Learning in Multiagent Systems

Reinforcement learning and some issues

Stéphane Airiau

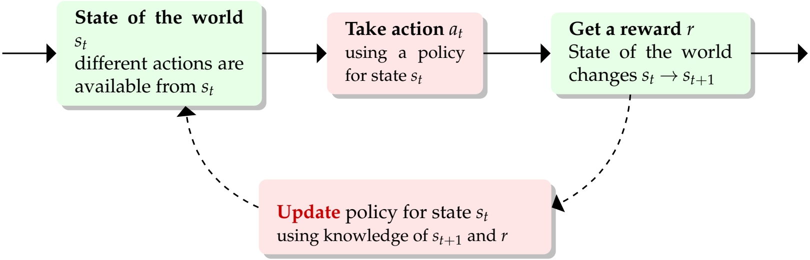
Université Paris Dauphine

Outline

- Reinforcement Learning (single agent)
  - ➩ Learning/solving a Markov decision process (MDP)
- Competitive interactions between two (or more) agents: learning to play a game (a game as in game theory)
  - ➩ Game and some solution concepts
  - ➩ Btw, what are we solving exactly?
- Cooperative interaction: learning to coordinate in a (potentially) large society of agents to reach a collective goal.

Reinforcement learning – single agent learning

Learning from interaction



- **Goal**: obtain as much reward as possible
  assumes that the agent's goals are modeled using utility function,
  $\to$ flexible but may be difficult to elicit
- **Reinforcement Learning:** specify how to update the policy.

Markov assumption

After taking an action a in a state s:

- the reward r obtained
- or the state s' reached

could in principle depend on everything that happened earlier.

However, we assume they depend on the **current state only**: this is called the Markov assumption.

ex: in chess – the state of the game does not depend on the history.

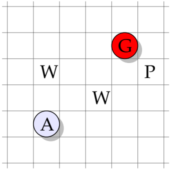
Markov Decision Process

A Markov Decision Process is defined by

- States of the world S
- Action set A
- Transition probabilities: probibility of reaching state $s' \in S$ when one takes action $a \in A$ in state $s \in S$
  ⇨ we write $Pr(s_{t+1} = s \mid s_t = s, a_t = a)$.
- Expected reward: the reward obtained after taking action a in state s when the agent ended up in state s'
  $E\{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\}$.

Example: robot looking for gold in a grid world

- state of the world: a grid $n \times n$
  - some states are walls: if the agent tries to get there, it bumps and remain in the same position.
  - some states are pits (holes): if the agent enters that state, it is the end of the episode and the game restarts
  - one state contain a pot of gold
- actions are moving one cell up, down, left or right. The actions are not deterministic: e.g. wheels may be blocked and the robot may end up in a different neighbouring cell
  ➩ we have a transition probabilities Pr
- reward: if the agent reached the gold, it gets a reward of 100, otherwise, it gets a reward of -1.



The problem

A **policy** $\pi : S \times A \to [0,1]$ is a probability distribution over the action set A telling the probability of taking action $a \in A$ when the agent is in state $s \in S$.

A solution to a Markov Decision process is a policy that "maximises reward".

What are we optimising?

- for episodic tasks:
  - there are some terminal states
  - when an agent reaches a terminal state: reset to a starting state and the agent starts to act

  ➩ maximise the expected return $R_T = r_1 + r_2 + \cdots + r_T$
- for continuing tasks

  ➩ maximise a discounted return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

  γ is called the discounted rate.
  - $\gamma = 0$ the agent is myopic: she cares only about the immediate reward
  - $0 < \gamma < 1$ when $\{r_t, t \in \mathbb{N}\}$ is bounded, R_T is well defined.
    ➩ The agent cares about the immediate reward but also for future ones (but cares more about reward in the near future than in the far one)
- we use continuing tasks
  (one can represent episodic tasks using continuing tasks.)

