub}^*$$

- Proof in a few lectures from now!
- **Useful fact**: If you play a minimax strategy, you can even tell your opponent your strategy without losing anything!

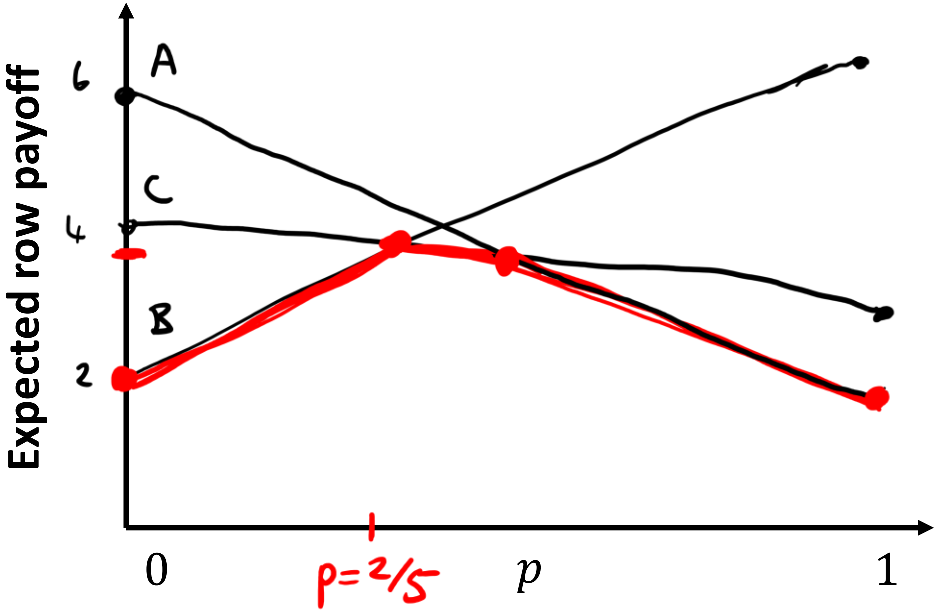
Techniques for solving games

- **Strategy #1**: Guess and bound. We did this before by finding a pair of strategies $\mathbf{p}$ and $\mathbf{q}$ such that $\mathrm{lb}(\mathbf{p}) = \mathrm{ub}(\mathbf{q})$

- **Strategy #2**: Graph and optimize. We can try to do this if guessing the optimal value is too hard. Requires the game to have two rows.

Graph and optimize

		column player		
		A	B	C
p row	1	2	6	3
$(1-p)$ player	2	6	2	4



- Say the row player plays row 1 with probability p and row 2 with probability $1-p$

Expected payoff if column plays A: $2p + (1-p)\cdot 6 = 6-4p$

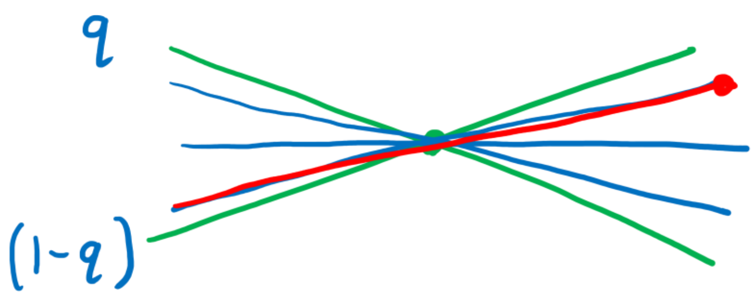
Expected payoff if column plays B: $6p + (1-p)2 = 2+4p$

Expected payoff if column plays C: $3p + (1-p)4 = 4-p$

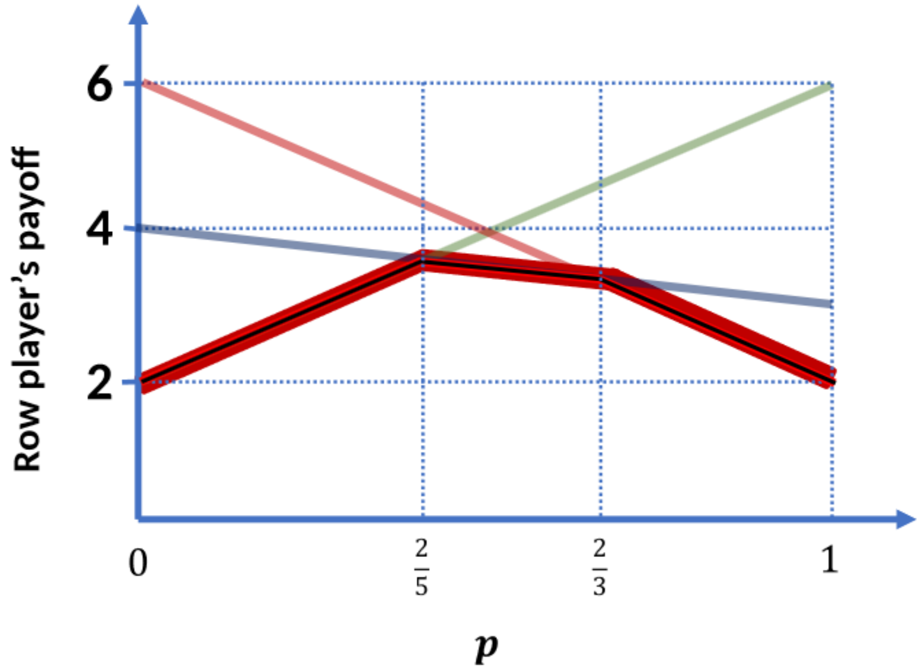
Expected minimum payoff: $\min(6-4p, 2+4p, 4-p)$

