# Artificial Intelligence Foundation – JC3001

## Lecture 10: Search IV: Adversarial Search I

**Prof. Aladdin Ayesh** (aladdin.ayesh@abdn.ac.uk)

**Dr. Binod Bhattarai** (binod.bhattarai@abdn.ac.uk)

**Dr. Gideon Ogunniye,** (g.ogunniye@abdn.ac.uk)

September 2025

Material adapted from:
Russell and Norvig (AIMA Book): Chapter 5
Russell and Norvig (AIMA Book): Chapter 17/18 (17.1/18.2)
Shoham and Leyton-Brown (Game Theory)

# Course Progression

- Part 1: Introduction
  1. Introduction to AI ✓
  2. Agents ✓
- Part 2: Problem-solving
  1. Search 1: Uninformed Search ✓
  2. Search 2: Heuristic Search ✓
  3. Search 3: Local Search ✓
  4. **Search 4: Adversarial Search**
- Part 3: Reasoning and Uncertainty
  1. Reasoning 1: Constraint Satisfaction
  2. Reasoning 2: Logic and Inference
  3. Probabilistic Reasoning 1: BNs
  4. Probabilistic Reasoning 2: HMMs
- Part 4: Planning
  1. Planning 1: Intro and Formalism
  2. Planning 2: Algos and Heuristics
  3. Planning 3: Hierarchical Planning
  4. Planning 4: Stochastic Planning
- Part 5: Learning
  1. Learning 1: Intro to ML
  2. Learning 2: Regression
  3. Learning 3: Neural Networks
  4. Learning 4: Reinforcement Learning
- Part 6: Conclusion
  1. Ethical Issues in AI
  2. Conclusions and Discussion

# Objectives

- Games
- Non-deterministic search
- Adversarial Search
  - MinMax Search
  - Alpha-Beta Pruning
  - Expectimax

- We have covered a number of search strategies which operate by systematically generating new states and testing them against a goal
- These are typically highly inefficient:
  - Apart from the state space and cost, they do not take any knowledge of the problem into account
  - They represent **uninformed search**
- By using problem specific knowledge (i.e., an **informed search**), we can perform better than the algorithms already encountered

- Idea: select node for expansion based on an **evaluation function**, $f(n)$
  - estimate of “desirability“
  - expand most desirable unexpanded node
- This function could measure distance from the goal, so expand node with lowest evaluation
- Typically implemented via a **priority queue**
- Our search is a best-first one; we expand the node that (we believe) is the best one first

- The A* search is probably the most used type of heuristic search
- It combines the cost to reach a node ($g(n)$) with the cost to get from the node to the goal ($h(n)$)
  $f(n) = g(n) + h(n)$
- $f(n)$ is the **estimated cost** of the cheapest solution through $n$
- Since we are trying to find the cheapest solution, it makes sense to try the node with the lowest $f(n)$ first
- Under certain conditions of $h(n)$, A* search is complete and optimal

- Certain classes of problems do not care about the path towards a solution or its intermediary states
- We call these methods local search: they consist of optimising an objective function
- Entire classes of algorithms focus on stochastic moves towards local/global maxima
  - (Stochastic) Hill Climbing / Gradient Descent (Ascent)
  - Beam Search
  - Genetic Algorithms

- Games
- Game Theory

- In multiagent environments, each agent must:
  - Consider everyone else's actions
  - Coordinate in order to act coherently
- These issues assumed agents were **cooperative**
- We have also seen that agents can be **competitive**
  - Main subject for **game theory**

- Game theory views any multiagent system as “game“
- Most commonly studied games in AI are called
  - Deterministic
  - Turn-taking (agents act alternately)
  - Two-player
  - Zero-sum (individual utility is always equal and opposite)
  - Perfect Information (fully observable)

- Games in game theory are not the same as the games you play
  - “Diablo 3“ can be modelled as a game
  - Other games include:
    - Naval patrols in pirate-infested seas
    - Police deployment
    - Financial market trading
    - Events-crowd management