Value function

How good it is to be in state $s \in S$ when the agent follows policy π?
➫ expected return when starting in s and following π thereafter.

$$V^{\pi}(s) = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\right\}.$$

Similarly, how good is it to take action a in state s following policy π?

$$Q^{\pi}(s,a) = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a\right\}$$

Bellman equation

notation: $P^a_{s\to s'} = E\{s_{t+1} = s \mid s_t = s, a_t = a\}$
$R^a_{s\to s'} = E\{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\}$

$$
\begin{aligned}
V^\pi(s) &= E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t = s\right\} \\
&= E_\pi\left\{r_{t+1} + \gamma\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \mid s_t = s\right\} \\
&= \sum_{a\in A}\pi(s,a)\sum_{s'\in S}P^a_{s\to s'}\left[R^a_{s\to s'} + \gamma E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \mid s_t = s\right\}\right] \\
&= \sum_{a\in A}\pi(s,a)\sum_{s'\in S}P^a_{s\to s'}\left[R^a_{s\to s'} + \gamma V^\pi(s')\right]
\end{aligned}
$$

Optimal Value Functions

we can define a partial order $\succeq$ over policies:
$\pi \succeq \pi'$ iff $\forall s \in S\ V^{\pi}(s) \geqslant v^{\pi'}(s)$

$\pi^\star$ is an optimal policy if it is not dominated by othe policies.

All optimal policies share the same

- state-value function, thus called optimal value function $V^\star = \max_\pi V^\pi(s)$
- action-value function $Q^\star = \max_\pi Q^\pi(s,a)$

Bellman optimality equation

$$
\begin{aligned}
V^{\pi}(s) &= \max_{a \in A} Q^{\pi^\star}(s,a) \\
&= \max_{a \in A} E_{\pi^\star}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right\} \\
&= \max_{a \in A} E_{\pi^\star}\left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s, a_t = a \right\} \\
&= \max_{a \in A} E_{\pi^\star}\{ r_{t+1} + \gamma V^\star(s_{t+1}) \mid s_t = s, a_t = a \} \\
&= \max_{a \in A} \sum_{s' \in S} P^a_{s \to s'} \left[R^a_{s \to s'} + \gamma V^\star(s') \right]
\end{aligned}
$$

Similarly, we have

$$
Q^\star(s,a) = \sum_{s' \in S} P^a_{s \to s'} \left[R^a_{s \to s'} + \gamma \max_{a' \in A} Q^\star(s',a') \right]
$$

solving Bellman optimality equation

For finite MDPs, the Bellman optimality equation has a **unique** solution independent of the policy.

- system of n equations with n unknowns
- many ways to solve for $V^\star$
  - dynamic programming (policy iteration, value iteration)
  - use of Monte Carlo methods for approximations
  - temporal difference learning $\rightarrow$ combine dynamic programming with Monte Carlo methods (Sarsa, Q-learning)
- once $V^\star$ is known, it is easy to compute $Q^\star$

Value Iteration (dynamic programming)

```
for each s ∈ S
    V(s) ← 0

repeat
    Δ ← 0
    for each s ∈ S
      v ← V(s)
      V(s) ← max_{a∈A} Σ_{s'∈S} P^a_{s→s'} [R^a_{s→s'} + γV(s')]
      Δ ← max(Δ, |v − V(s)|)
until Δ < ε
```

Not very useful in practice:

- need to know the dynamics of the environment
- requires large computational resources
- Markov property

RL typically uses an approximation method.

One solution: Q-learning

- We want to estimate the value $Q(s,a)$ of taking action a in a state s.
- The update rule for Q-learning is:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha \left(r_t + \gamma \max_{a \in A(s)} Q(s_t,a) - Q(s_t,a_t) \right),$$

  where α is called the learning rate.

⇨ do not require a model of the environment, only experience.

Exploitation Vs Exploration

Suppose you estimate the value $Q(s,a)$ of taking an action a in state s. What should you do?

