Apprentissage multi-agent par renforcement

Rand ASSWAD

INSA Rouen Normandie
Département Génie Mathématique

Plan

Apprentissage par renforcement

Apprentissage par renforcement - Définition

Un agent interagit séquentiellement avec son environnement.
Son environnement est caractérisé par:

- S : l'espace d'états de l'agent
- A : l'espace d'actions de l'agent
- $T : S \times A \times S \to \mathbb{R}$ la fonction de transition dans l'environnement

$$T(s,a,s') = \mathbb{P}(s_{t+1} = s' | s_t = s, a_t = a)$$

- $R : S \times A \to \mathbb{R}$ la fonction de récompense, $R(s,a)$ est la récompense que l'agent obtient en faisant l'action a à l'état s.
- $\pi : S \times A \to [0,1]$ la politique d'action à chaque état, $\pi(s,a)$ est la probabilité d'effectuer l'action a à l'état s

Apprentissage par renforcement - Définition

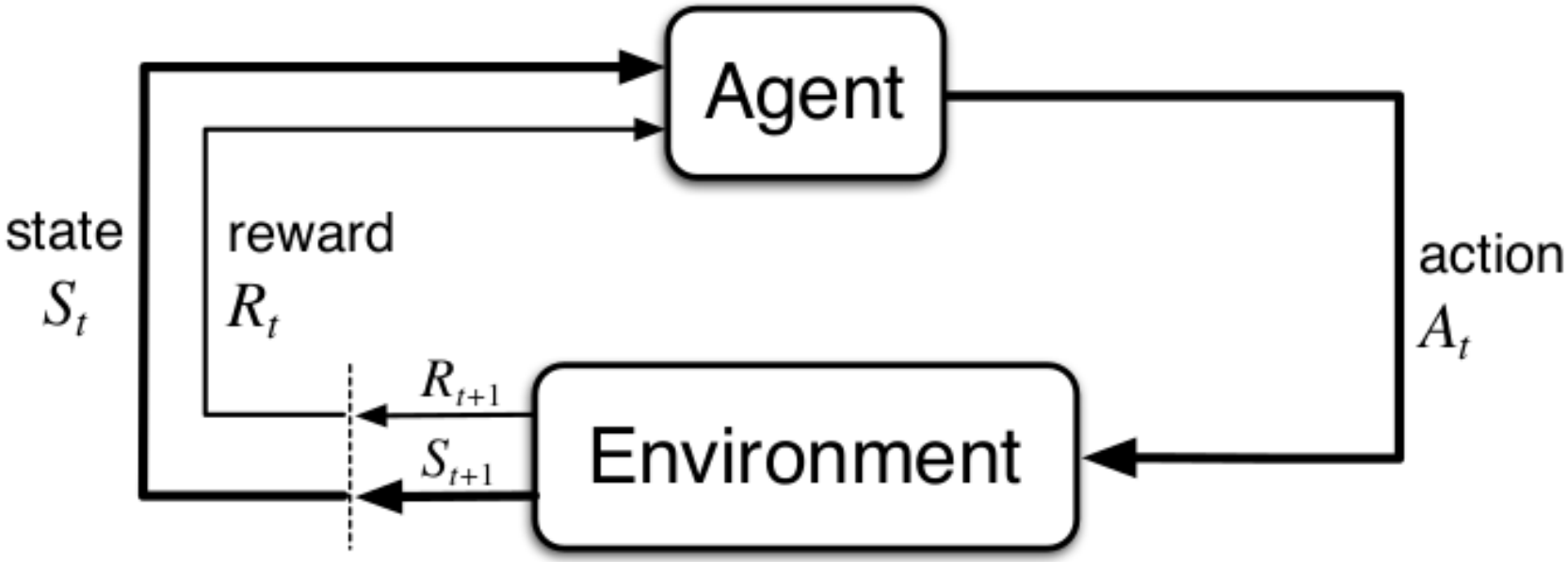


FIG.: Interaction d'un agent avec son environnement

Apprentissage par renforcement - Problématique

Objectif: Trouver une politique optimale pour l'agent

$$\pi(s,a) = \mathbb{P}(a_t = a | s_t = s)$$

Critère: Maximiser les récompenses dans le futur.

Apprentissage par renforcement - politiques optimales

Définition (Politique déterministe)

Une politique est déterministe si elle associe à chaque état une seule action (les autres ont une probabilité nulle).
Soit $\pi : S \rightarrow A$.

