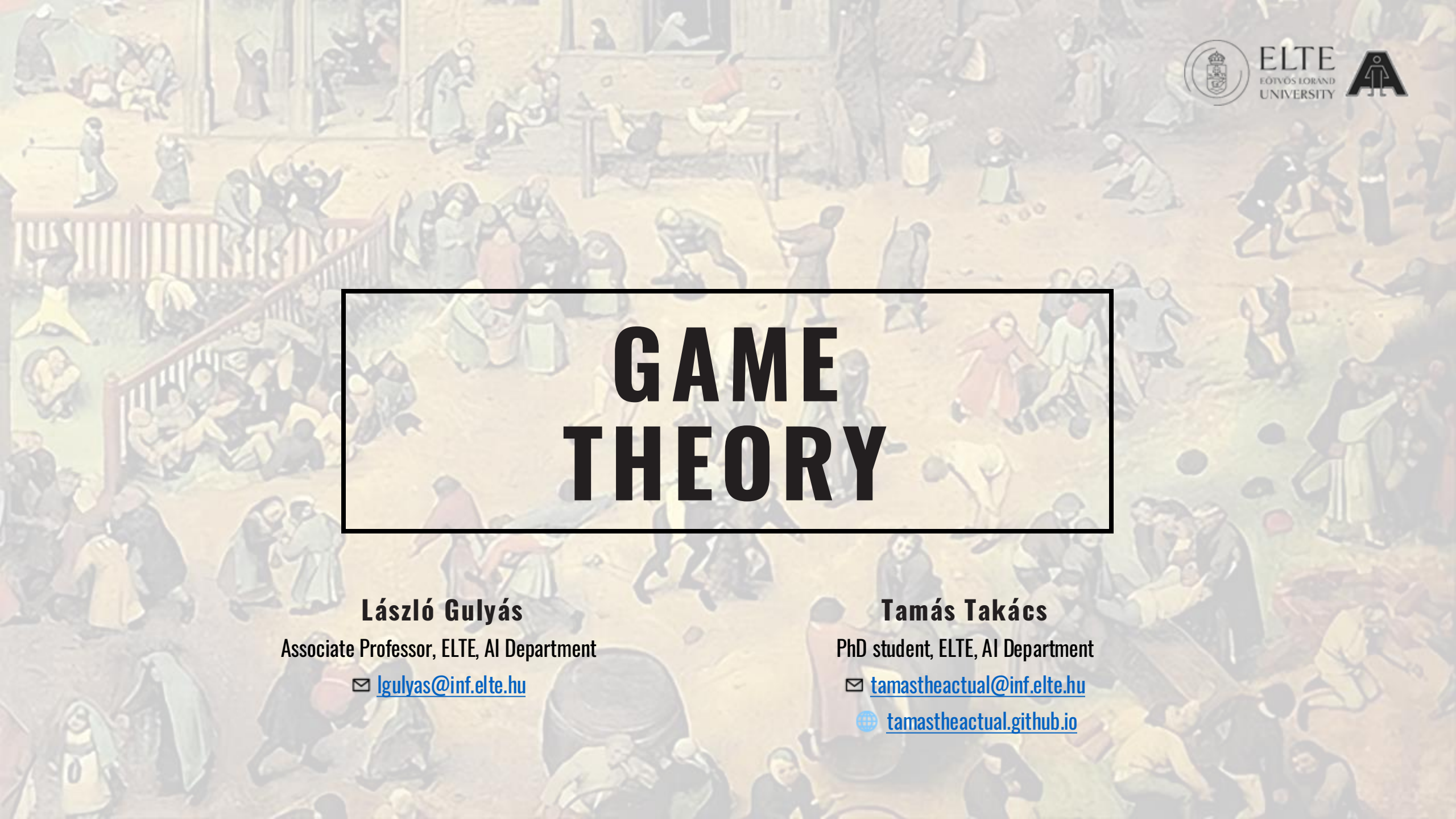


GAME THEORY

László Gulyás

Associate Professor, ELTE, AI Department

lgulyas@inf.elte.hu

Tamás Takács

PhD student, ELTE, AI Department

tamastheactual@inf.elte.hu

tamastheactual.github.io

Game Theory

ELTE EÖTVÖS LORÁND UNIVERSITY

Lecture 6

Stochastic Games and Communication

1 Stochastic Games

2 Alternative SGs

3 Communication

Why Sequential Games?

Stage game: Row payoff matrix $A \in R^{m \times n}$, Column payoff $-A$.
Round $t = 0, 1, 2, \ldots$: players choose pure actions $i_t \in [m], j_t \in [n]$.
Instant payoff: $r_t = A_{i_t j_t}$ to Row (and $-r_t$ to Column).

A **joint policy** $\Pi = (\pi_1, \pi_2)$ specifies, for each round t, a (possibly history-dependent) mixed action for each player:

$$\pi_1(\cdot \mid h_t) \in \Delta^m, \quad \pi_2(\cdot \mid h_t) \in \Delta^n, \qquad h_t = (i_0, j_0, \ldots, i_{t-1}, j_{t-1}).$$

- **Stationary (memoryless) policy:** $\pi_1(\cdot \mid h_t) \equiv p \in \Delta^m, \pi_2(\cdot \mid h_t) \equiv q \in \Delta^n$ for all t.
- Unless stated otherwise, draws are **independent across players and across time** under a stationary policy.