- **exploitation**: choose action $a^\star = \text{argmax}_{a \in A(s)} Q(s,a)$
- **exploration**: choose action $a \neq a^\star$

- you cannot exploit all the time (maybe your experience is not enough to make a good choice)
- you cannot explore all the time (at some point, you should use your knowledge), but can never stop exploring (as you are never sure you are doing well)

Two classical methods for trading off exploration and exploitation

- **ϵ-greedy**
  $$a_t = \begin{cases} a^\star = \text{argmax}_{a \in A(s)} Q(s,a) \text{ with probability } 1-\epsilon \\ \text{pick a random action in } A(s) \text{ with probability } \epsilon \end{cases}$$
  ϵ may decrease during learning.
- **Boltzmann softmax**
  uses a temperature parameter T
  pick an action using the distribution in which the probability of picking action a is proportional to $e^{\frac{Q(s,a)}{T}}$.
  T can be decreased during learning.

Partially observable MDP

Only **partial** information about the current state is available.
➯ the agent is uncertain about what the current state is.

the agent senses observations (responses, perceptions, views, etc) that provide some clues about the current state

- many states may share the same observation
- noisy or faulty sensors provide incomplete information from which the agent cannot infer the current state
- combinaison of both

POMDP

A Partially Observable Markov Decision Process is defined by

- States of the world S ✔
- Action set A ✔
- Observation set Ω
- Transition probabilities: probability of reaching state $s' \in S$ when one takes action $a \in A$ in state $s \in S$
  ➩ we write $Pr(s_{t+1} = s \mid s_t = s, a_t = a)$.✔
- Expected reward: the reward obtained after taking action a in state s when the agent ended up in state s' $E\{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\}$.✔
- Observation probability: probability of observing $o \in \Omega$ when action a was taken in state s $\mathcal{O} : S \times A \times \Omega \to [0,1]$

➩ the agent builds a belief about the current state and tries to find the optimal policy.
➩ quite complex, active area of research.

Learning to play a game against another learning agent

interlude about game theory

- Agents have goals, they want to bring about some states of the world, they can take actions in their environment.
- In a multiagent system, agents interact, the actions of one may affect many other agents.
- How can we formally model such interactions?

Game theory is one way.

Prisoner's dilemma

Two partners in crime, Row (**R**) and Column (**C**), are arrested by the police and are being interrogated in separate rooms. From Row's point of view, four different outcomes can occur:

- only R confesses ⇨ R gets 1 year.
- only C confesses ⇨ R spends 4 years in jail
- both confess ⇨ Both spend 3 years in prison.
- neither one confesses ⇨ both get 2 years in prison

The utility of an agent is (5 - number of years in prison).

	Column confesses	Column does not
Row confesses	2,2	4,1
Row does not	1,4	3,3

We can abstract this game and provide a generic game representation as follows:

Definition (Normal form game)

A **normal form game (NFG)** is $(N, (S_i)_{i \in N}, (u)_{i \in N})$ where

- N is the set of n players
- S_i is the set of strategies available to agent i.
- $u_i : S_1 \times \cdots \times S_n \to \mathbb{R}^n$ is the **payoff function** of agent i. It maps a **strategy profile** to a **utility**.

Terminology:

- an element $s = \langle s_1, \ldots, s_n \rangle$ of $S_1 \times \cdots \times S_n$ is called a **strategy profile** or a **joint-strategy**.
- Let $s \in S_1 \times \cdots \times S_n$ and $s_i' \in S_i$. We write (s_i', s_{-i}) the joint-strategy which is the same as s except for agent i which plays strategy s_i', i.e., $(s_i', s_{-i}) = \langle s_1, \ldots, s_{i-1}, s_i', s_{i+1}, \ldots, s_n \rangle$

What would you do?

- $N = \{Row, Column\}$
- $S_{Row} = S_{Column} = \{cooperate,\ defect\}$
- u_{Row} and u_{Column} are defined by the following bi-matrix.