Définition (Politique stochastique)

Une politique est stochastique si pour chaque état elle associe une distribution de probabilité sur les actions.
On note $\pi : S \rightarrow \mathcal{PD}(A)$ ou $\pi : S \times A \rightarrow [0,1]$.

Apprentissage par renforcement - Récompenses

- **Modèle fini simple:** la somme de récompenses pour les prochaines M étapes.
$$\mathbb{E}\left\{\sum_{j=0}^{M} r_{t+j}\right\}$$
- **Modèle infini amorti:** la somme de récompenses *amortis* au long-terme.
$$\mathbb{E}\left\{\sum_{j=0}^{\infty} \gamma^j r_{t+j}\right\} \quad \gamma \in [0,1]$$
- **Modèle infini moyen:** la *moyenne* de récompenses au long-terme.
$$\lim_{M\to\infty} \mathbb{E}\left\{\frac{1}{M}\sum_{j=0}^{M} r_{t+j}\right\}$$

Processus de décision markovien (MDP)

Définition (MDP)

Un processus de décision markovien (Markov Decision Process, **MDP**) est un modèle stochastique permettant de définir une politique stochastique de l'agent.

Une fois la politique déterminée, la fonction de transition se réduit à

$$T(s,s') = \mathbb{P}(s_{t+1} = s' | s_t = s)$$

résultant d'un agent dont le comportement est une chaîne de Markov.

Comment déterminer la politique optimale ?

La résolution des MDP fait appelle à la **programmation dynamique** [Bellman, 1957].
Méthodes classiques utilisées:

- Value Iteration
- Policy Iteration

Ceux deux méthodes se basent sur les équations d'optimalité de Bellman.

$$V(s) = \max_{a \in A} Q(s,a) \tag{1}$$

$$Q(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V(s') \tag{2}$$

- $V(s)$ la *valeur* de l'état s
- $Q(s,a)$ la *qualité* de l'action a à l'état s

Value Iteration (Model-based)

Initialiser $V(s)$ arbitrairement
répéter
  pour chaque $s \in S$ **faire**
    pour chaque $a \in A$ **faire**
      $Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V(s')$
    fin
    $V(s) \leftarrow \max_{a \in A} Q(s,a)$
  fin
jusqu'à $V(s)$ *converge* $\forall s \in S$
$\pi(s) \leftarrow \underset{a \in A}{\operatorname{argmax}}\, Q(s,a), \forall s \in S$

Q-learning (Model-free) [Watkins, 1989]

Idée:

$$Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in S} \cancel{T(s,a,s')} V(s')$$

Principe: En lançant l'agent à un état s, il passera à l'état s' en effectuant l'action a avec une probabilité de $T(s,a,s')$ *à la limite*.

Méthode:

- $Q(s,a) \leftarrow r + \gamma V(s')$
- Facteur d'apprentissage $\alpha_t \in [0,1]$

Q-learning Pseudo-code

Initialiser $Q(s,a)$ arbitrairement
Initialiser s
boucle
  $a \leftarrow \underset{a \in A}{\operatorname{argmax}}\, Q(s,a)$
  Effectuer l'action a, recevoir une récompense r et arriver à l'état s'
  $V(s') \leftarrow \max_{a' \in A} Q(s',a')$
  $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\,(r + \gamma V(s'))$
  $s \leftarrow s'$
fin
$\forall s \in S, \pi(s) \leftarrow \underset{a \in A}{\operatorname{argmax}}\, Q(s,a)$

Apprentissage multi-agent par renforcement

Du mono-agent au multi-agent

Caractéristiques de l'environnement:

- $\mathbf{A} = A_1 \times \ldots \times A_n$: l'espace d'actions des agents où A_i est l'espace d'actions de l'agent i.
- $R_i : S \times \mathbf{A} \rightarrow \mathbb{R}$ la fonction de récompense pour l'agent i.

Remarques:

- Du point de vue de chaque agent, l'environnement n'est plus stationnaire.
- La politique d'un agent dépend de ses adversaires/camarades, il faut donc apprendre à coopérer et/ou faire des compromis.

Systèmes multi-agent et théorie des jeux

La théorie des jeux étudie les modèles mathématiques de prise de décision *rationnelle* des agents du système.
On s'intéresse à deux modèles :

- Jeu de matrice (*anglais:* Matrix Game)
- Jeu stochastique (jeu markovien, *anglais:* Markov Game)

Les jeux se catégorisent aussi selon la nature des relations entre les agents du systèmes.