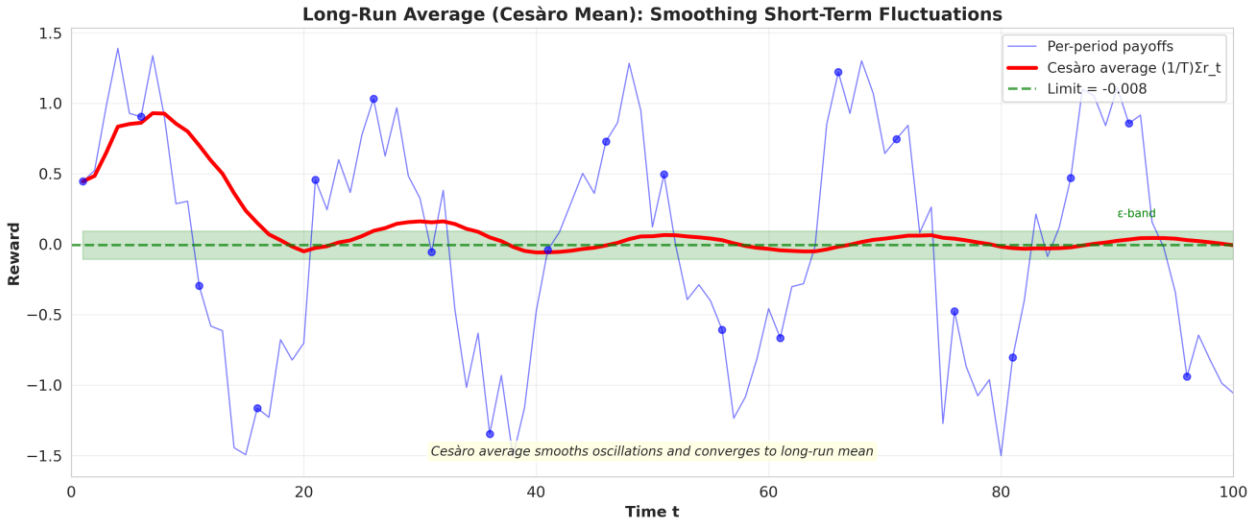
Notation: e_i denotes the i-th standard basis vector.
$\mathbf{1}$ denotes an all-ones vector.
$\Delta^k = \{x \in R^k : x \geq 0,\ \mathbf{1}^\top \mathbf{x} = 1\}$ is the probability simplex.

Long-run average reward (Cesàro average)

Under stationary independent mixing (p, q),

$$\bar{J}(p,q) \coloneqq \lim_{T\to\infty} \frac{1}{T} E\left[\sum_{t=0}^{T-1} r_t\right] = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} E[r_t] = p^\top A q$$

Interpretation: The one-shot value $p^\top A q$ is the **per-period** expected return of the stationary joint policy; discounting just rescales it by $(1-\gamma)^{-1}$.



Minimax Theorem

Theorem (von Neumann). For any finite two-player zero-sum matrix game with Row payoff $A \in R^{m \times n}$,

$$\max_{p \in \Delta^m} \min_{q \in \Delta^n} p^\top A\, q = \min_{q \in \Delta^n} \max_{p \in \Delta^m} p^\top A\, q = v$$

There exist optimal mixes $p^* \in \Delta^m,\ q^* \in \Delta^n$ such that

$$p^{*\top} A q \geq v \ \ \forall q, \qquad p^\top A q^* \leq v \ \ \forall p$$

Row can **guarantee** at least v (security), Column can **hold** Row down to at most v; equality pins down the **value** and optimal mixed strategies.

Example 1 (2×2 by indifference)

$$A = \begin{pmatrix} 2 & -1 \\ -3 & 4 \end{pmatrix}$$

Let $p = \Pr[U]$, $q = \Pr[L]$.

Row indiff. (equalize Column's payoffs):

$$2q + (-3)(1-q) = -1 \cdot q + 4(1-q) \Rightarrow 3q - 1 = -7q + 4 \Rightarrow q^* = 0.5$$

Column indiff. (equalize Row's payoffs):

$$2p + (-1)(1-p) = -3p + 4(1-p) \Rightarrow 3p - 1 = -7p + 4 \Rightarrow p^* = 0.7$$

Value:

$$v = p^{*\top} A q^* = (0.7 \quad 0.3) \begin{pmatrix} 2 & -1 \\ -3 & 4 \end{pmatrix} (0.5 0.5) = 0.05.$$

ε –security

For any $\varepsilon > 0$, there exists p_ε s.t.

$$\min_q p_\varepsilon^\top A\, q \;\geq\; v - \varepsilon,$$

and q_ε s.t.

$$\max_p p^\top A\, q_\varepsilon \;\leq\; v + \varepsilon$$

Practice: When you compute $(\hat{p}, \hat{q})$ numerically, report the **deviation incentives**

$$\varepsilon_{\text{row}} = \max_i (A\hat{q})_i - \widehat{p^\top} A\hat{q}, \qquad \varepsilon_{\text{col}} = \widehat{p^\top} A\hat{q} - \min_j \left(\widehat{p^\top} A\right)_j$$

and use $\max(\varepsilon_{\text{row}}, \varepsilon_{\text{col}})$ as a conservative error bound.

Game Theory

Lecture 6

Stochastic Games and Communication

1 Stochastic Games

2 Alternative SGs

3 Communication

Why Games with States?

Consider these real-world scenarios:

- **Cybersecurity patrol**: Defender monitors servers. Attacker probes vulnerabilities. The system state (compromised/secure) changes based on both players' actions.
- **Drone surveillance**: Two drones track a moving target. What they see depends on where they are (state) and where they move (actions).
- **Stock trading**: Traders compete, but market conditions (bull/bear) change the game's payoffs.

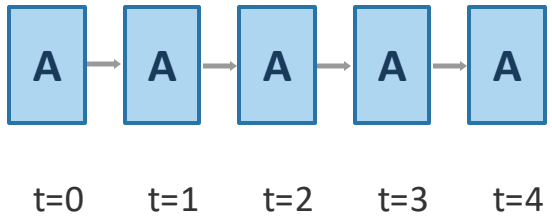
*What's common? The **context** (state) matters, and actions affect future contexts.*

From Repeated Games to Stochastic Games

Repeated Matrix Game	Stochastic Game
Same payoff matrix every round	Payoffs change with state
Independent rounds	State transitions couple rounds
History doesn't affect payoffs	Current state summarizes relevant history
Value = stage game value × discount factor	Value depends on state

