ARTIFICIAL INTELLIGENCE

Unit 1 Chapter 3

Topic: Partial Observable Games & State of the art games

VBDS1402

PARTIALLY OBSERVABLE GAMES

- A **partially observable game** is a type of game in which **players do not have complete information** about the current state of the game.
- Some parts of the environment or the opponent's moves are **hidden or uncertain**.
- In other words, players must make decisions **based on limited or incomplete knowledge.**

KEY CHARACTERISTICS

Partial Observability: In POMDPs, the agent's observations are incomplete and do not directly reveal the true state of the environment. This introduces uncertainty, as the agent must reason about the possible states given its observations

Hidden States: The environment's true state, also known as the hidden state, evolves according to a probabilistic process. The agent's observations provide noisy or incomplete information about this hidden state.

Belief State: To handle partial observability, the agent maintains a belief state, which is a probability distribution over possible hidden states. The belief state captures the agent's uncertainty about the true state of the environment.

Action and Observation: The agent takes actions based on its belief state, and it receives observations that depend on the hidden state. These observations help the agent update its belief state and make decisions.

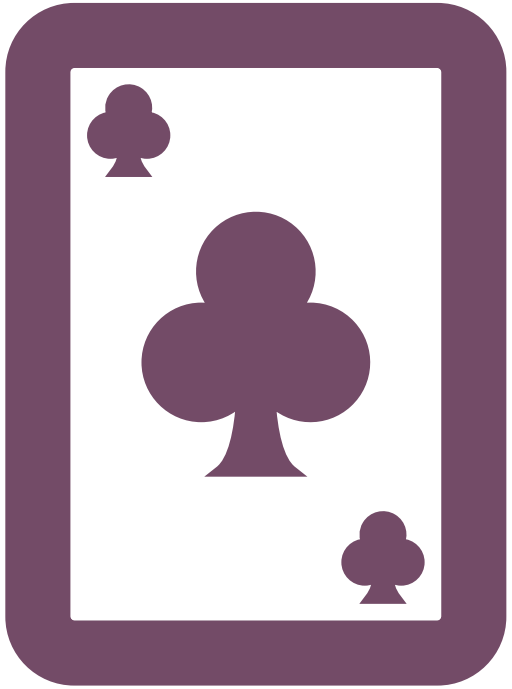
Objective and Policy: The agent's goal is to find a policy—a mapping from belief states to actions—that maximizes a specific objective, such as cumulative rewards or long-term expected utility.

Game	Type of Hidden Information
Poker	Opponent’s cards are hidden
Bridge	Only some cards are visible
Battleships	Opponent’s ship positions are unknown
Blind Tic-Tac-Toe	You play without seeing the opponent’s moves
Scrabble	Opponent’s tiles are hidden

EXAMPLES

POKER AS A PARTIAL OBSERVABLE GAME – A CASE STUDY

- In **Poker**, players have **incomplete information** about the game state — each player knows **their own cards**, but **not their opponents'**.
  - This means the **true state of the game** (all players' hands + community cards + deck) is **hidden** from any single player.
  - Players must **make decisions based on beliefs** about what the opponent might have.
- Thus, Poker is a **multi-agent**, **partially observable**, and **stochastic** game.

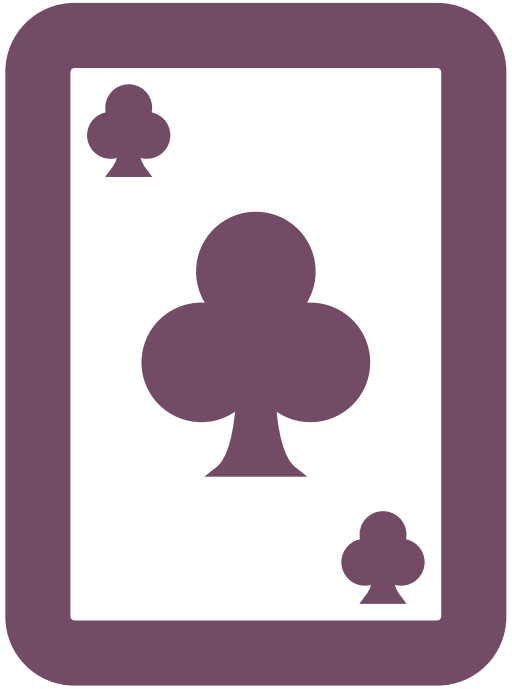


COMPONENTS OF THE GAME MODEL

Component	Description
Players	Usually 2–10 human or AI agents
States (S)	Combination of all players' cards + community cards + betting state
Actions (A)	Fold, Check, Call, Bet, Raise
Observations (O)	Each player observes their private cards + public actions/cards
Transition Function (P)	Determined by random card draws (chance moves) and player actions
Rewards (R)	Gain/loss of money based on pot and hand outcome
Information Sets	Group of states indistinguishable to a player (different opponent cards, same visible info)

