

# Evolution of Cooperation

Alex Joseph, Marc Domingo, Jaden Suh, Austin Simpson

# What is the Prisoner's Dilemma?

- Two prisoners must choose between remaining silent or betraying the other
  - Cooperate (C) vs Defect (D)
- 4 possible outcomes to the game:
  - Both cooperate: 2 years in prison for both
  - Both defect: 5 years in prison for both
  - 1 cooperate and 1 defect: Defector goes free, cooperator serves 10 years in prison
- Rationally, cooperation makes no sense
  - Many human players choose to cooperate anyway

		Prisoner B	
		Prisoner B stays silent ( <i>cooperates</i> )	Prisoner B betrays ( <i>defects</i> )
Prisoner A	Prisoner A stays silent ( <i>cooperates</i> )	Each serve 2 years	Prisoner A: 10 years Prisoner B: goes free
	Prisoner A betrays ( <i>defects</i> )	Prisoner A: goes free Prisoner B: 10 years	Each serve 5 years

# Why is it Interesting?

- Examines human biases towards cooperation
- Why do we trust each other even when it makes no logical sense to do so?
- When do the benefits of betrayal outweigh the negatives for humans?
- *The Dark Knight* (2008)



# Literature Review

# Regarding the Prisoner's Dilemma

## Effective Choice in the Prisoner's Dilemma (Axelrod 1980)

- Social Psychology
  - The Prisoner's Dilemma encapsulates the struggle between individual and group rationality in humans
    - Individual Rationality of selfishness to be able to reap the most benefit
    - Group Rationality of mutual cooperation to benefit all actors
  - Applications of Iterative variations of the Prisoner's Dilemma to analyze the concept of "Effective Choice" in:
    - Relationships between conflicting entities (Axelrod used the US and Soviet Union)
    - The effects of Westernization in Central Africa
    - The difference in effects of learning using abstract vs concrete thinking styles

# An Overview of the First Axelrod Tournament

## Effective Choice in the Prisoner's Dilemma (Axelrod 1980)

- Tournament Details
  - 15 Submitted Strategies by economists, psychologists, sociologists, mathematicians, and political scientists
  - Tournament itself was styled as a Round Robin Competition, meaning that each strategy competed against each other at some point of the tournament
  - Each match between strategies were 200 rounds
  - Point Breakdown:
    - 3 points for Mutual Cooperation (C, C)
    - 1 point for Mutual Defection (D, D)
    - 5 points for a successful Defection (D, C)
    - 0 points for an unsuccessful Cooperation (D, C)

# A Look Into Successful Strategies in the First Tournament

## Effective Choice in the Prisoner's Dilemma (Axelrod 1980)

- Strategies that were the most successful in the Tournament were:
  - **Nice** in that they did not Defect (D) during the first turn or were not the first strategy to Defect (D) before the match ended
  - **Effective when in a match with special strategies**
    - DOWNING was a strategy that attempted to predict the opponent's move based on its own previous move
    - GRAASKAMP was a strategy that played Tit-For-Tat for the first 50 turns, Defects (D) once, and then attempts to determine whether it's playing against itself, Tit-For-Tat or some similar variant, or the Random strategy
  - **Forgiving** in that they did not excessively punish an opponent after the opponent Defects (D)
- Out of all of the strategies, the basic Tit-For-Tat strategy won the tournament, even against opponents who made variations to attempt to improve Tit-For-Tat
  - Strategy Cooperates (C) with the opponent until the opponent Defects (D). Tit-For-Tat then Defects (D) next turn and goes back to Cooperating (C) following turns

# Observations from other Prisoners' Dilemma Strategy Papers

Probabilistic Tit-for-Tat Strategy vs Nash Equilibrium for Infinitely Repeating Games (Madhumidha et al. 2017)

- Nash Equilibrium - concept that every player within a game has selected a strategy and cannot perform better by changing the strategy
  - When players obtain equal end results in a game or repeated game, Nash Equilibrium becomes less optimal
  - Does not provide conclusive results when multiple solutions to Nash Equilibria are found
- The Tit-for-Tat strategy is not optimal when “noise” caused by players deviating from their typical behavior due to mistakes and misconceptions exists