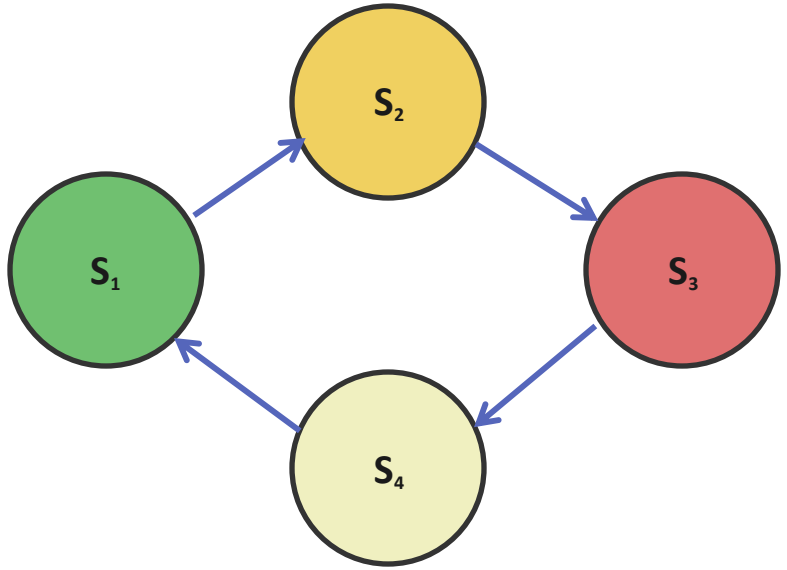
Why Games with States?

Repeated Matrix Game



Same payoff matrix every round

Stochastic Game



Payoffs change with state
Transitions depend on actions

Why Games with States? (Cont.)

- **Negotiation rounds**: In multi-round negotiations, what was offered before (history/state) affects current leverage
- **Traffic routing**: Road congestion state changes as drivers choose routes; tomorrow's congestion depends on today's choices
- **Resource allocation**: Budget state decreases with spending; future opportunities depend on current resource levels

In all cases:

- **Payoffs depend on state**
- **Joint actions determine state transitions**

A Real-World Story: The Patrol Game

Setting: A security guard patrols two locations (North, South). An intruder tries to infiltrate.

States:

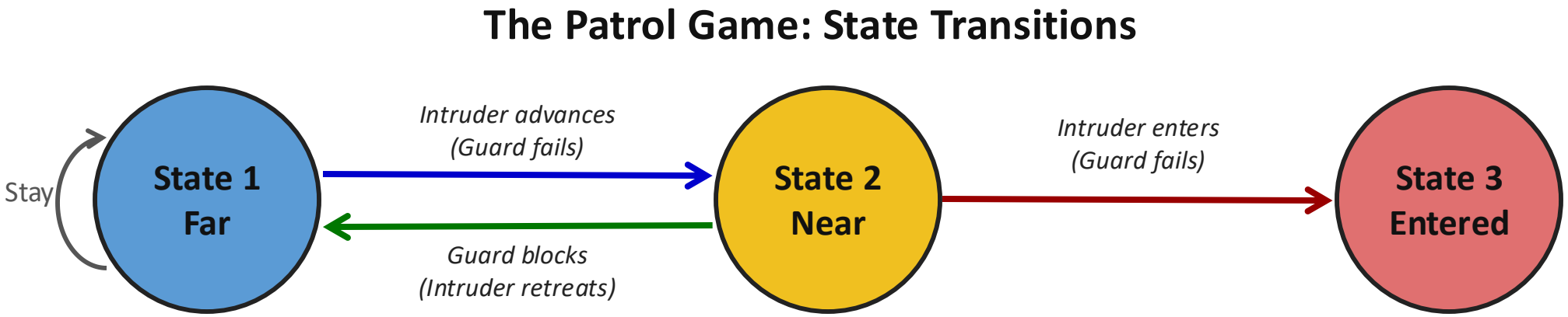
- State 1: "Intruder is far away"
- State 2: "Intruder is near"
- State 3: "Intruder has entered"

Actions: Guard chooses where to patrol; intruder chooses where to attack.

Transitions: If guard isn't at the right place, intruder moves closer (state advances). If guard blocks, state reverts.

Should the guard always patrol the same location? How does the current state affect strategy?

A Real-World Story: The Patrol Game (Cont.)



Stochastic Games: Formal Definition

A finite two-player **stochastic game** is:

$$\mathcal{G} = (S, A, B, P, r, \gamma)$$

State space: $S = \{1,2, \dots, |S|\}$ (finite)
Action sets: Row $A = \{1, \dots, m\}$, Column $B = \{1, \dots, n\}$
Transition kernel: $P(s' \mid s, a, b) =$ probability of next state s' given current state s and joint action (a, b)
Reward function: $r(s, a, b) =$ immediate payoff to row player
Discount factor: $\gamma \in (0,1)$

In zero-sum games, column receives $-r(s, a, b)$.

How Play Proceeds

At each round t:

1. Observe current state s_t
2. Row chooses action $a_t \in A$, Column chooses $b_t \in B$(simultaneously)
3. Players receive rewards: $r(s_t, a_t, b_t)$ to row
4. State transitions: $s_{t+1} \sim P(\cdot \mid s_t, a_t, b_t)$
5. Repeat from new state s_{t+1}

Policies: What's a Strategy Here?

Stationary policy: Choose action based only on current state.

- Row policy: $\pi(a \mid s) =$ probability of action a in state s
- Column policy: $\sigma(b \mid s) =$ probability of action b in state s

History-dependent policy: Could condition on entire history $h_t = (s_0, a_0, b_0, s_1, \ldots, s_t)$

*For discounted zero-sum games, it turns out **stationary policies are sufficient** for optimality!*

Value of a Policy Pair

Discounted value starting from state s:

$$V^{\pi,\sigma}(s) = E[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, b_t) \,|s_0 = s]$$

- Sum of discounted rewards over infinite horizon
- Expectation over random actions and state transitions
- Depends on both players' policies (π, σ)

For initial state distribution μ:

$$J_\gamma(\mu; \pi, \sigma) = \sum_{s \in S} \mu(s), V^{\pi,\sigma}(s)$$

Example: Special Case - Repeated Matrix Game

If there's only **one state** and $P(s' \mid s, a, b) = 1$ (always stay in same state):

$$V^{\pi,\sigma}(s) = \sum_{t=0}^{\infty} \gamma^t E[r(s, a_t, b_t)] = \frac{E[r(s, a, b)]}{1 - \gamma}$$

Stationary mixing in repeated matrix games!

Stochastic games **generalize** this by letting the "matrix game" change with state.