- Many games (of the type you play) have been an early focus of AI
- Most real games are **very hard** to solve (e.g. chess)
  - Branching factor of about $35$
  - Matches often go into $50$ rounds ($35^{100}$ nodes)
- Games penalise inefficiency severely (if you wait too long, you lose)

- Key Ingredients:
  - **Players:** who are the decision makers?
    People, governments, companies
  - **Actions:** what can the players do?
    Enter a bid in an auction? Decide whether to end a strike? Decide when to sell a stock? Decide how to vote?
  - **Payoffs:** what motivates players?
    Do they care about some profit? Do the care about other players?

- Two Standard Representations:
  - **Normal Form:** (a.k.a. Matrix Form, Strategic Form)
    List what payoffs get as a function of their actions
    - It is as if players moved simultaneously
    - But strategies encode many things
  - **Extensive Form:** includes timing of moves
    - Players move sequentially, represented as a tree
    - Keeps track of what each player knows when they make each decision

- Finite, n-person **normal form** game $\langle N, A, u \rangle$:
  - **Players:** $N = \{1, \ldots, n\}$ is a finite set of $n$, indexed by $i$
  - **Action Set** for player $i A_i$
    - $a = (a_1, \ldots, a_n) \in A = A_1 \times \cdots \times A_n$ is an **action profile**
  - **Utility function** or Payoff function for player $i : u_i : A \mapsto \mathbb{R}$
    - $u = (u_1, \ldots, u_n)$, is a **profile of utility functions**

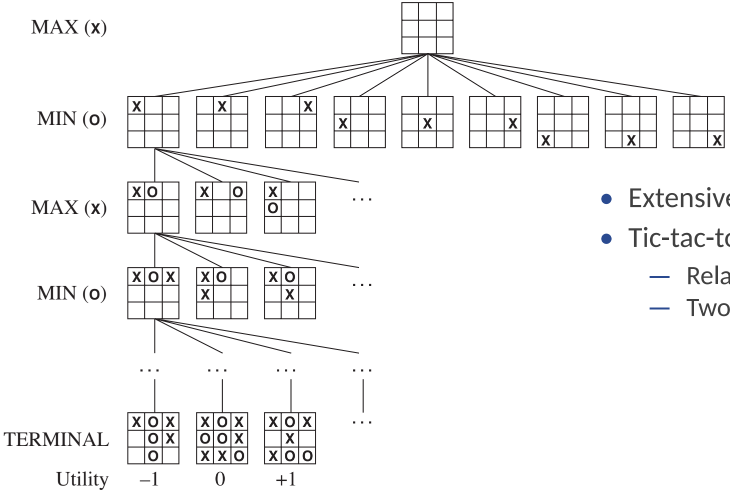
| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | -1, -1 | -3, 0 |
| | D | 0, -3 | -2, -2 |

- Normal Form Game
- Prisoner's Dilemma
  - 2-player game written as a **matrix**
    - row player is player 1, column player is player 2
    - rows correspond to actions $a_1 \in A_1$
    - columns correspond to $a_2 \in A_2$
    - cells listing utility/payoff for each player: row player first, then column

# Game Formalism: Extensive Form

- An extensive form game is defined as a search problem with the following elements:
  - $S_0$: initial state
  - $Player(s)$: which player moves at state $s$
  - $Actions(s)$: what are the actions available at state $s$
  - $Result(s, a)$: **transition model**
  - $Terminal\,Test(s)$: true when the game is over, defines **terminal states**
  - $Utility(s, p)$: utility function (or payoff) for player $p$ at terminal state $s$
    (in zero sum games $Utility(s, p_1) = -Utility(s, p_2)$)

# Example Game




- Extensive form game
- Tic-tac-toe
  - Relatively small tree ($9!$ terminals)
  - Two players (Max and Min)

- Games
- Game Theory

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 4, 4 | 0, 6 |
| | D | 6, 0 | 2, 2 |