$Row \setminus Column$	defect	cooperate
defect	2,2	4,1
cooperate	1,4	3,3

1. Wait to know the other action?
2. Not confess?
3. Confess?
4. Toss a coin?

Can you use some general principles to explain your choice?

Definition (strong dominance)

A strategy $x \in S_i$ for player i **(strongly) dominates** another strategy $y \in S_i$ if independently of the strategy played by the opponents, agent i (strictly) prefers x to y, i.e. $\forall s \in S_1 \times \cdots \times S_n$, $u_i(x, s_{-i}) > u_i(y, s_{-i})$

Prisoner's dilemma

	C confesses	C does not
R confesses	**2,2**	4,1
R does not	1,4	3,3

Both players have a dominant strategy: to confess! From Row's point of view:

- if C confesses: R is better off confessing as well.
- if C does not: R can exploit and confess.

Battle of the sexes

	L	R
T	2,2	4,3
B	3,4	1,1

- **Problem:** Where to go on a date: Soccer or Opera?
- **Requirements:**
  - have a date!
  - be at your favourite place!

Do players have a dominant strategy?

Definition (Best response)

A strategy s_i of a player i is a **best response** to a joint-strategy s_{-i} of its opponents iff

$$\forall s_i' \in S_i,\ u_i(s_i, s_{-i}) \geqslant u_i(s_i', s_{-i}).$$

Definition (Nash equilibrium)

A joint-strategy $s \in S_1 \times \cdots \times S_n$ is a **Nash equilibrium** if each s_i is a best response to s_{-i}, that is

$$(\forall i \in N)\left(\forall s_i' \in S_i\right)\ u_i(s_i, s_{-i}) \geqslant u_i(s_i', s_{-i})$$

Battle of the sexes possesses two Nash equilibria $\langle T, R\rangle$ and $\langle B, L\rangle$.

A **Nash equilibrium** is a joint-strategy in which no player could improve their payoff by unilaterally deviating from their assigned strategy.

Prisoner's dilemma

	C confesses	C does not
R confesses	**2,2**	4,1
R does not	1,4	3,3

Unique Nash equilibrium: both players confess!

- if R changes unilaterally, R loses!
- if C changes unilaterally, C loses!

Definition (Pareto optimal outcome)

A joint-strategy s is a **Pareto optimal outcome** if for no joint-strategy s'
$\forall i \in N\ u_i(s') \geqslant u_i(s)$ and $\exists i \in N u_i(s') > u_i(s)$

A joint-strategy is a Pareto optimal outcome when there is no outcome that is better for all players.

Prisoner's dilemma: Remaining silent is Pareto optimal.

discussion: It would be **rational** to confess! This seems counter-intuitive, as both players would be better off by keeping silent.

➯ There is a conflict: the **stable** solution (i.e., the Nash equilibrium) is not **efficient**, as the outcome is not Pareto optimal.

Chicken game

In *Rebel Without a Cause*, James Dean's character's, Jim, is challenged to a "Chickie Run" with Buzz, racing stolen cars towards an abyss. The one who first jumps out of the car loses and is deemed a "chicken" (coward).

	Jim drives on	Jim turns
Buzz drives on	-10,-10	5,0
Buzz turns	0,5	1,1

Dominant Strategy?

Nash equilibrium ?

Nash equilibrium

- When there is no dominant strategy, an equilibrium is the next best thing.
- A game may not have a Nash equilibrium.
- If a game possesses a Nash equilibrium, it may not be unique.
- Any combinations of dominant strategies is a Nash equilibrium.
- A Nash equilibrium may not be Pareto optimal.
- Two Nash equilibria may not have the same payoffs

Definition (Mixed strategy)

A mixed strategy p_i of a player i is a probability distribution over its strategy space S_i.