Game Theory

ELTE EÖTVÖS LORÁND UNIVERSITY

Lecture 6

Stochastic Games and Communication

1 Stochastic Games

2 Alternative SGs

3 Communication

Zero-Sum Stochastic Games

In a matrix game, we asked:

$$\max_p \min_q \; p^\top A q = \min_q \max_p \; p^\top A q = v$$

In stochastic games, we ask the same question at each state, but now:

- Immediate reward: $r(s,p,q) = \sum_{a,b} p(a)q(b)r(s,a,b)$
- Future value: depends on where we transition to

The trick: turn this into a recursive equation!

The Shapley Equation: Intuition

At state s, row wants to:

- **Maximize** current reward + discounted future value
- Knowing column will **minimize** it

This gives us the **Shapley operator** T:

$$(TV)(s) = \max_{p \in \Delta(A)} \min_{q \in \Delta(B)} \left\{ r(s,p,q) + \gamma \sum_{s'} P(s' \mid s,p,q) V(s') \right\}$$

where:

- $r(s,p,q)$ = expected immediate reward
- $\sum_{s'} P(s' \mid s,p,q) V(s')$ = expected value of next state
- $V(s')$ = value function for state s' (to be determined)

The Shapley Equation: Fixed Point

Key insight: The true value function V^*must satisfy:

$$V^*(s) = (TV^*)(s) \quad \text{for all states } s$$

This is a **fixed-point equation**!

- If we know $V^*(s')$ for all future states s', we can compute $V^*(s)$
- But all states depend on each other...
- **Solution**: Iterate until convergence!

Why This Works: Contraction Mapping

Theorem (Banach Fixed Point): If T is a **contraction mapping** (shrinks distances), then:

1. T has a **unique fixed-point** V^*
2. Starting from any V_0, iterating $V_{k+1} = TV_k$ **converges** to V^*
3. Convergence is **geometric**: $|V_k - V^*|_\infty \leq \gamma^k |V_0 - V^*|_\infty$

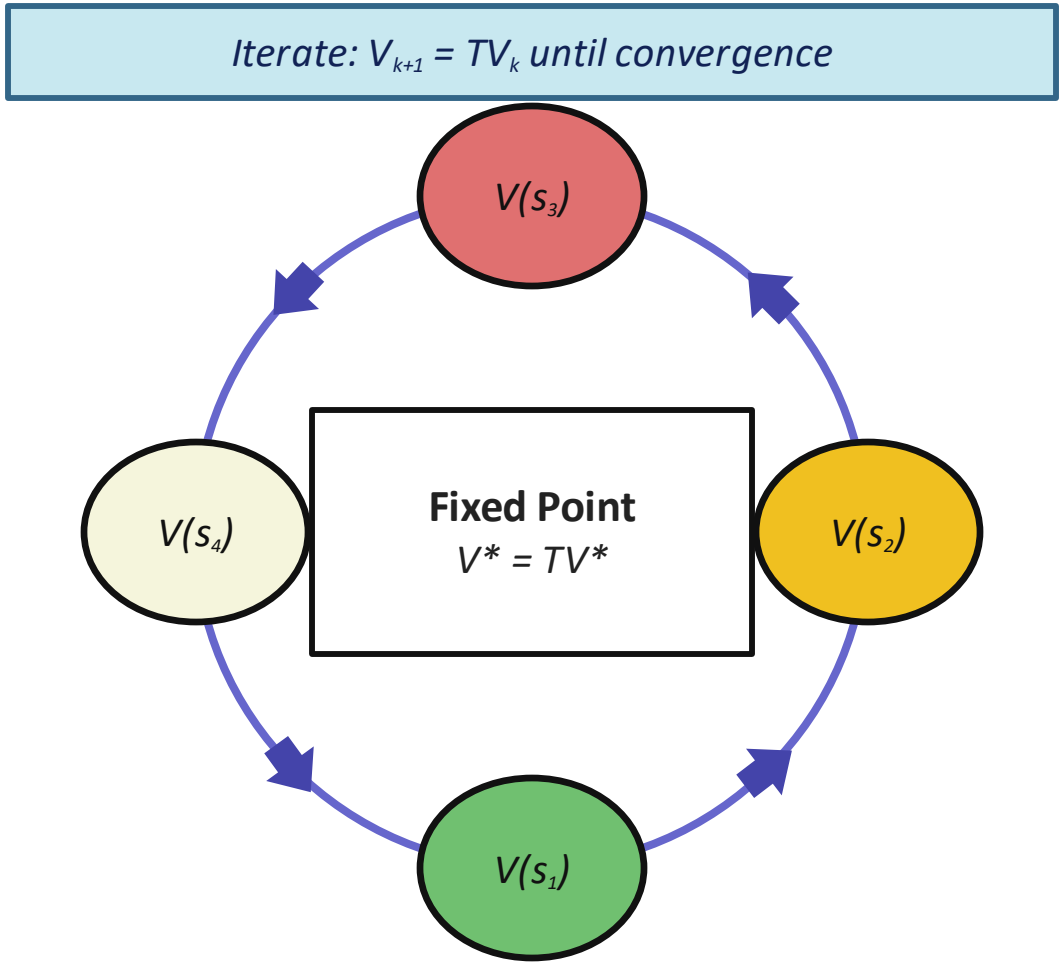
For the Shapley operator:

$$|TV - TW|_\infty \leq \gamma |V - W|_\infty$$

The discount factor $\gamma < 1$ is the contraction rate!

Think of it as: "Future uncertainty shrinks by factor γ each step"

Why This Works: Contraction Mapping (Cont.)