- Prisoner's Dilemma, rescaled
  - Double and add six
  - Dominant strategies
  - Equilibria (e.g. Nash)
  - Efficiency/Optimality (e.g. Pareto)

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 4, 4 | 0, 6 |
| | D | ✓6, 0 | 2, 2 |

- Prisoner's Dilemma, rescaled
  - Double and add six
  - Dominant strategies
  - Equilibria (e.g. Nash)
  - Efficiency/Optimality (e.g. Pareto)

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 4, 4 | 0, 6 |
| | D | ✓6, 0 | ✓2, 2 |

- Prisoner's Dilemma, rescaled
  - Double and add six
  - Dominant strategies
  - Equilibria (e.g. Nash)
  - Efficiency/Optimality (e.g. Pareto)

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Player 2</td></tr>
<tr><td>C</td><td>D</td></tr>
<tr><td rowspan="2">Player 1</td><td>C</td><td>4, 4</td><td>0, 6✓</td></tr>
<tr><td>D</td><td>✓6, 0</td><td>✓2, 2</td></tr>
</table>

- Prisoner's Dilemma, rescaled
  - Double and add six
  - Dominant strategies
  - Equilibria (e.g. Nash)
  - Efficiency/Optimality (e.g. Pareto)

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 4, 4 | 0, 6✓ |
| | D | ✓6, 0 | ✓2, 2✓ |

- Prisoner’s Dilemma, rescaled
  - Double and add six
  - Dominant strategies
  - Equilibria (e.g. Nash)
  - Efficiency/Optimality (e.g. Pareto)

| | | Player 2 | |
|---|---|---|---|
| | | Heads | Tails |
| Player 1 | Heads | 1, -1 | -1, 1 |
| | Tails | -1, 1 | 1, -1 |

- Matching Pennies
  - One player wants to match; the other wants to mismatch

| | | Player 2 | |
|---|---|---|---|
| | | Heads | Tails |
| Player 1 | Heads | ✓ 1, -1 | -1, 1 |
| | Tails | -1, 1 | 1, -1 |

- Matching Pennies
  - One player wants to match; the other wants to mismatch

| | | Player 2 | |
|---|---|---|---|
| | | Heads | Tails |
| Player 1 | Heads | ✓ 1, -1 | -1, 1 |
| | Tails | -1, 1 | ✓ 1, -1 |

- Matching Pennies
  - One player wants to match; the other wants to mismatch

| | | Player 2 | |
|---|---|---|---|
| | | Heads | Tails |
| Player 1 | Heads | ✓ 1, -1 | -1, 1 ✓ |
| | Tails | -1, 1 | ✓ 1, -1 |

- Matching Pennies
  - One player wants to match; the other wants to mismatch

| | | Player 2 | |
|---|---|---|---|
| | | Heads | Tails |
| Player 1 | Heads | ✓1, -1 | -1, 1✓ |
| | Tails | -1, 1✓ | ✓1, -1 |

- Matching Pennies
  - One player wants to match; the other wants to mismatch

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1 | 1, -1 |
| | Paper | 1, -1 | 0, 0 | -1, 1 |
| | Scissors | -1, 1 | 1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1 | 1, -1 |
| | Paper | ✓ 1, -1 | 0, 0 | -1, 1 |
| | Scissors | -1, 1 | 1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1 | 1, -1 |
| | Paper | ✓1, -1 | 0, 0 | -1, 1 |
| | Scissors | -1, 1 | ✓1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1 | ✓ 1, -1 |
| | Paper | ✓ 1, -1 | 0, 0 | -1, 1 |
| | Scissors | -1, 1 | ✓ 1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1✓ | ✓1, -1 |
| | Paper | ✓1, -1 | 0, 0 | -1, 1 |
| | Scissors | -1, 1 | ✓1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1✓ | ✓1, -1 |
| | Paper | ✓1, -1 | 0, 0 | -1, 1✓ |
| | Scissors | -1, 1 | ✓1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | | |
|---|---|---|---|---|
| | | Rock | Paper | Scissors |
| Player 1 | Rock | 0, 0 | -1, 1✓ | ✓1, -1 |
| | Paper | ✓1, -1 | 0, 0 | -1, 1✓ |
| | Scissors | -1, 1✓ | ✓1, -1 | 0, 0 |