- **Jeu à somme nulle (anglais: zero-sum game)**
- Jeu d'équipe (*anglais:* team game)
- Jeu à somme générale (*anglais:* general-sum game)

Définition (Jeu à somme nulle)

Un jeu à somme nulle est un jeu où les récompenses des agents s'additionne à 0 à chaque moment du jeu. i.e. $\forall t, \sum_{i=0}^{n} R_i(s_t, \mathbf{a}_t) = 0$

Jeu de matrice

Définition (Jeu de matrice)

Un jeu de matrice est un jeu à plusieurs joueurs à un seul état où la récompense de chaque agent dépend des actions prises par tous les agents.
C'est-à-dire, $R_i : \mathbf{A} \rightarrow \mathbb{R}$.

Exemple (Jeu Chifoumi)

Littman étudie l'exemple de deux joueurs du jeu « Chifoumi ».

- Les deux agents partagent le même espace d'état S
- La somme de chaque ligne est (0,0)

	Rock	Paper	Scissors
Rock	(0 , 0)	(-1 , 1)	(1 , -1)
Paper	(1 , -1)	(0 , 0)	(-1 , 1)
Scissors	(-1 , 1)	(1 , -1)	(0 , 0)

Résolution d'un jeu de matrice à deux joueurs

Notation (Jeu de matrice à deux joueurs)

On note a l'action du premier agent, et o l'action de son adversaire.
La récompense du premier agent se note alors $R_{a,o}$.

Problème (politique optimale)

Toute politique déterministe peut être battue systématiquement.
On cherche une politique stochastique optimale $\pi \in \mathcal{PD}(A)$ pour laquelle le premier agent gagne le plus de récompenses futurs.

Résolution d'un jeu de matrice à deux joueurs

Proposition (Politique optimale [Littman, 1994])

Une politique optimale maximise la récompense minimale de l'agent. Cette politique à une récompense attendue d'au moins V pour toute action possible de l'adversaire.

$$V = \max_{\pi \in \mathcal{PD}(\mathbf{A})} \min_{o \in O} \sum_{a \in A} R_{a,o}\pi$$

Exemple (chifoumi)

$$\begin{array}{rcrcrcl}
 & & \pi_{\text{feuille}} & - & \pi_{\text{ciseaux}} & \geq & V \quad \text{(vs. pierre)} \\
- & \pi_{\text{pierre}} & & + & \pi_{\text{ciseaux}} & \geq & V \quad \text{(vs. feuille)} \\
 & \pi_{\text{pierre}} & - \ \pi_{\text{feuille}} & & & \geq & V \quad \text{(vs. ciseaux)} \\
 & \pi_{\text{pierre}} & + \ \pi_{\text{feuille}} & + & \pi_{\text{ciseaux}} & = & 1
\end{array}$$

Programmation linéaire $\rightsquigarrow V = 0$ et $\pi = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$.

Jeu stochastique

Définition (Jeu stochastique)

Les jeux stochastiques (introduits par [Shapley, 1953]) sont des extensions des jeux de matrice et/ou MDP car elles traîtent les cas multi-agents dans une situation multi-état [Neto, 2005].

Proposition (Politique minimax [Littman, 1994])

Dans un jeu stochastique, une politique optimale s'obtient en appliquant les méthodes de résolution de MDP où:

- *$V(s)$ est estimé par le principe minimax des jeux de matrice*
- *Les récompenses futurs sont estimées par $Q(s,a,o)$*

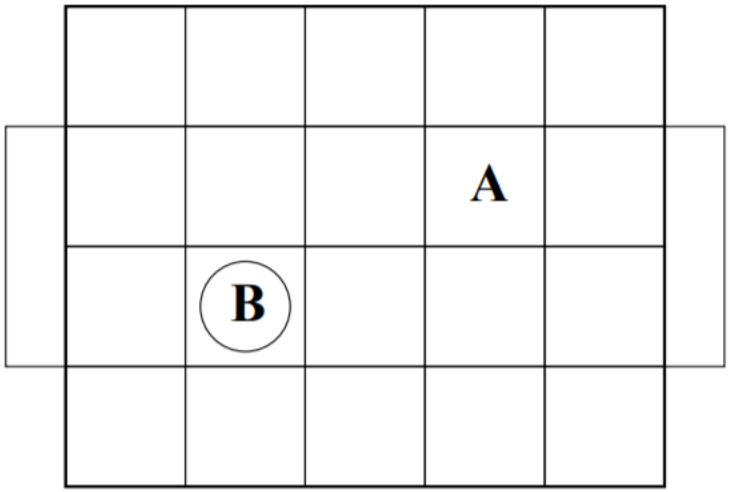
$$V(s) = \max_{\pi \in \mathcal{PD}(\mathbf{A})} \min_{o \in O} \sum_{a \in A} Q(s,a,o)\pi \tag{3}$$