Each state value depends on others through transitions

Value Iteration Algorithm

Initialize: Start with any V_0 (e.g., all zeros)

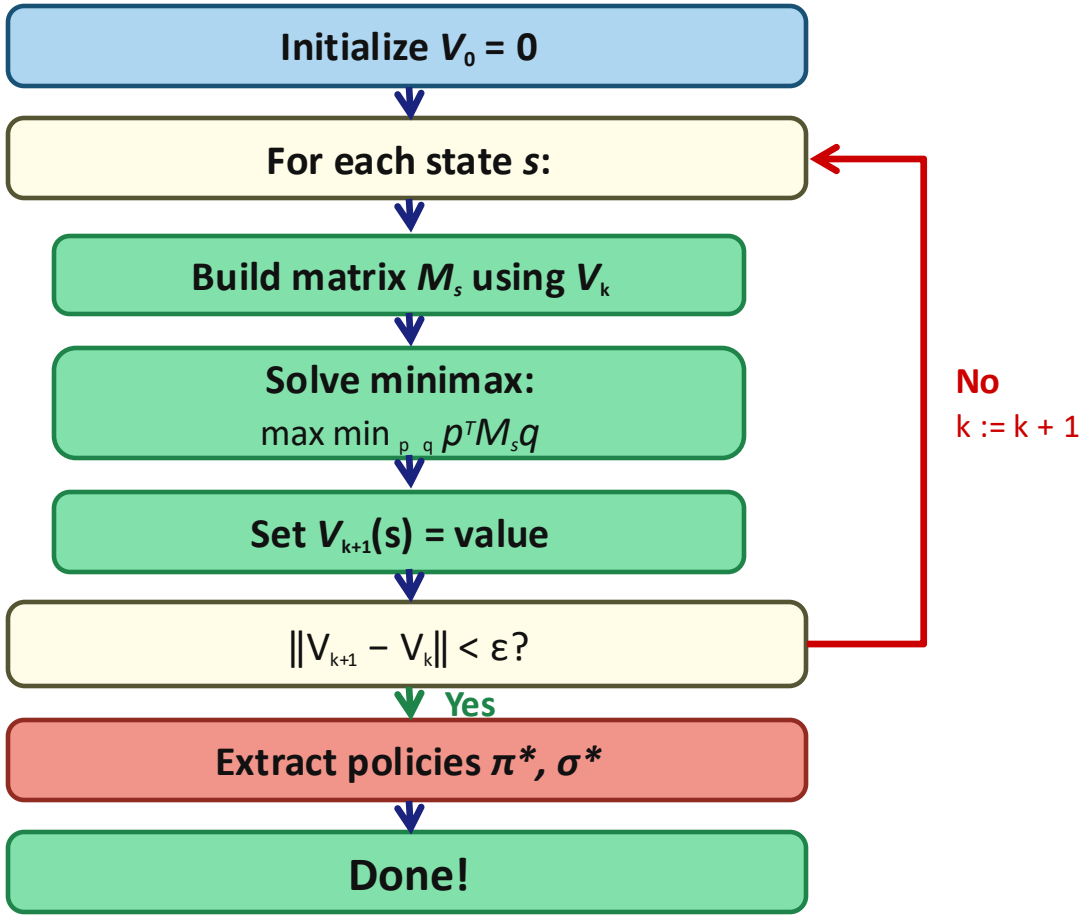
Iterate: For $k = 0, 1, 2, \ldots$

$$V_{k+1}(s) = \max_p \min_q \left\{ r(s,p,q) + \gamma \sum_{s'} P(s' \mid s,p,q) V_k(s') \right\}$$

Stop: When $|V_{k+1} - V_k|_\infty < \epsilon$ (convergence tolerance)

Extract policy: The (p,q) that achieve the max-min at each state form **stationary minimax policies**

Why This Works: Contraction Mapping (Cont.)



Value Iteration: What's Actually Happening?

At iteration k, for each state s:

1. **Build a matrix game** with payoff:

$$M_s(a,b) = r(s,a,b) + \gamma \sum_{s'} P(s' \mid s,a,b) V_k(s')$$

2. **Solve the matrix game:** Find minimax value and optimal mixes (p_s, q_s)
3. **Update** $V_{k+1}(s)$ to this minimax value
4. **Repeat** for all states

Each state's game depends on other states' values from the previous iteration!

Example 1: Two-State Security Game

States: $S = \{1, 2\}$ (System Secure, System Vulnerable)
Actions: Defender $A = \{\text{Monitor}, \text{Patch}\}$, Attacker $B = \{\text{Probe}, \text{Wait}\}$
Discount: $\gamma = 0.8$

Rewards (to defender):

$$r(1) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad r(2) = \begin{pmatrix} -5 & -3 \\ -2 & -1 \end{pmatrix}$$

Transitions:

- State 1: If attacker probes and defender doesn't monitor → state 2 $(prob\ 0.7)$, else stay
- State 2: If defender patches → state 1 $(prob\ 0.6)$, else stay

State 1 is relatively safe, state 2 is dangerous!

Example 1: Iteration 0

Start: $V_0 = (0,0)$

State 1: Build $M_1 = r(1) + 0.8 \cdot (\text{transition} \times V_0)$

- (M, P): reward 0, stay
- reward 1, stay
- reward -1, go to state
- reward 0, stay $\rightarrow M_1(P, W) = 0$

$$M_1 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Solve: mix uniformly, $V_1(1) = 0$

Example 1: Iteration 0 (Cont.)