Assume that there are three strategies: $S_i = \{1,2,3\}$. Player i may decide to play strategy 1 with a probability of $\frac{1}{3}$, strategy 2 with a probability of $\frac{1}{2}$ and strategy 3 with a probability of $\frac{1}{6}$. The mixed strategy is then denoted as $\left\langle \frac{1}{3}, \frac{1}{2}, \frac{1}{6} \right\rangle$.

Given a mixed strategy profile $p = \langle p_1, \ldots, p_n \rangle$, the expected utility for agent i is computed as follows:

$$E_i(p) = \sum_{s \in S_1 \times \cdots \times S_n} \left(\left(\prod_{j \in N} p_j(s_j) \right) \times u_i(s) \right)$$

Battle of the sexes

		y	$1-y$
		L	R
x	T	2,2	4,3
$1-x$	B	3,4	1,1

The expected utility for the Row player is:

$xy \cdot 2 + x(1-y) \cdot 4 + (1-x)y \cdot 3 + (1-x)(1-y) \cdot 1$

$= -4xy + 3x + 2y + 1$

Given a mixed strategy profile $p = \langle p_1, \ldots, p_n \rangle$, we write (p'_i, p_{-i}) the mixed strategy profile which is the same as p except for player i which plays mixed strategy p'_i, i.e., $(p'_i, p_{-i}) = \langle p_1, \ldots, p_{i-1}, p'_i, p_{i+1}, \ldots, p_n \rangle$.

Definition (Mixed Nash equilibrium)

A **mixed Nash equilibrium** is a mixed strategy profile p such that $E_i(p) \geqslant E_i(p'_i, p_i)$ for every player i and every possible mixed strategy p'_i for i.

Battle of the sexes

	L	R
T	2,2	4,3
B	3,4	1,1

Let us consider that each player plays the mixed strategy $\langle \frac{3}{4}, \frac{1}{4} \rangle$. None of the players have an incentive to deviate:

$$E_{row}(T) = \frac{3}{4} \cdot 2 + \frac{1}{4} \cdot 4 = \frac{5}{2} \qquad E_{row}(B) = \frac{3}{4} \cdot 3 + \frac{1}{4} \cdot 1 = \frac{5}{2}$$

(players are indifferent)

Theorem (J. Nash, 195))

Every finite strategic game has got at least one mixed Nash equilibrium.

note: The proofs are non-constructive and use Brouwer's or Kakutani's fixed point theorems.

J.F. Nash. Equilibrium points in *n*-person games. in *Proc. National Academy of Sciences of the United States of America*, 36:48-49, 1950.

Computing a Nash equilibrium

Complexity: In general, it is a hard problem. It is a PPAD-complete problem.

Daskalakis, Goldberg, Papadimitriou: **The complexity of computing a Nash equilibrium**, in *Proc. 38th Ann. ACM Symp. Theory of Computing (STOC)*, 2006

There are complexity results and algorithms for different classes of games. We will not treat then in this tutorial.

Y. Shoham & K. Leyton-Brown: **Multiagent Systems**, Cambridge University Press, 2009. (Chapter 4)

Nisan, Roughgarden, Tardos & Vazirani: **Algorithmic Game Theory**, Cambridge University Press, 2007. (chapters 2, 3)

Other types of solution concepts for NFGs

Safety strategy

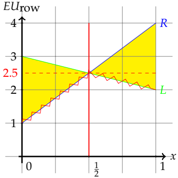
With Nash equilibrium, we assumed that the opponents were **rational agents**. What if the opponents are potentially **malicious**, i.e., their goal could be to minimize the payoff of the player?

Definition (Maxmin)

For player i,
the **maxmin strategy** is $\arg\max_{s_i \in S_i} \min_{s_{-i} \in S_{-i}} u_i(s_i, s_{-i})$,
and its **maxmin value** or **safety level** is $\max_{s_i \in S_i} \min_{s_{-i} \in S_{-i}} u_i(s_i, s_{-i})$.

