Introduction to Multiagent Learning

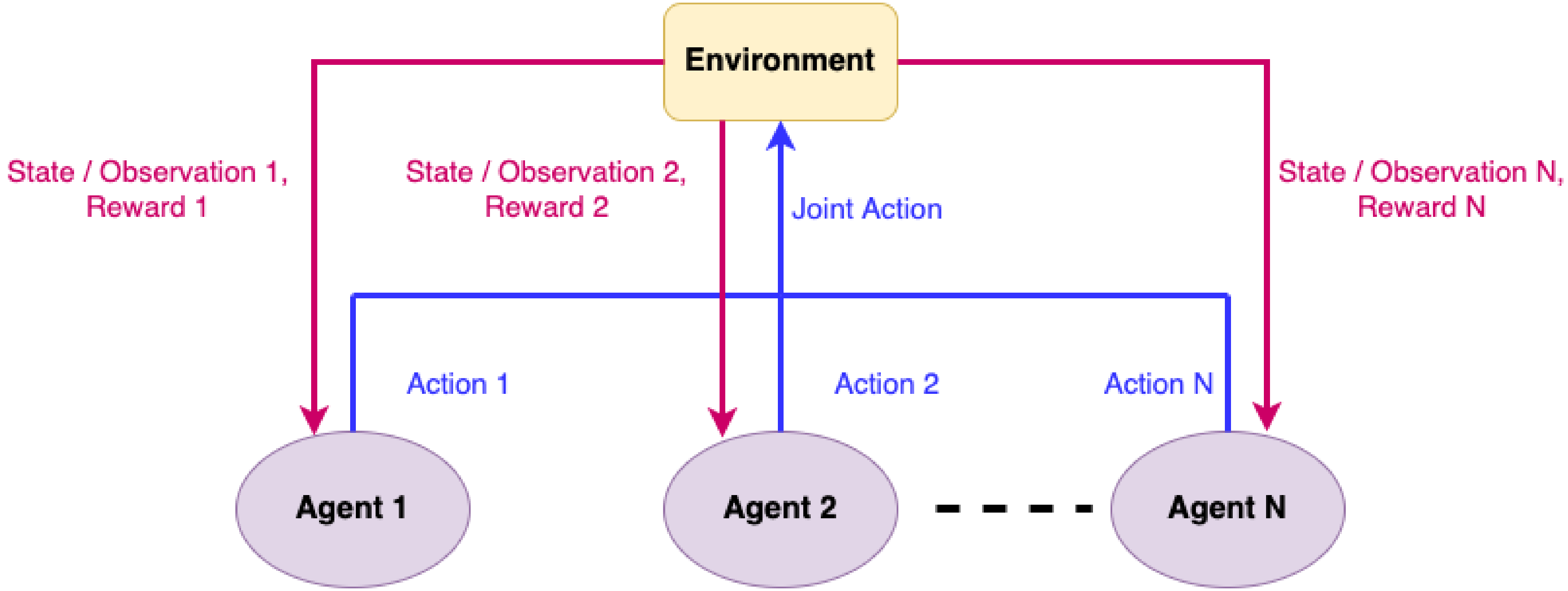
Kate Larson

Cheriton School of Computer Science

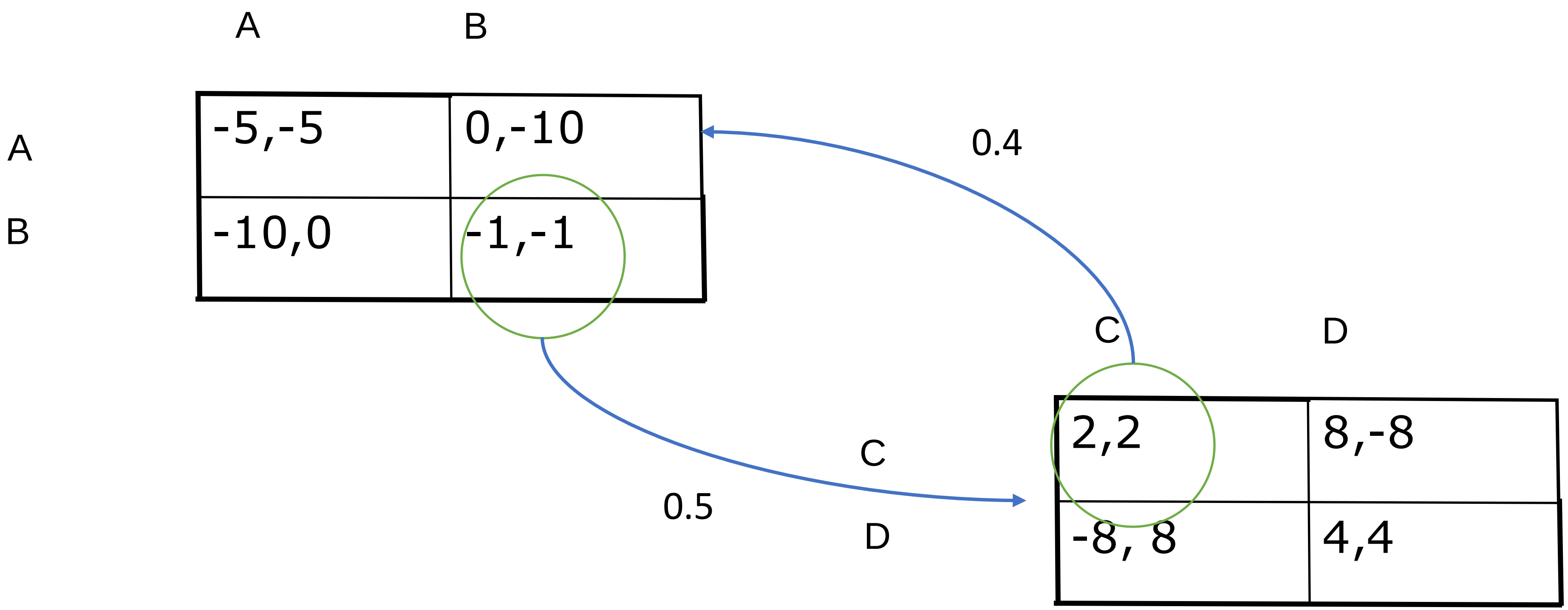
University of Waterloo

UNIVERSITY OF WATERLOO

Multiagent Reinforcement Learning



Stochastic Games (think of this as an n-agent MDP)



Stochastic Game

- Normally represented by a tuple SG=$< N,S,A,R,T,\gamma >$
  - N: set of agents
  - S: state space
  - $A=A_1 \times \ldots \times A_n$: joint action space
  - $R=R_1 \times \ldots \times R_n$: joint reward function
    - $R_i(s,a)$ for $a=(a_1,\ldots,a_n)$ in A
  - T: transition function $T(s',a, s) = P(s'|s, a)$ for $a=(a_1,\ldots,a_n)$
  - γ: discount factor $0< \gamma \leq 1$

Stochastic Games (think of this as an n-agent MDP)

	A	B
A	-5,-5	0,-10
B	-10,0	-1,-1

0.4

0.5

	C	D
C	2,2	8,-8
D	-8, 8	4,4

A={Agent 1, Agent 2}
Two states
Ai={A,B,C,D}
Rewards: R1(s1, (A,A))=-5

Stochastic Game

- Normally represented by a tuple SG=$<$ N,S,A,R,T,$\gamma >$
  - N: set of agents
  - S: state space
  - A=A_1 x …x A_n: joint action space
  - R=R_1x… xR_n: joint reward function
    - $R_i(s,a)$ for $a=(a_1,\dots,a_n)$ in A
  - T: transition function T(s',a, s) = P(s' | s, a) for $a=(a_1,\dots,a_n)$
  - γ: discount factor 0< $\gamma \leq 1$

Policy: $\pi_i: S \rightarrow \Delta Ai$

Goal: Find a policy $\pi^* = (\pi_1^*, \dots, \pi_n^*)$ such that π_i^* = arg max $\sum \gamma^t \sum E[r(s,a)]$ where expectation is conditioned on joint policy π

Playing a Stochastic Game

- Players choose their actions at the same time