payoff $= 3 + 3/5$

Graph and optimize



		column player		
		A	**B**	**C**
row	**1**	2	6	3
player	**2**	6	2	4



- Given the row player strategy $\left(\frac{2}{5}, \frac{3}{5}\right)$ with payoff $3 + \frac{3}{5}$, how can we find the column player strategy?

$$q(2+4p) + (1-q)(4-p) \equiv 3 + 3/5$$

$$= 5pq - 2q - p + 4 \equiv 3 + 3/5$$

Choose q to make p coeff $\equiv 0$

$$(5q-1)p - 2q + 4 \equiv 3 + 3/5$$

$$\boxed{q = 1/5} \qquad -2 \cdot \tfrac{1}{5} + 4 = \underline{3 + 3/5}$$

Additional tricks

- Removing ***dominated*** rows or columns. Say the (row) payoff matrix is:

$$\begin{bmatrix} 2 & 3 & 1 \\ 3 & 3 & 5 \\ 7 & 1 & 8 \end{bmatrix}$$

- ***Convex combinations*** of rows or columns

$$\begin{bmatrix} 10 & 0 \\ 0 & 10 \\ 3 & 3 \end{bmatrix} \begin{matrix} \frac{1}{2} \\ \frac{1}{2} \\ \end{matrix}$$

Applications

Lower bounds for randomized algorithms!

- Earlier in the class we proved a worst-case $\Theta(n \log n)$ lower bound for sorting in the comparison model, for **deterministic** algorithms

Theorem: Any randomized sorting algorithm in the comparison model requires at least $\Omega(\log n!) = \Omega(n \log n)$ **expected** comparisons in the worst case.

- We will prove this using game theory!!

Turning algorithms into game theory

- The "game" is played by **the adversary (the row player)** whose actions are to choose an input for the problem, and **the algorithm designer (the column player)** who chooses an algorithm.
- The payoff is the cost (number of comparisons)
- A deterministic algorithm is a pure strategy

Key idea: A randomized algorithm is a mixed strategy! In other words, a randomized algorithm is the same thing as randomly picking from all possible deterministic algorithms.

Turning algorithms into game theory

- Say the adversary (row player) picks a pure strategy (an input) i and the column player picks a mixed strategy (= randomized algorithm) $\mathbf{q}$.
- A complexity lower bound is

Von-Neumann's Theorem

$$\min_{\substack{\text{randomized} \\ \text{algorithms } \mathbf{q}}} \max_{\text{inputs } i} V_R(i, \mathbf{q}) = \max_{\substack{\text{input} \\ \text{distributions } \mathbf{p}}} \overset{\text{min}}{\cancel{\max}}_{\substack{\text{deterministic} \\ \text{algorithms } j}} V_R(\mathbf{p}, j)$$

- So, we can instead find a mixed strategy (= distribution of inputs) $\mathbf{p}$ for the adversary (row player) such that every column (deterministic algorithm) has high cost. What is this called??

Turning algorithms into game theory

Lemma: For any randomized algorithm, its expected worst-case cost is at least as large as the average cost of the best deterministic algorithm over any input distribution.

- So, we just need to prove that the **average cost** of any deterministic algorithm for sorting is at least $\Omega(n \log n)$.

Exercise (or see notes)

Hint: Use decision trees and argue that most of the leaves have high depth

Summary of today

- We learned about two-player zero-sum games
- We saw how to find minimax-optimal strategies
  - Guess and bound technique using lb and ub
  - Graph and optimize (for two-row games)
  - Eliminating dominated strategies
- The ***minimax theorem*** is a powerful tool
- We can prove worst-case lower bounds for randomized algorithms by converting them into average-case lower bounds for deterministic algorithms!