ReinforceGT

Game-Theoretic Extensions to Reinforcement Learning Algorithms.

Based On

Game Theoretic Modelling in Multi-Agent Reinforcement Learning Environments.

Mentors:

- Aayushman Kumar (230029)
- Jeevesh Narayan (230508)
- Hardik Tiwari (230437)
- Ikrima Badr Shamim Ahmed (230482)

INTRODUCTION TO GAME THEORY.

- Game theory analyzes strategic interactions where agents’ outcomes depend on others' actions, using concepts like players, strategies, payoffs, and Nash Equilibrium.
- It models decision-making in competitive and cooperative environments, helping predict rational agent behavior.
- In ReinforceGT, game theory forms the foundation for multi-agent simulations. By integrating it with reinforcement learning, agents dynamically learn optimal strategies through repeated interactions, adapting to changing environments.

INTRODUCTION TO DECENTRAlIZED MARL

- In the deep learning ecosystem, Multi-Agent Reinforcement Learning(MARL) is the area that focuses on the implementation of autonomous, self-learning systems with multiple agents. Conceptually, multi-Agent Reinforcement Learning(MARL) is the deep learning discipline that focuses on models that include multiple agents that learn by dynamically interacting with their environment.
- While in single-agent reinforcement learning scenarios the state of the environment changes solely as a result of the actions of an agent, in MARL scenarios the environment is subjected to the actions of all agents.
- Hence, the complexity of MARL scenarios increases with the number of agents in the environment.

MARL ARCHITECTURE

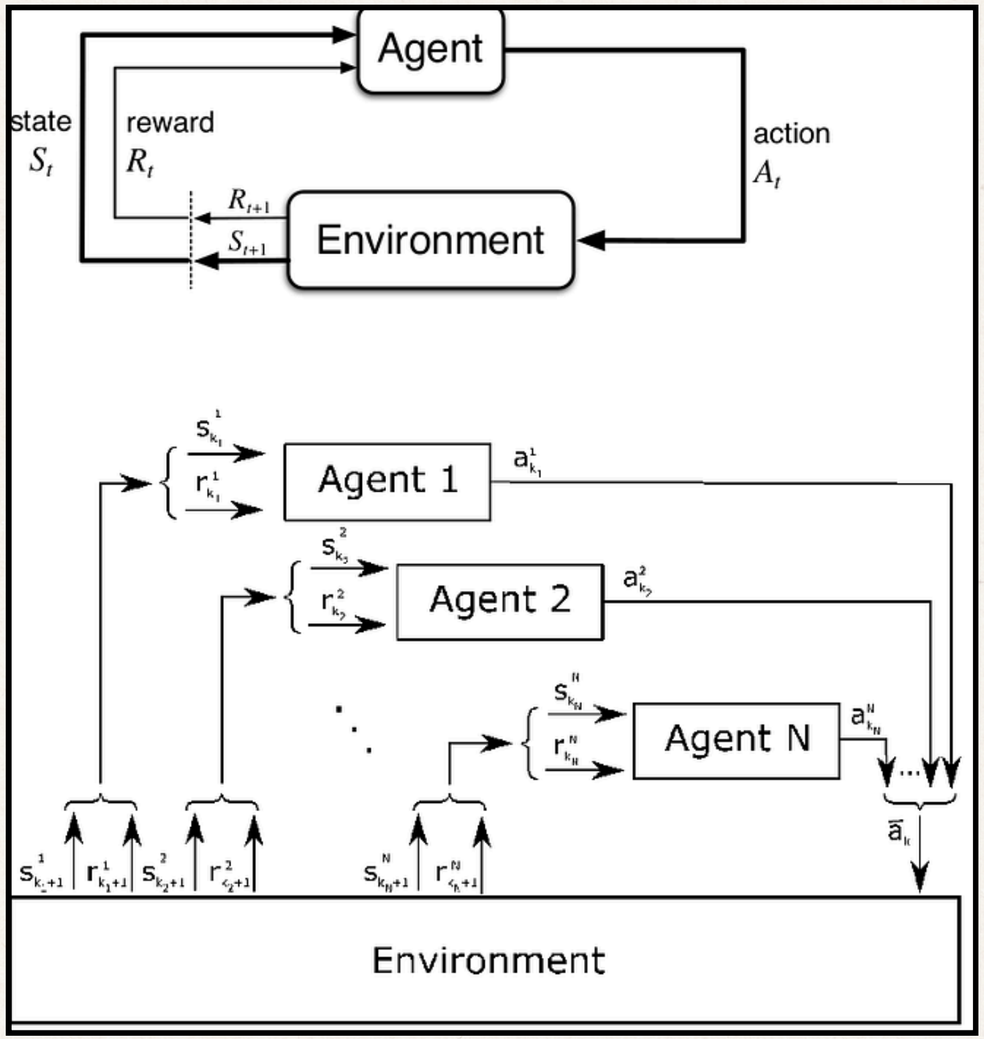

Image Credit : SecondMind

- **Decentralized Agents**: Each agent acts based on its own local observations.
- **Shared Environment**: All actions influence the global state of the environment.
- **Non-Stationarity**: From any single agent's perspective, the environment appears non-stationary because other agents are also learning and adapting.
- **Coordination/Competition**: Depending on the task, agents may be cooperative (e.g., teamwork), competitive (e.g., adversaries), or mixed.