# Observations from other Prisoners' Dilemma Strategy Papers (Continued)

No Strategy Can Win in the Repeated Prisoner's Dilemma: Linking Game Theory and Computer Simulations (García and Veelen 2018)

- Argues that cooperation is inherently unstable due to mutants and deviations
- Regardless of strategy, no equilibrium is fundamentally more stable than any other

Evolution of cooperation through cumulative reciprocity (Li et al. 2022)

- Traditional models of cooperation use strategies with restricted memory.
- Strategies where players only remember the last encounter they had are mathematically convenient, but they miss important aspects of human reciprocity, where defections can have lasting effects.

Design

# Axelrod API

- Python library created by Dr. Vince Knight at Cardiff University (Wales, UK).
- Research tool for the iterated Prisoner's Dilemma.
  - Includes strategies from literature and Axelrod's tournament.
- Generates reproducible results of iterative round robin tournaments.
- Features tools to model population dynamics with the tournaments.
  - Strategies that perform better reproduce more between iterations
  - Players are randomly removed between iterations.
  - Better performing strategies are more likely to “survive”
- Documentation with examples of the library's capabilities.
- Lack of intuitive output format and difficult to add custom strategies.

# Project Scope

- **Goal: Create a UI that makes the Axelrod API much easier to use for those without coding experience**
  - Curated list of strategies with strategy explanations that remove the need to look at the algorithm
  - Tournament output that is less confusing and presented in a cleaner way
  - User control over various parameters like number of rounds, strategy selected, number of participants, etc.
  - Ability to code a custom strategy and add it to the game
  - Allow user to run multiple tournaments back to back without restarting the program
  - Inclusion of “Advanced Mode” that expands the strategy list and adds more complex parameters for the user to manipulate, as well as more detailed output

# Design Decisions

- Chose to use Python as our programming language
  - Originally had intended on using Java for its greater focus on object-oriented programming and classes, but switched focus to Python upon discovering the exhaustive Axelrod API
- Built off of the Axelrod API as a foundation
  - Originally had intended to create our own tournament program from scratch
  - Discovery of the API meant we had an opportunity to improve upon the large amount of work already done
- User Experience focused direction
  - Experimentation with the API revealed that it's clunky to use and difficult to understand for non-programmers
  - Streamlining the experience would allow for a greater range of access to researchers and enthusiasts

# Workflow & Contributions

- Weekly agile structure used to adapt to project needs on a short-term basis
- Weekly meetings to check in on progress and alter direction/tasks
- Overall contributions:
  - Alex - Project leader, ensured tasks were completed and kept project on track, assisted with output and program logic
  - Marc - Implemented curated strategy list with explanations and assisted with program logic
  - Jaden - Created logic surrounding input of user parameters and chose which parameters to make available
  - Austin - Implemented the ability to add custom strategies and streamlined API output for readability

# Algorithms Involved

- Algorithms classified by properties
  - Depth of memory
  - Randomization (stochasticity)
  - Strategies that are known to increase tournament run-time
- Tit-for-tat: Simple strategy with a memory depth of 1
  - Only looks at the opponent's last move
  - Strategy wasn't dominant in every match it played but scored well on average
  - Variations of Tit-for-tat: Tit-for-2-tats, 2-tits-for-tat, dynamic
- Custom strategies: Creatable by users
  - Chaotic Clairvoyant: Demonstrates higher memory depth usage.
  - FibTitForTat: Tit for Tat with random chance to defect based on fibonacci sequence.

# Results & Future Development

- What do we have to show for it?
  - Much easier to use/simplified version of the Axelrod python API
  - Custom strategy implementation
- How can we continue to develop this project in the future?
  - Sleeker and further streamlined GUI
  - Custom strategy implementation that doesn't require scripting knowledge
  - Further simplifying the output for easier interpretation
  - Adding additional games, like the closed-bag exchange or the Guardian's Dilemma
  - Incorporation of randomized player evolution feature
  - Turning the tournament into a simple game that takes user input
    - Players input their moves each turn. See scores and which strategies their moves were most similar to



Questions?