1) player i chooses a (possibly mixed) strategy.
2) the opponents $-i$ choose a (possible mixed) strategy that minimize i's payoff.
➩ the maxmin strategy maximizes i's **worst case** payoff.

		y	$1-y$
		L	R
x	T	2,2	4,3
$1-x$	B	3,4	1,1



Payoff of Row
when Column plays pure strategy (T or R)
or any mixed strategy (yellow area)

Whatever Column does, Row can guarantee itself a payoff of 2.5 by playing the mixed strategy $\langle \frac{1}{2}, \frac{1}{2} \rangle$.

Minimax regret

Instead of assuming the opponents are rational (Nash equilibrium) or malicious (minimax), one can assume the **opponent is unpredictable** ⇨ avoid **costly mistakes**/minimize their worst-case losses.

	L	R
T	100,100	0,0
B	0,0	1,1

(T, L) is preferred by both agents.
However, (B, R) is also a NE.
There is no dominance.
How to explain that (T, L) should be preferred?

One can build a **regret-recording** game where the payoff function r_i is defined by $r_i(s_i, s_{-i}) = u_i(s_i^\star, s_{-i}) - u_i(s_i, s_{-i})$, where $s_i^\star$ is i's best response to s_{-i}, i.e., $r_i(s_i, s_{-i})$ is i's **regret to have chosen s_i instead of $s_i^\star$**.

$r_i \backslash r_j$	L	R
T	**0,0**	1,100
B	100,1	0,0

We define $regret_i(s_i)$ as the maximal regret i can have from choosing s_i.
A **regret minimization strategy** is one that **minimizes the $regret_i$ function**.

Repeated games

Prisoner's dilemma

	Defect	Cooperate
Defect	2,2	4,1
Cooperate	1,4	3,3

When players are **rational**, both players confess!
If they trusted each other, they could both not confess and obtain $\langle 3,3 \rangle$.
If the same players have to repeatedly play the game, then it could be rational not to confess.

- **One shot games**: there is no tomorrow.
  This is the type of games we have studied thus far.
- **Repeated games**: model a likelihood of playing the game again with the same opponent. The NFG (N, S, u) being repeated is called the **stage game**.
  - finitely repeated games ⇨ represent using a EFG and use backward induction to solve the game.
  - infinitely repeated games: the game tree would be infinite, use different techniques.

What is a strategy? In a repeated game, a **pure strategy** depends also on the **history** of play thus far.

ex: Tit-for-Tat strategy for the prisoner's dilemma:
Start by not confessing. Then, play the action played by the opponent during the previous iteration.

What is the players' objective?

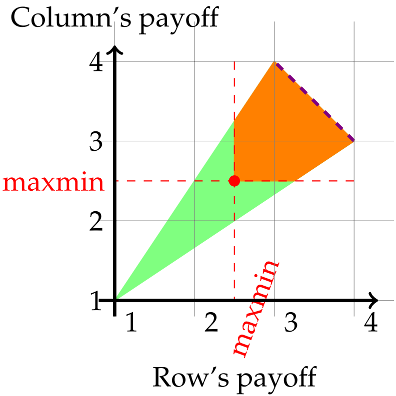
- **Average criterion:** Average payoff received throughout the game by player i: $\lim_{t\to\infty} \frac{\sum_{t=1}^{k} u_i(s^t)}{k}$, where s^t is the joint-strategy played during iteration t.
- **Discounted-sum criterion:** Discounted sum of the payoff received throughout the game by player i: $\sum_{t=0}^{\infty} \gamma^t u_i(s^t)$, where γ is the discount factor (γ models how much the agent cares about the near term compared to long term).