State 2: Build $M_2 = r(2) + 0.8 \cdot (\text{transition} \times V_0)$

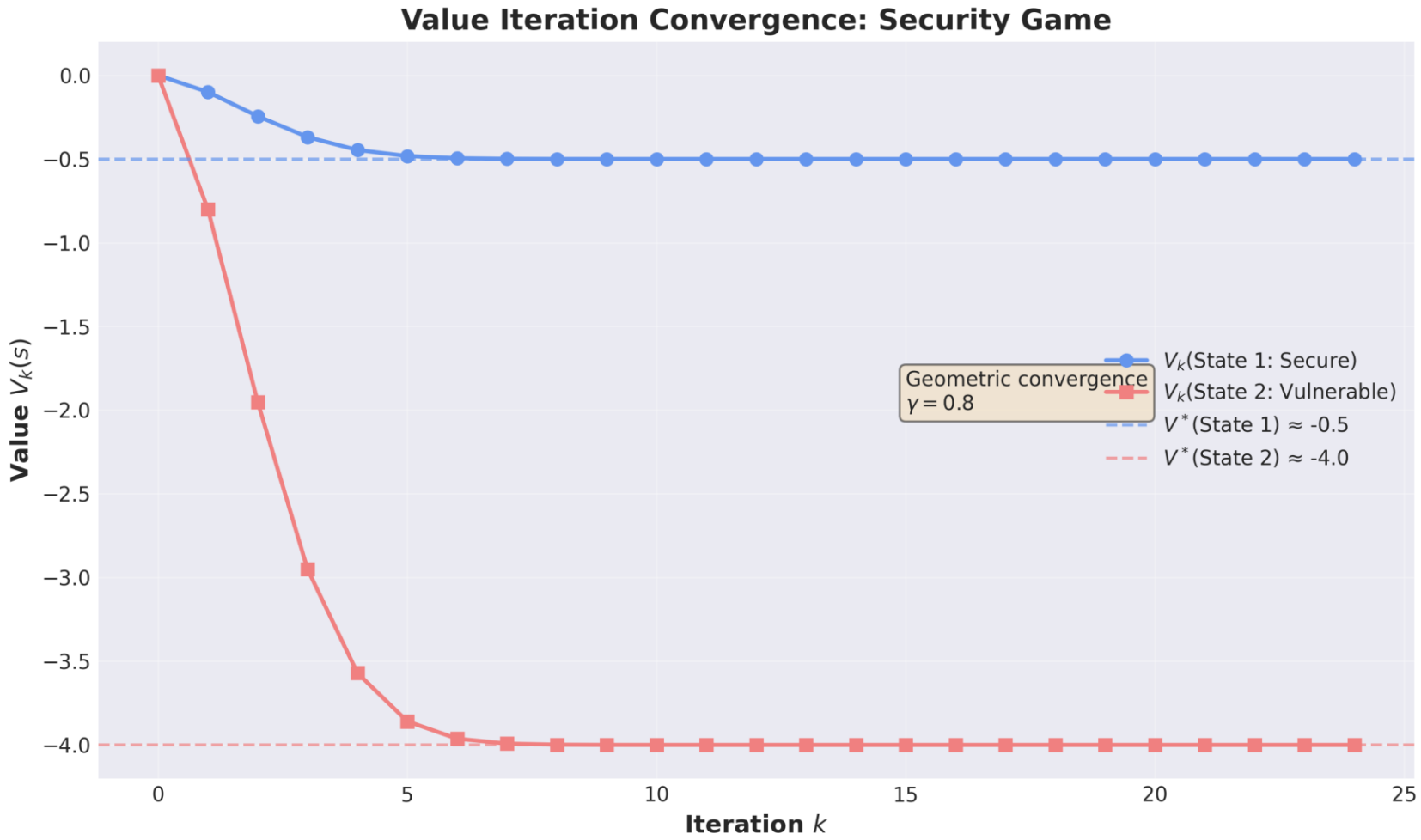
$$M_2 = \begin{pmatrix} -5 & -3 \\ -2 & -1 \end{pmatrix}$$

Solve: Row minimizes loss by playing Patch more often, $V_1(2) \approx -2.5$

After one iteration: $V_1 = (0, -2.5)$

State 2 is worse! This will affect state 1 strategy in next iteration.

Example 1: Convergence



Example 1: Interpretation

After several iterations:

- $V^*(1) \approx -0.5$ - even "secure" state has negative value (defender is losing overall)
- $V^*(2) \approx -4.0$ - vulnerable state is much worse
- **Defender strategy at state 1:** Mix Monitor/Patch to balance staying secure vs preventing attacks
- **Defender strategy at state 2:** Heavily favor Patch to escape back to state 1

Key insight: State values couple through transitions, creating strategic interdependence!

Q-Functions: An Alternative View

Define the **state-action value** (Q-function):

$$Q_V(s,a,b) = r(s,a,b) + \gamma \sum_{s'} P(\, s' \mid s,a,b\,) V(s')$$

Then the Shapley operator becomes:

$$(TV)(s) = \max_p \min_q \sum_{a,b} p(a) q(b) Q_V(s,a,b)$$

Interpretation: Q-values capture "how good is action pair (a,b) at state s?"

Familiar from reinforcement learning? Q-learning extends this to unknown games!

Extracting Policies

Once V_k converges to V^*:

For each state s:

1. Build final matrix game M_s using V^*
2. Solve for minimax strategies (p_s^*, q_s^*)
3. These form **stationary minimax policies**:

Properties:

- These policies are **optimal** for both players
- They achieve the **value** $V^*(s)$ at each state
- No player can improve by deviating (Nash equilibrium!)

Computational Considerations

Solving each matrix game:

- For 2×2 games: Use indifference principle
- For larger games: Linear programming
- For very large games: Approximate methods (beyond scope)

Convergence speed:

- Depends on γ: smaller $\gamma \rightarrow$ faster convergence
- Typical: $10 - 100$ iterations for moderate-sized games
- Each iteration: $O(|S|)$ matrix game solves

Beyond Zero-Sum

In **general-sum** stochastic games:

- Each player i has their own reward function $r_i(s, a, b)$
- Not necessarily $r_2 = -r_1$
- Seek **Nash equilibrium** instead of minimax

Key result: Stationary Nash equilibria **exist** under mild conditions (Fink, 1964)

Challenge: Computing them is much harder!

- No single value function
- Must find mutual best responses at each state
- Multiple equilibria possible

Bellman Equations for General-Sum

For each player i, given opponent policies π_{-i}:

$$V_i^{\pi}(s) = \sum_a \pi(a \mid s) \left[r_i(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V_i^{\pi}(s') \right]$$

Nash equilibrium: Policies π^* such that no player can improve:

$$V_i^{\pi^*}(s) \geq V_i^{(\pi_i', \pi_{-i}^*)}(s) \quad \forall i, s, \pi_i'$$

Solution methods:

- Policy iteration (Pollatschek & Avi-Itzhak, 1969)
- Linear complementarity problems
- Approximate/learning methods

Example 2: Coordination Game with States

States: $S = \{1, 2\}$ (Market A, Market B)
Actions: Each firm chooses $\{\text{Invest}, \text{Wait}\}$

Payoffs:
- State 1: Coordination game favoring (Invest, Invest) → $(3,3)$
- State 2: Coordination game favoring (Wait, Wait) → $(2,2)$

Transitions: Joint investment in state 1 → state 2 (new market opens)

Equilibria:
- Always coordinate on high-investment in state 1
- Always coordinate on low-investment in both states
- Mixed strategies possible

Multiple equilibria reflect coordination challenges across time!