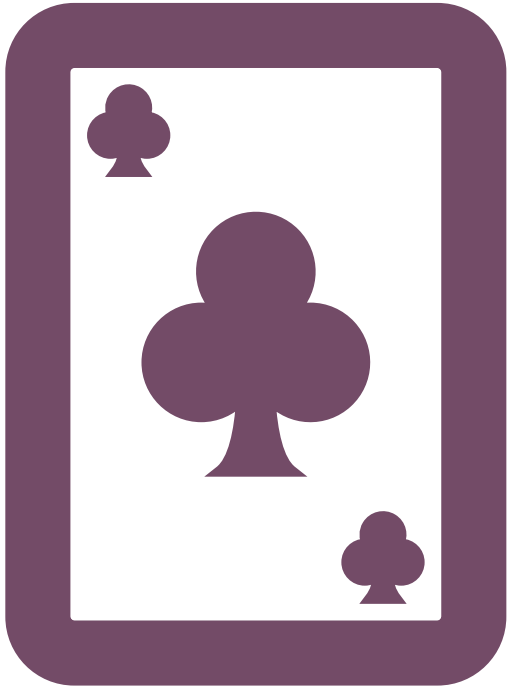
SOURCES OF UNCERTAINTY

- **Hidden Cards:**
  - Each player can see only their own “hole” cards.
  - Opponent’s cards are **unknown** (hidden state).
- **Chance Moves:**
  - The deck is shuffled randomly, so card dealing introduces **stochasticity**.
- **Opponent Strategies:**
  - Opponents can bluff or play unpredictably.
  - Their decisions add strategic uncertainty.



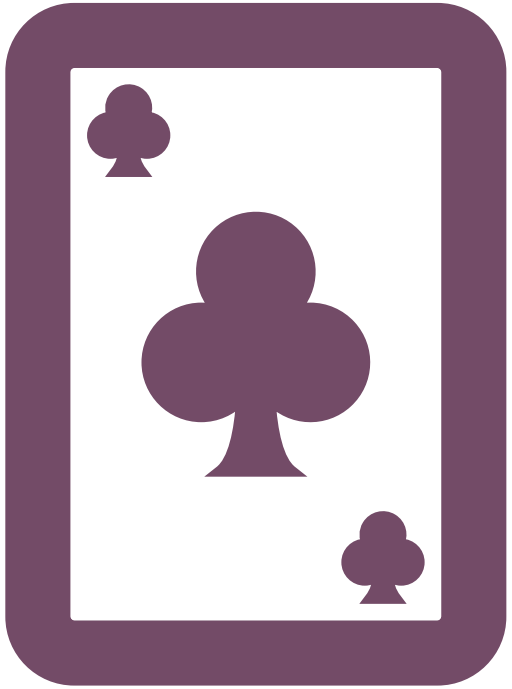
WHAT MAKES POKER HARD FOR AI

- Poker involves **incomplete information** — unlike games like Chess or Go where the full board is visible.
- An optimal strategy depends not only on cards but also on **what you believe about others' cards and behavior**.
- Thus, Poker is a **game of strategy under uncertainty** — players must balance:
  - **Risk**
  - **Deception (bluffing)**
  - **Inference** (guessing opponent's hidden cards)



BELIEF STATE IN POKER

- Since players don't know opponents' cards, they form a **belief** — a **probability distribution** over what cards the opponent might have.
- Example (simplified):
  - Suppose you hold King♠, Queen♠ and the board is 10♠, J♦, 2♣.
  - You might believe there's a **40% chance** your opponent has an Ace (making a straight), and a **60% chance** they have something weaker.
- All your actions — bet, call, or fold — are based on that belief.

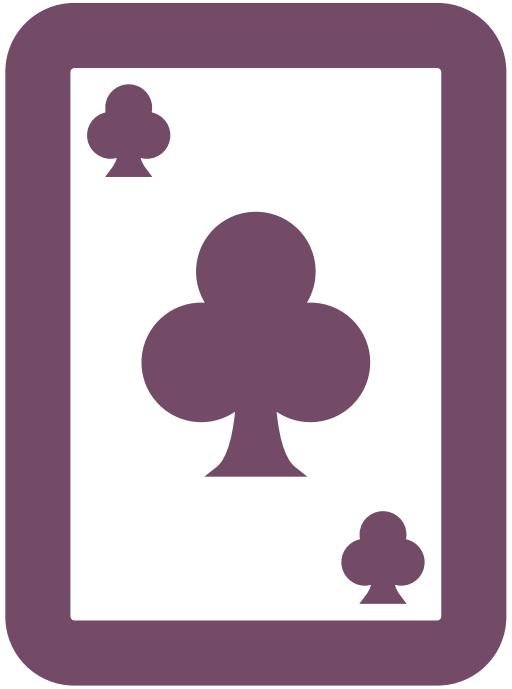


STATE TRANSITIONS IN POKER

Round	Type	Description
Pre-Flop	Chance	Each player gets 2 private (hole) cards
Flop	Chance	3 community cards revealed
Turn	Chance	1 community card revealed
River	Chance	Final community card revealed
Showdown	Deterministic	Hands compared to determine winner