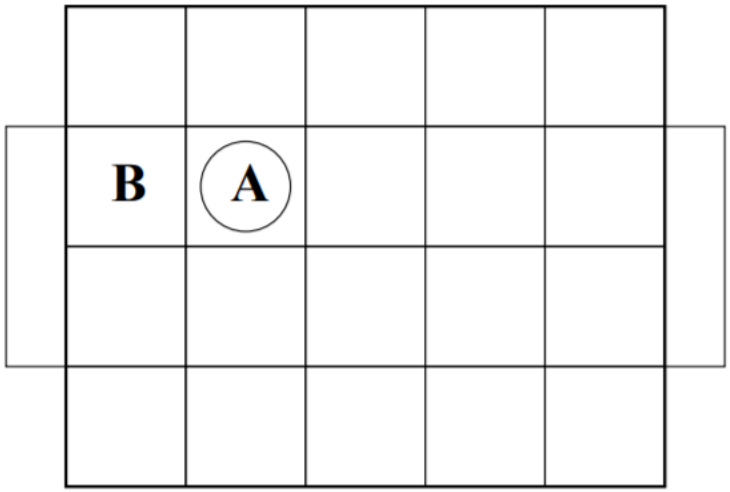
$$Q(s,a,o) = R(s,a,o) + \gamma \sum_{s' \in S} T(s,a,o,s')V(s') \tag{4}$$

Algorithme Minimax-Q [Littman, 1994]

Initialiser $V(s)$ et $Q(s,a,o)$ à 1,$\forall s \in S, a \in A, o \in O$
$\pi(s,a) \leftarrow 1/|A|, \forall s \in S, a \in A$
$\alpha \leftarrow 1$
Initialiser s
boucle
  si *avec une probabilité* p_{explor} **alors**
    $a \leftarrow$ tirée aléatoirement suivant une loi uniforme sur A
  sinon
    $a \leftarrow$ tirée aléatoirement avec une probabilité $\pi(s,a)$
  fin
  Réaliser (s,a,o,s',r)
  $Q(s,a,o) \leftarrow (1-\alpha)Q(s,a,o) + \alpha(r + \gamma V(s'))$
  $\pi(s) \leftarrow \underset{\pi' \in \mathcal{PD}(A)}{\operatorname{argmax}} \min_{o' \in O} \sum_{a' \in A} Q(s,a',o')\pi'(s,a')$ (programme linéaire)
  $V(s) \leftarrow \min_{o' \in O} \sum_{a' \in A} Q(s,a',o')\pi(s,a')$
  $\alpha \leftarrow \alpha \cdot \delta$
  $s \leftarrow s'$
fin
$\forall s \in S, \pi(s) \leftarrow \underset{a \in A}{\operatorname{argmax}} Q(s,a)$

Example d'application de Minimax-Q: football





- **MR:** agent minimax-Q entraîné contre un agent aléatoire.
- **MM:** agent minimax-Q entraîné contre un agent minimax-Q.
- **QR:** agent Q-learning entraîné contre un agent aléatoire.
- **QQ:** agent Q-learning entraîné contre un agent Q-learning.

Evaluation des agents

Chaque agent est évalué contre trois adversaires:

- Agent d'une politique aléatoire
- Agent d'une politique déterministe implémentée à la main
- Agent *challenger*

	MR		MM		QR		QQ	
	% won	games	% won	games	% won	games	% won	games
vs. random	99.3	6500	99.3	7200	99.4	11300	99.5	8600
vs. hand-built	48.1	4300	53.7	5300	26.1	14300	76.3	3300
vs. MR-challenger	35.0	4300						
vs. MM-challenger			37.5	4400				
vs. QR-challenger					0.0	5500		
vs. QQ-challenger							0.0	1200

Notion d'optimalité

Définitions I

Définition (Politique commune, *anglais:* joint policy)

Dans un système multi-agents, une politique commune est une collection de n politiques (une pour chaque agent), elle s'écrit $\pi = (\pi_i, \pi_{-i})$ où π_{-i} est la politique commune de tous les agents sauf l'agent i.