Example 2: Coordination Game with States

States: $S = \{1, 2\}$ (Market A, Market B)
Actions: Each firm chooses $\{\text{Invest}, \text{Wait}\}$

Payoffs:
- State 1: Coordination game favoring (Invest, Invest) → $(3,3)$
- State 2: Coordination game favoring (Wait, Wait) → $(2,2)$

Transitions: Joint investment in state 1 → state 2 (new market opens)

Equilibria:
- Always coordinate on high-investment in state 1
- Always coordinate on low-investment in both states
- Mixed strategies possible

Multiple equilibria reflect coordination challenges across time!

Game Theory

ELTE EÖTVÖS LORÁND UNIVERSITY

Lecture 6

Stochastic Games and Communication

1 Stochastic Games

2 Alternative SGs

3 Communication

Why Partial Observability?

In many real-world scenarios, players don't see the full state:

Poker: You see your cards, not opponent's
Drone teams: Each drone has limited sensor range
Cybersecurity: Defender doesn't know attacker's capabilities
Negotiation: Each party has private information

Partially Observable Stochastic Game (POSG): Each player receives a **private observation** instead of seeing the true state.

POSG Definition

A two-player POSG adds:

Observation sets: Ω_1, Ω_2 for each player
Observation kernels: $O_i(\, o_i \mid s, a, b, s' \,) =$ probability player i observes o_i after transition

Play:

1. True state is s_t (hidden)
2. Players choose actions based on their **observation histories** $h_t^i = (o_{i,1}, a_{i,0}, o_{i,2}, a_{i,1}, \ldots)$
3. Receive private observations $o_{i,t+1} \sim O_i(\,\cdot \mid s_t, a_t, b_t, s_{t+1}\,)$

Challenge: Players must **infer** the state and **coordinate** without shared information!

Belief States

Each player maintains a **belief** over hidden states:

$$b_i(s) = \Pr(\, s_t = s \mid h_t^i \,)$$

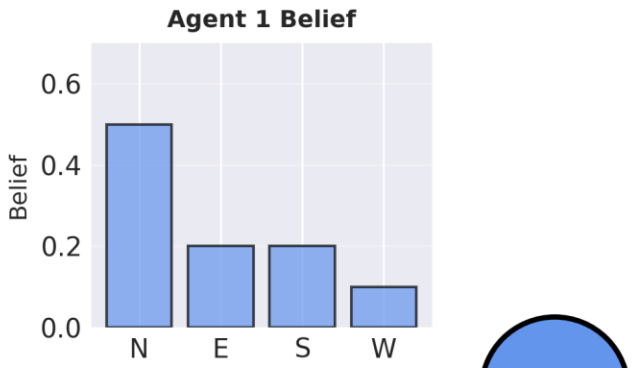
Belief update (Bayesian):

$$b_{i,t+1}(s') \propto \sum_s P(\, s' \mid s, a_t, b_t \,)\, O_i(\, o_{i,t+1} \mid s, a_t, b_t, s' \,) b_{i,t}(s)$$

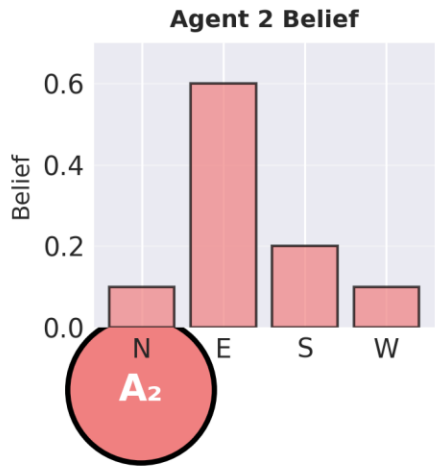
Problem: Each player's belief depends on their own observations only → **asymmetric information**

Consequence: Optimal policies can be very complex (history-dependent)

Belief States



Hidden State: Target at NORTH



Agent 1 (Drone)

Agent 2 (Drone)

"Same world, different beliefs"

Example 3: Cooperative Drone Surveillance

Scenario: Two drones track a target moving between locations.

States: Target is at location $\{1, 2, 3, 4\}$ (grid)

Observations: Each drone sees "near" or "far" based on its position and target position (noisy)

Actions: Move $North/South/East/West$

Reward: $+10$ if both drones reach target, -1 per timestep

Without communication: Each drone only knows its own observations → hard to coordinate

With communication: Can share observations → centralized planning possible

The Role of Communication

Cheap talk: Costless, non-binding messages between players

- Sender has private information (type θ)
- Sends message m to receiver
- Receiver chooses action a

Equilibrium: Message strategy + action strategy that are mutually optimal

Even "cheap" talk can be informative if incentives are aligned!

Sender-Receiver Game

Setup:

- Sender observes type $\theta \in \{\theta_1, \theta_2\}$ with prior π
- Sends message $m \in \{m_1, m_2\}$
- Receiver updates belief $\mu = \Pr(\theta_1 \mid m)$ and chooses a $\in \{a_1, a_2\}$

Equilibria:

Pooling: Sender always sends same message (uninformative)
Separating: Each type sends different message (fully revealing)
Partial pooling: Some types pool, others separate

Incentive constraints: Sender must prefer truth-telling given receiver's response

Example 4: Cheap Talk vs Costly Signaling

Cheap talk (costless messages):

- Worker can claim "I'm great" or "I'm okay" at no cost
- If lying is free → pooling equilibrium (uninformative)
- Only informative when incentives are naturally aligned

Costly signaling:

- Education acts as costly signal (higher cost for low ability)
- Cost differential enables separating equilibrium
- See Spence's job market model for full treatment

Costly signals can be credible even with misaligned incentives

Communication in Stochastic Games

Public signals: A shared random variable observed by both players

- Can coordinate actions through **correlated equilibrium**
- Example: Traffic light as public signal for drivers

Private messages: Each player sends/receives private information

- Can improve coordination in POSGs
- Can enable information sharing about hidden state

Correlated equilibrium: A distribution over joint actions that players want to follow given the public signal