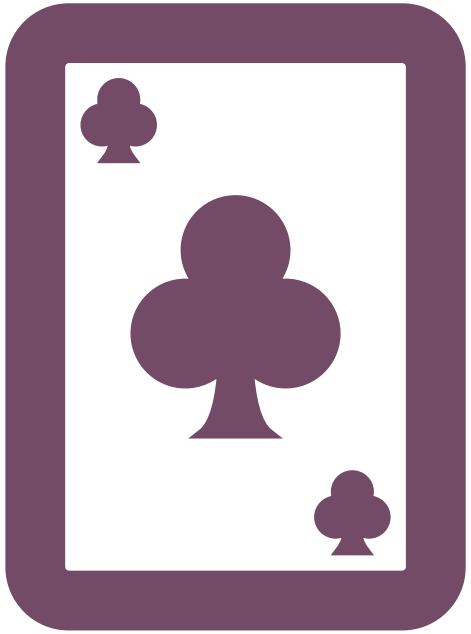
HOW POKER IS MODELED IN GAME THEORY

- In **extensive form**, the game tree has:
  - **Chance nodes** → dealing cards.
  - **Decision nodes** → player actions.
  - **Information sets** → group nodes where a player cannot tell them apart.
  - **Terminal nodes** → showdowns or folded hands (payoffs assigned).
- The **solution** to this type of game is a **Nash equilibrium strategy**, where no player can improve their expected outcome by changing strategy alone.



STRATEGIES IN PARTIAL OBSERVABILITY

- **Belief Updating:**
  Updating probabilities of what cards opponents might hold, based on their actions.
  - e.g., if the opponent raises heavily, you might infer strong cards.
- **Mixed Strategies:**
  Randomizing actions (bet/check/fold) to remain unpredictable.
  Example: Bluffing occasionally even with weak hands.
- **Opponent Modeling:**
  Predicting opponent play patterns to adapt strategy.
- **Expected Value Calculation:**
  Computing expected gain/loss given beliefs and possible outcomes.



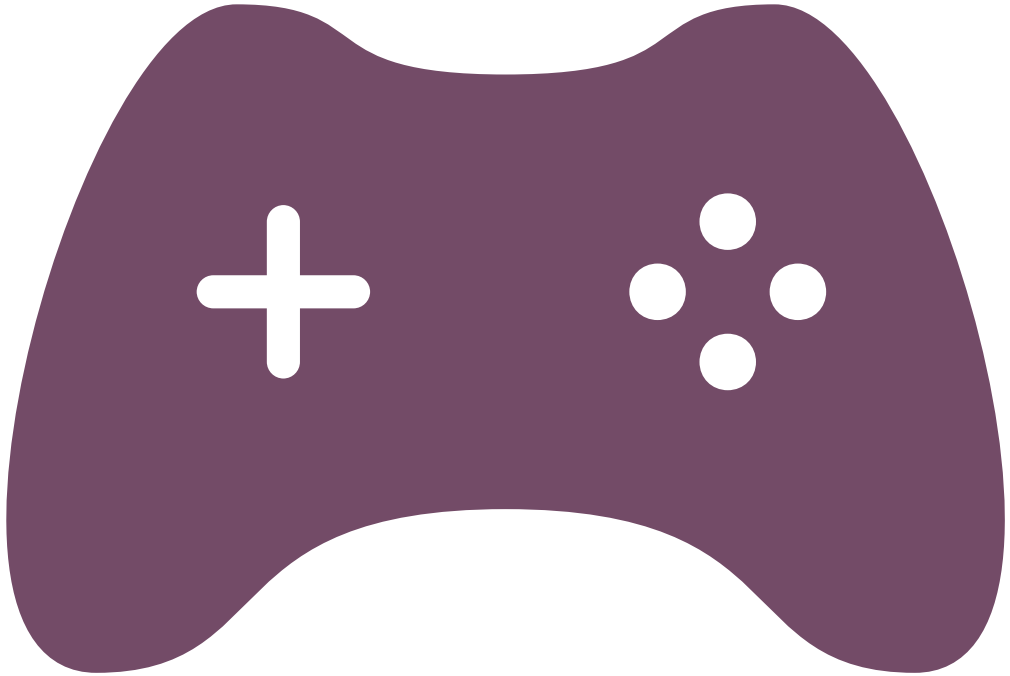
WHY POKER IS AN IDEAL AI TESTBED

- Combines **stochasticity (random cards) + strategic reasoning (bluffing) + partial information (hidden cards)**.
- Demonstrates **real-world decision-making under uncertainty**.
- Methods used in Poker (belief modeling, regret minimization) are now used in:
  - **Cybersecurity (defense strategies)**
  - **Negotiation and auctions**
  - **Robotics and multi-agent RL**