APPLICATIONS OF DECENTRAlIZED MARL

Multi-Agent Reinforcement Learning (MARL) has rich applications in game theory, especially where multiple decision-makers interact strategically. List of key game-theoretic applications of MARL are:

1. Competitive and Adversarial Games

- Agents learn adversarial strategies in zero-sum or auction-based environments, often converging to Nash equilibria through self-play.

2. Cooperative Games

- Agents coordinate to maximize shared rewards, learning decentralized policies in tasks like traffic control or multi-agent planning.

3. Mixed-Motive Games

- Agents navigate trade-offs between individual and collective goals, modeling trust, negotiation, and social dilemmas in strategic settings.

LEARNING OBJECTIVES.

- Grasp core concepts of game theory and strategic decision-making.
- Implement classical games and simulations using Python.
- Understand and apply reinforcement learning in multi-agent environments.
- Design adaptive agents that learn optimal strategies through interaction.
- Develop collaborative coding skills using Git, GitHub, and LaTeX documentation.

PROJECT LOGISTICS.

- This project will run for **8 weeks.**
- We are going to recruit around **~30 mentees.**
- **Time Commitment: 6–8 hours/week** (flexible based on depth of engagement)
- Includes mentor sessions, coding tasks / Assignments, conceptual readings, and optional report writing.
- All materials, references, and starter code maintained on a central GitHub repository

PROJECT TIMELINE

Week 01: Foundations

- Introduction to Game Theory**:** strategies, payoffs, Nash equilibrium
- Basics of agent-based modeling and simulations
- Python refresher (functions, classes, loops) – optional session

Week 2: Classical Games Implementation

- Code and simulate basic strategic games (Prisoner's Dilemma, Stag Hunt, Matching Pennies)
- Analyze outcomes using payoff matrices and visualization
- Discuss concepts of dominance, rationality, and equilibria

Week 3: Auction Models & Strategic Behavior

- Implement English, Dutch, and Sealed-Bid auctions
- Explore bidding strategies under different rules
- Introduce incomplete information and utility optimization

Week 4: Introduction to Reinforcement Learning

- Concepts: rewards, policies, value functions, exploration vs. exploitation
- Implement basic RL algorithms (Q-Learning or SARSA) in single-agent settings
- Visualize learning curves and convergence

PROJECT TIMELINE

Week 5: Multi-Agent Reinforcement Learning (MARL)

- Extend RL to multi-agent settings: shared vs. individual rewards
- Challenges in MARL: non-stationarity, credit assignment
- Implement a MARL environment (e.g., competitive or cooperative game)

Week 6: Strategic Simulation with Learning Agents

- Combine game-theoretic reasoning with RL (e.g., agents learning in social dilemmas or auctions)
- Analyze emergent behaviors and equilibrium dynamics
- Optional: Experiment with policy gradient methods

Week 7–8: Final Extensions & Reporting (Optional/Dynamic)

- Explore additional strategic models (negotiation, coordination, mixed-motive games)
- Work on final projects or agent experiments
- Prepare optional LaTeX-based reports and submit via GitHub PRs

ASSIGNMENTS

Week 1: Game Theory Fundamentals

- Implement payoff matrix calculators for 2-player strategic games (Prisoner's Dilemma, Stag Hunt, Battle of the Sexes).
- Write a short Python function to detect dominant strategies and pure-strategy Nash equilibria.
- Bonus: Visualize payoff matrices using heatmaps.

Week 2: Classical Game Simulations

- Simulate repeated games (e.g., Iterated Prisoner's Dilemma) between basic hard-coded strategies (Tit-for-Tat, Always Defect, Random).
- Log payoffs and plot how different strategies perform over time.
- Bonus: Implement noise to simulate imperfect play and analyze impact.

Week 3: Auction Modeling

- Implement at least two auction formats (English, Dutch, or First-Price Sealed Bid).
- Design simple agent strategies (e.g., value-based, random, aggressive) and simulate bidding behavior.
- Analyze which strategy wins most often and under what conditions.

ASSIGNMENTS

Week 4: Reinforcement Learning Basics

- Implement Q-Learning for a grid-world or maze-like environment.
- Visualize how the agent improves over episodes (e.g., using heatmaps of value function).
- Bonus: Add stochasticity to rewards and evaluate robustness of the policy.

Week 5: Multi-Agent Reinforcement Learning (MARL)

- Implement a simple 2-agent environment (e.g., 2-player grid game or resource competition).
- Extend Q-Learning to both agents and analyze convergence (or divergence) of policies.
- Bonus: Compare cooperative vs. selfish reward setups.

Week 6: Strategic Learning Agents

- Choose a known strategic game (e.g., Iterated Public Goods or Rock-Paper-Scissors).
- Train RL agents in this environment and analyze emergent behaviors.
- Document whether equilibria emerge and how agents adapt to each other.

THANK YOU!

CONTACT DETAILS

- **Aayushman Kumar** +91 81088 04449 aayushmank23@iitk.ac.in
- **Jeevesh Narayan** +91 62047 75278 jeevesh23@iitk.ac.in
- **Hardik Tiwari** +91 87569 97008 hardikt23@iitk.ac.in
- **Ikrima Badr Shamim Ahmed** +91 80978 44011 ikrimab23@iitk.ac.in