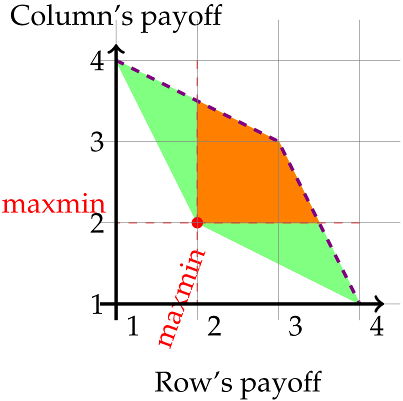
Theorem (A Folk theorem)

Using the average criterion, any payoff vector v such that

- v is **feasible**, i.e., $\exists \lambda \in [0,1]^{\prod_{j\in N}|S_j|}$ s.t. $v_i = \sum_{s\in\prod_{j\in N} S_j} \lambda_s v_i(s)$
- v is **enforceable** $v_i \geqslant \max_{s_i\in S_i} \min_{s_{-i}\in S_{-i}} u_i(s_i, s_{-i})$

can be sustained by a Nash equilibrium.





- In repeated games, the **same** stage game was played repeatedly.
- A **Stochastic game** is a set of NFGs. The agents **repeatedly** play games from this set. The next game is chosen with a probability which depends on the current game and the joint-action of the players.

Definition (Stochastic games)

A stochastic game is tuple $(N, (S_i)_{i \in N}, Q, P, (u_i)_{i \in N})$ where

- N is the set of players
- S_i is the strategy space of player i
- Q is a set of NFGs $q = (N, (S_i)_{i \in N}, (v_i^q)_{i \in N})$
- $P : Q \times \prod_{i \in N} S_i \times Q \to [0,1]$ is the **transition function**.
  $P(q, s, q')$ is the probability that game q' is played after game q when the joint-strategy s was played in game q.
- $u_i : Q \times \prod_{i \in N} S_i$ is the **payoff function**
  $u_i(q, s)$ is the payoff obtained by agent i when the joint-strategy s was played in game q.

- For stochastic games, the players know which game is currently played, i.e., they know the players of the game, the actions available to them, and their payoffs.
- In **Bayesian games**,
  - there is **uncertainty** about the game currently being played.
  - players have private information about the current game. The definition uses **information set**.

Back to Learning! (finally!)

Learning to play a repeated game

	Soccer	Opera
Soccer	3,4	1,1
Opera	2,2	3,4

Battle of the sexes

	Defect	Cooperate
Defect	2,2	4,1
Cooperate	1,4	3,3

Prisoners' dilemma

Assumptions:

- each player can observe the action taken by its opponent (perfect information)
- a player may not know the payoff of the other agent (incomplete information)
- the game is played repeatedly

we could make it more complex using a stochastic game.

⇨ all theoretical results about solving single-agent MDPs no longer apply!

What are we trying to do?

- descriptive approach: study the way learning takes place in real life
  - ➯ show similarities between the formal model and nature
  - ➯ it is interesting if the formal model possesses some nice properties (e.g. convergence to a solution concept)
    - convergence to Nash equilibrium of the stage game?
    - frequency of play converges to Nash equilibrium
    - convergence to a special Nash equilibrium of the repeated game (e.g. that is also Pareto efficient).
- Prescriptive theory: how (artificial) agents should learn.
  - a learning rule should guarantee at least its maxmin payoff (safety/Individual rationality)
  - if the opponent(s) play a stationary strategy, the learning rule should play a best-response to that strategy.
  - a learning strategy should have no regret.
  - learning rule should converge in self play.

First algorithm: Fictitious Play

The learner believes its opponent is playing a fixed mixed strategy given by the empirical distribution of the opponents previous action.