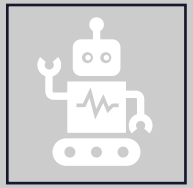
FULLY OBSERVABLE VS. PARTIALLY OBSERVABLE ENVIRONMENT IN AI

Aspects	Fully Observable Environment	Partially Observable Environment
Access	Complete access to the environment's state	Limited or incomplete access to environment's state
Information Availability	All relevant aspects are directly observable	Some aspects may be obscured, uncertain, or missing
Decision-Making	Straightforward decision-making based on complete information	More complex due to incomplete information
Memory Requirement	No or minimal memory requirements.	Needed to track previous observation.
Solution	Optimal and Transparent.	Sub-optimal and unexpected.
Example	Chess, Tic-tac-toe	Poker Game, Autonomous driving, Robot navigation

STATE-OF-THE-ART GAME PROGRAMS



STATE OF THE ART GAME PROGRAMS



State-of-the-art (SOTA) game programs are AI systems that play games at or beyond **human expert level**.

They demonstrate how different AI techniques — from **search algorithms** and **heuristics** to **deep learning** and **reinforcement learning** — can solve complex, strategic environments.

These systems act as **benchmarks** for AI progress, showing how computers can:

- Plan ahead
- Handle uncertainty
- Learn from experience
- And compete with (or outperform) humans

HISTORICAL EVOLUTION OF GAME-PLAYING AI

Era	Milestone	Game	Technique	Achievement
1950s–1980s	**Early AI Programs**	Checkers (Samuel, 1959)	Heuristic search	First learning program
1990s	**Search-based mastery**	Chess (Deep Blue, 1997)	Minimax + Alpha–Beta + Heuristics	Defeated World Champion Garry Kasparov
2000s	**Probabilistic AI**	Backgammon (TD-Gammon, 1992)	Neural nets + Temporal-Difference learning	Matched human experts
2010s	**Deep Learning Revolution**	Go (AlphaGo, 2016)	Deep neural networks + Monte Carlo Tree Search	Beat world champion Lee Sedol
Late 2010s–2020s	**Self-Learning & Multi-Agent AI**	Poker (Libratus, 2017), Dota 2 (OpenAI Five, 2019), StarCraft II (AlphaStar, 2019)	Reinforcement learning, self-play, opponent modeling	Surpassed top human professionals

COMPARISON TABLE OF MAJOR SOTA GAMES

Game	Info Type	AI System	Year	Key Technique	Outcome
Checkers	Perfect	Samuel's Program	1959	Reinforcement + Heuristics	World-champion level
Chess	Perfect	Deep Blue	1997	Minimax + Alpha–Beta	Beat world champion
Backgammon	Stochastic	TD-Gammon	1992	TD Learning + Neural Nets	Expert level
Go	Perfect	AlphaGo / AlphaZero	2016–18	Deep RL + MCTS	Superhuman
Poker	Imperfect	Libratus / Pluribus	2017–19	CFR + RL + Game Theory	Superhuman
Dota 2	Real-time	OpenAI Five	2019	Deep RL + Self-play	Beat professionals
StarCraft II	Real-time	AlphaStar	2019	Multi-agent Deep RL	Beat professionals

KEY AI TECHNIQUES BEHIND SOTA GAMES

Technique	Description	Example
Minimax + Alpha–Beta Pruning	Search optimization for deterministic games	Chess, Checkers
Monte Carlo Tree Search (MCTS)	Random sampling of future plays to evaluate states	Go, AlphaGo
Neural Networks (Policy/Value Nets)	Learn to predict best moves and outcomes	AlphaGo, AlphaZero
Reinforcement Learning (RL)	Learn via trial and error from self-play	TD-Gammon, OpenAI Five
Counterfactual Regret Minimization (CFR)	Compute equilibrium in imperfect-information games	Libratus, Pluribus
Self-Play Learning	Compete against oneself to improve strategy	AlphaZero, OpenAI Five
Population-Based Training (PBT)	Multi-agent evolutionary learning	AlphaStar