Correlated Equilibrium Intuition

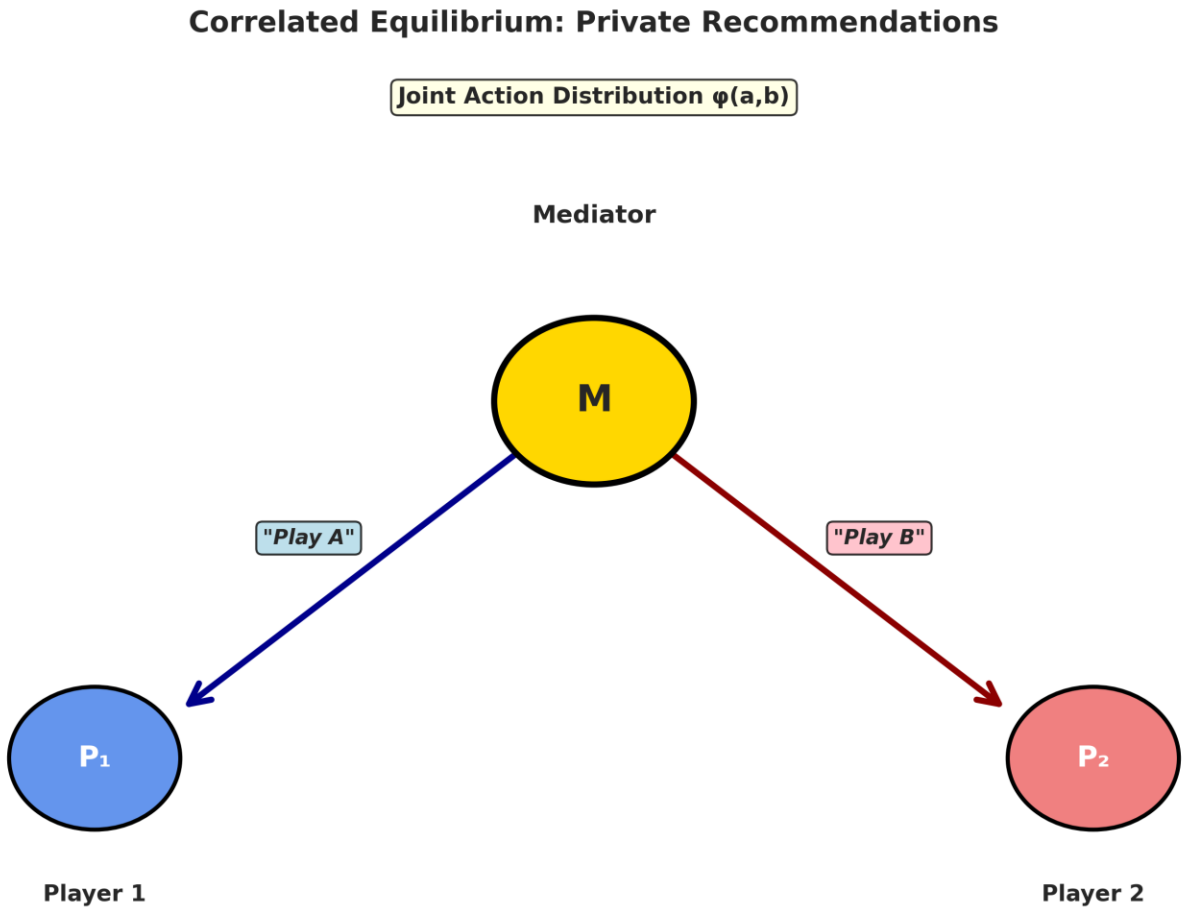
Preview (covered in later lectures): A mediator recommends actions; players obey if it's a best response.

In stochastic games: Mediator can recommend state-dependent action profiles

- At each state s, draw joint action from $\phi_s(a, b)$
- Send private recommendations
- Players obey if no profitable deviation exists

Benefit: Can achieve payoffs beyond Nash equilibrium (better coordination)

Correlated Equilibrium Intuition (Cont.)



Example 5: Correlated Patrol Strategy

Recall patrol game: Guard and intruder in states 1, 2, 3

Mediator device:

- At state 1: Recommend (N, S) or (S, N) with equal probability (guard and intruder separate)
- At state 2: Recommend mixed strategy to guard

Benefit: Coordination avoids worst outcomes, improves average payoff

Implementability: Players must prefer to obey recommendations

When to Use Stochastic Games?

Use stochastic games when:

- Context/state significantly affects payoffs
- Actions influence future states (not just immediate rewards)
- You need to model dynamic strategic interactions
- Long-term thinking matters (discounting)

Examples:

- Cybersecurity (system states: secure/compromised)
- Resource management (budget/inventory states)
- Multi-round negotiations (reputation states)
- Robotics (position/environment states)

Questions

1. How does the discount factor γ affect optimal strategies?
2. When would history-dependent policies be necessary despite stationarity results?
3. How does partial observability change the nature of equilibrium?
4. Can you think of a real-world stochastic game? What are its states and transitions?
5. When would communication be most valuable in a strategic setting?

Practical Exercise

Design your own two-state stochastic game:

1. Choose a real-world scenario (cybersecurity, traffic, negotiation, etc.)
2. Define two states and their interpretations
3. Specify actions for both players
4. Design rewards that reflect the scenario
5. Define transition probabilities
6. Set $\gamma = 0.9$ and run $1 - 2$ iterations of value iteration by hand
7. Interpret the resulting optimal strategies

Summary

- Stochastic games: states, transitions, policies
- Shapley equation and value iteration for zero-sum games
- General-sum equilibria
- Partial observability and the role of communication

Course Textbooks

- Bonanno, G. (2024). ***Game Theory (3rd ed.)***. University of California, Davis. Received from: GT Book
- Axelrod, R. (1984). ***The Evolution of Cooperation***. Basic Books. Received from: Axelrod Article
- Nisan, N., Roughgarden, T., Tardos, É., & Vazirani, V. V. (2007). ***Algorithmic Game Theory***. Cambridge University Press. Received from: AGT Book
- Myerson, R. B. (1991). ***Game Theory: Analysis of Conflict.*** Harvard University Press. Received from: GT Book 2F.
- Christianos et al., ***Multi-Agent Reinforcement Learning: Foundations and Modern Approaches***, 2023. Received from: MARL Book.pdf
- Shoham, Y., & Leyton-Brown, K. (2008). ***Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations***. Cambridge University Press. Received from: MARL Book.pdf
- `***nashpy***`documentation (readthedocs). Link: NashPy Docs

That’s All for Today!