➯ the learner plays a best response to this mixed strategy.

```
intialize frequencies of the actions played by the opponent
repeat
  play a best response to p
  observe the action played by the opponent
   and update frequencies
```

Theorem

If the empirical distribution of each player's strategies converges in fictitious play, the it converges to a Nash equilibrium

- the play converges to a NE, but the players may not play a NE and may not receive a NE expected payoff (ex anti-coordination game)
- convergence is not always guaranteed (ex Rock-paper-cisors)

Joint Action Learning

- consider cooperative games
- ⇨ observing its own payoff is enough
- learns Q values for joint-actions
- update of Q-learning is $Q(a) \leftarrow Q(a) + \alpha(r - Q(a))$

Nash-Q

- assumes a stochastic game
- must observe payoff of all players
- learns Q values for joint-actions
- update of Q-learning is
  $Q(s,a_1,\ldots,a_n) \leftarrow (1-\alpha)Q(s,a_1,\ldots,a_n) + \alpha(r + \beta NashQ(s')$
  where *NashQ* is the payoff of a selected Nash equilibrium
- converges to Nash equilibrium under some conditions
- improvements with *Friend of Foe Q-learning [Littman 01]*

Gradient ascent and hill climbing

- *Infinitesimal Gradient Ascent* (IGA) policy gradient ascent (convergence not guaranteed for all games)
- *Generalized* IGA $\rightarrow$ use regret based learning
  IGA converges to a Nash equilibrium when the game has a pure Nash equilibrium.
- *Win or Lose Fast* IGA (WoLF-IGA)
  Converges to NE for two-agent two action games
- Policy Hill Climber (PHC) and WoLF-PHC

Comparison

It is difficult to compare these algorithms

- may have guarantee in self play
- some algorithms do better on certain games, against some opponents
- What criteria to use for comparison? On what testbed? What ranking method to use?

Powers and Shoham 05, Airiau & Sen 05

Application to controlling a multiagent system

Scenario

- Collection of autonomous learning agents (e.g. robots, uavs, traffic controllers) works **for a system designer**
- The system designer wants to optimize a **collective criterion** (e.g. some objective function)
- ➾ The utility function of the agents can be set up by the system designer.
- Agents cannot explicitly reason and communicate to reach the goal (system is too large, too difficult to compute).
- Agents only use their own experience

How to set up the individual utility functions so that, when each agents optimize its personal utility, the system converges to a good state?

Difference Utility

- $N = \{1, \ldots, n\}$ is the set of agents
- $A = \{a_1, \ldots, a_k\}$ is the set of actions available to each agent
- $z \in A^N$ is the joint-action of the agents in the system (this may contain many entries)
  $\rightarrow z_i$ is the action of agent i
- $G : A^N \rightarrow \mathbb{R}$ is the collective utility function (set by the system designer).

The difference reward for agent i is of the form:

$$D_i = G(z) - G(z - z_i \cdot e_i + c_i \cdot e_i),$$

where $e_i \in A^n$ such that $e_i(j) = 0$ if $i \neq j$ and $e_i(i) = 1$.

$$D_i = G(z) - G(z - z_i \cdot e_i + c_i \cdot e_i),$$

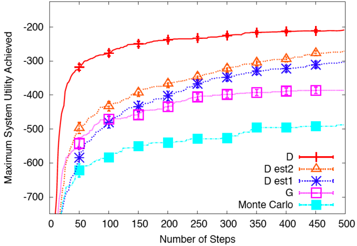
the action of agent i z_i is replaced by c_i

Sometimes, it is possible to choose c_i such that $z - z_i \cdot e_i + c_i \cdot e_i$ is **as if** i left the system.

⇨ D evaluates the contribution of agent i

- better signal ("learnability")
- As $G(z - z_i \cdot e_i + c_i \cdot e_i)$ does not depend on i, any action that improves D_i also improves G! ("factoredness")

The form of G may be complex, but sometimes, each agent can "easily" approximate its D_i.



Tumer & Agogino (AAMAS-07)

Application to air-traffic control

Conclusion

- Multiagent learning is an active area of research
- Has the potential to be useful in many applications
- In this talk, I focused on learning repeated games. There are more general classes of games (e.g. stochastic games) for which there are some algorithms.
- There are also games for which a game theoretic approach may not be feasible (e.g. RoboCup soccer)

Some events

- Workshop at AAMAS (ALA Adaptive and Learning Agents)
- Tutorial this year at AAMAS