- Rock, Paper, Scissors

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 0, 0 | -1, 1 |
| | D | 1, -1 | -2, -2 |

- Quitting Game
  - Who Quits First?
  - Anti-coordination game

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 0, 0 | -1, 1 |
| | D | ✓ 1, -1 | -2, -2 |

- Quitting Game
  - Who Quits First?
  - Anti-coordination game

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 0, 0 | ✓ -1, 1 |
| | D | ✓ 1, -1 | -2, -2 |

- Quitting Game
  - Who Quits First?
  - Anti-coordination game

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 0, 0 | ✓ -1, 1✓ |
| | D | ✓ 1, -1 | -2, -2 |

- Quitting Game
  - Who Quits First?
  - Anti-coordination game

| | | Player 2 | |
|---|---|---|---|
| | | C | D |
| Player 1 | C | 0, 0 | ✓ -1, 1 ✓ |
| | D | ✓ 1, -1 ✓ | -2, -2 |

- Quitting Game
  - Who Quits First?
  - Anti-coordination game

| | | Player 2 | |
|---|---|---|---|
| | | Left | Right |
| Player 1 | Left | 1, 1 | 0, 0 |
| | Right | 0, 0 | 1, 1 |

- Coordination Game
  - Which side of the road should you drive on?

| | | Player 2 | |
|---|---|---|---|
| | | Left | Right |
| Player 1 | Left | ✓ 1, 1 | 0, 0 |
| | Right | 0, 0 | 1, 1 |

- Coordination Game
  - Which side of the road should you drive on?

| | | Player 2 | |
|---|---|---|---|
| | | Left | Right |
| Player 1 | Left | ✓1, 1 | 0, 0 |
| | Right | 0, 0 | ✓1, 1 |

- Coordination Game
  - Which side of the road should you drive on?

| | | Player 2 | |
|---|---|---|---|
| | | Left | Right |
| Player 1 | Left | ✓1, 1✓ | 0, 0 |
| | Right | 0, 0 | ✓1, 1 |

- Coordination Game
  - Which side of the road should you drive on?

| | | Player 2 | |
|---|---|---|---|
| | | Left | Right |
| Player 1 | Left | ✓1, 1✓ | 0, 0 |
| | Right | 0, 0 | ✓1, 1✓ |

- Coordination Game
  - Which side of the road should you drive on?

| | | Player 2 | |
|---|---|---|---|
| | | B | F |
| Player 1 | B | 2, 1 | 0, 0 |
| | F | 0, 0 | 1, 2 |

- Matching Game
  - Both players want to **match**

| | | Player 2 | |
|---|---|---|---|
| | | B | F |
| Player 1 | B | ✓ 2, 1 | 0, 0 |
| | F | 0, 0 | 1, 2 |

- Matching Game
  - Both players want to **match**

| | | Player 2 | |
|---|---|---|---|
| | | B | F |
| Player 1 | B | ✓ 2, 1 | 0, 0 |
| | F | 0, 0 | ✓ 1, 2 |

- Matching Game
  - Both players want to **match**

| | | Player 2 | |
|---|---|---|---|
| | | B | F |
| Player 1 | B | ✓2, 1✓ | 0, 0 |
| | F | 0, 0 | ✓1, 2 |

- Matching Game
  - Both players want to **match**

| | | Player 2 | |
|---|---|---|---|
| | | B | F |
| Player 1 | B | ✓2, 1✓ | 0, 0 |
| | F | 0, 0 | ✓1, 2✓ |

- Matching Game
  - Both players want to **match**

To continue in the next session.