  - No communication
  - No observation of the other agents' actions at that time step
- At each stage, players are facing a normal form game
  - Q-values of the current state and joint action are the payoffs for the agents
- Stochastic game is a generalization of a repeated game

Optimal Policies

- Recall, agents are learning in a multi-agent setting
  - Optimal policies should correspond to some equilibrium of the stochastic game

- Nash equilibrium is one example
  - Value function

$$V_i^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi}[r_{i,t} | s_o = s, \pi]$$

  - Nash Equilibrium

$$V_i^{(\pi_i^*, \pi_{-i}^*)}(s) \geq V_i^{(\pi_i, \pi_{-i}^*)}(s), \forall s \in S, \forall i \in N, \forall \pi_i \neq \pi_i^*$$

Independent Learners

- Naïve approach: Each agent uses Q-learning directly, assuming the other agents' are part of the environment

$$Q_i(s, a_i) \leftarrow Q_i(s, a_i) + \alpha(r_i + \gamma \max_{a_i'} Q_i(s', a_i') - Q_i(s, a_i))$$

- Pro: Simple, easy to apply
- Cons:
  - Non-stationary transition and reward models
  - Does not work well against opponents playing complex strategies
  - No convergence guarantees

Opponent Modelling

- We need to have some idea what other agents are doing
  - (but this is not directly observable at time t)
- Agents maintain a **belief** over over the actions taken by other agents
  - Opponent modelling

- Types of opponent modelling
  - Fictitious play
  - Solving unique equilibrium in the stage game
  - Gradient based methods
  - Bayesian approaches

Fictitious Play

- Each agent assumes all others are playing a stationary strategy
- Agents maintain a count of the number of times another agent has taken action a_j in state s

$$n_i^t(s, a_j) \leftarrow 1 + n_i^{t-1}(s, a_j), \forall j, \forall i \in N$$

- Agents update and sample from their belief about this strategy at each stage

$$\mu_i^{j,t}(s) \sim \frac{n_i^t(s, a_j)}{\sum_{a'_j} n_i^t(s, a'_j)}$$

- Agents best-respond according to this belief

Cooperative Stochastic Games

- Normally represented by a tuple SG=< N,S,A,R,T,γ >
  - N: set of agents
  - S: state space
  - $A=A_1$ x …x A_n: joint action space
  - **$R=R_1$x… xR_n: joint reward function**
    - **$R_i(s,a)=R(s,a)$ for $a=(a_1,\dots,a_n)$ in A**
  - T: transition function T(s’,a, s) = P(s’|s, a) for $a=(a_1,\dots,a_n)$
  - γ: discount factor $0< \gamma \leq 1$

Optimal Policies for Cooperative Games

- Pareto dominating (Nash) equilibrium
- Even though rewards/payoffs of agents are aligned, there is still a coordination problem

	A	B
A	2,2	0,0
B	0,0	1,1

Learning in Cooperative Stochastic Games

- Joint Action Learner (JAL) or Joint Q Learning (JQL)
- Must respond to the environment as well as the other agents.
- Similar to Q-learning by agents also include other agents' actions in the update

$$Q_i(s, a_i, a_{-i}) \leftarrow Q_i(s, a_i, a_{-i}) + \alpha(r_i + \gamma \max_{a_i'} Q_i(s', a_i', a_{-i}') - Q_i(s, a_i, a_{-i}))$$

- Two objectives:
  - Agent: find the optimal policy for best response
  - System: Find the NE of the stochastic game (or Nash Q-function of the game)
- Nash Q-function: agent's discounted future rewards when all agents follow the NE policy

Joint Q-Learning

Initialize Q-values

Repeat until convergence of Q values

Repeat for each agent i

- Select and execute a_i
- Observe s', r_i, a_{-i}
- Update counts for states/joint actions: $n(s,a) \leftarrow 1 + n(s,a)$ note that a is the joint action
- Update learning rate: $\alpha \leftarrow 1/n(s,a)$
- Update counts for states/individual agent actions: $n_i(s,a_j) \leftarrow 1 + n_i(s,a_j)$
- Update beliefs:

$$\mu_i^j(s) \sim \frac{n_i(s,a_j)}{\sum_{a'_j} n_i(s,a'_j)}$$

- Update Q-value:

$$Q_i(s,a_i,a_{-i}) \leftarrow Q_i(s,a_i,a_{-i}) + \alpha(r_i + \gamma \max_{a'_i} Q_i(s',a'_i,\mu_i^{-i}(s')) - Q_i(s,a_i,a_{-i}))$$

Convergence of Joint Q-Learning

- If the game is finite, then play will converge to true response to other agents in self-play
  - Self-play: all agents use the same algorithm

- Joint Q-learning converges to Nash Q-values in cooperative stochastic games
  - Every state is visited infinitely often (due to exploration)
  - Learning rate is decreased fast enough but not too fast (same conditions as for Q-learning)
- In cooperative stochastic games, Nash-Q values are unique (unique equilibrium point in terms of utilities)

Joint Q-Learning

Initialize Q-values

Repeat until convergence of Q values

Repeat for each agent i

- **Select and execute a_i**
- Observe s', r_i, a_{-i}
- Update counts for states/joint actions: $n(s,a) \leftarrow 1 + n(s,a)$ note that a is the joint action
- Update learning rate: $\alpha \leftarrow 1/n(s,a)$
- Update counts for states/individual agent actions: $n_i(s,a_j) \leftarrow 1 + n_i(s,a_j)$
- Update beliefs:

$$\mu_i^j(s) \sim \frac{n_i(s,a_j)}{\sum_{a'_j} n_i(s,a'_j)}$$

- Update Q-value:

$$Q_i(s,a_i,a_{-i}) \leftarrow Q_i(s,a_i,a_{-i}) + \alpha(r_i + \gamma \max_{a'_i} Q_i(s',a'_i,\mu_i^{-i}(s')) - Q_i(s,a_i,a_{-i}))$$

Exploration-Exploitation Tradeoff

- Epsilon-greedy
  - Like in the single case, but now you are taking the best-response action given your beliefs
- Boltzmann exploration
  - "Temperature" parameter T (high T increases randomness, low T is less random)

$$P(a) = \frac{e^{\frac{Q_i(s, a_i, \mu_i^{-i}(s))}{T}}}{\sum_{a'} e^{\frac{Q_i(s, a_i', \mu_i^{-i}(s))}{T}}}$$

Summary

- Stochastic Games
- Fictitious Play
- How to learn in Cooperative Stochastic Games