Définition (Best-responce policy)

Une politique individuelle est dîte *best-response policy* si pour une politique commune de tous les autres agents π_{-i} elle réalise la récompense la plus élevée possible.
On note $\pi_i \in \mathcal{BR}_i(\pi_{-i})$ où $\mathcal{BR}_i(\pi_{-i})$ représente l'ensemble des meilleurs politiques (best-response policies) pour l'agent i face à la politique commune π_{-i}.

$$\pi_i^* \in \mathcal{BR}_i(\pi_{-i}) \Leftrightarrow \forall \pi_i \in \mathcal{PD}(A_i), R_i((\pi_i^*, \pi_{-i})) \geq R_i((\pi_i, \pi_{-i}))$$

Définitions II

Définition (Équilibre de Nash)

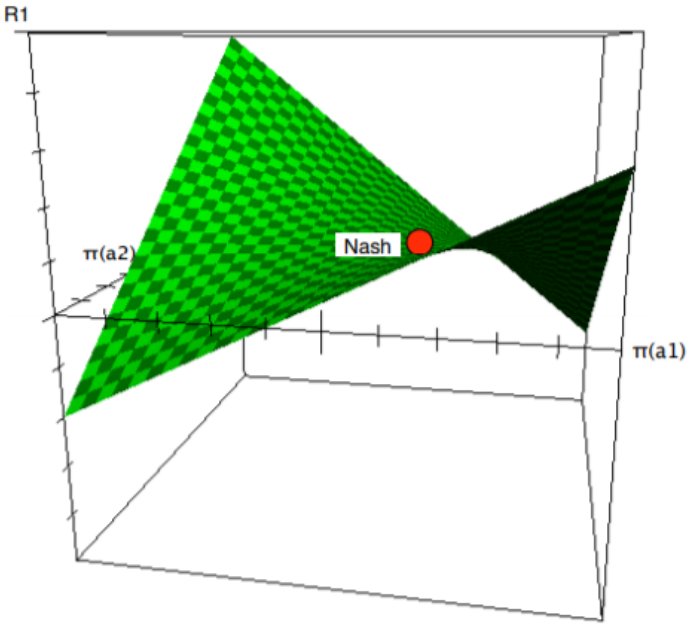
L'équilibre de Nash est une collection de politiques, une pour chaque agent, où chacune de ces politique est une best-response policy, ce qui signifie qu'aucun de ces agents à intérêt de changer de stratégie si les autres gardent les leurs.

$$\forall i \in \{1, \ldots, n\}, \pi_i \in \mathcal{BR}_i(\pi_{-i})$$

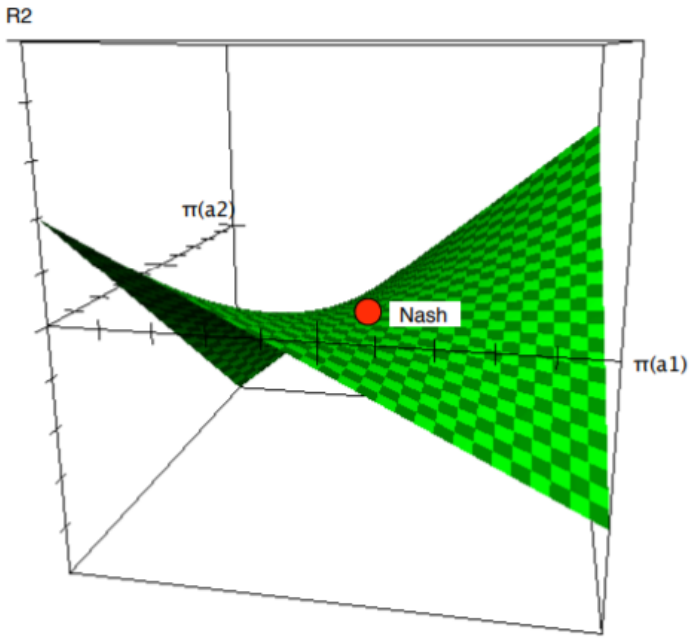
Remarque

Dans un jeu de matrice il existe au moins un équilibre de Nash.

Equilibre de Nash dans un jeu à somme nulle



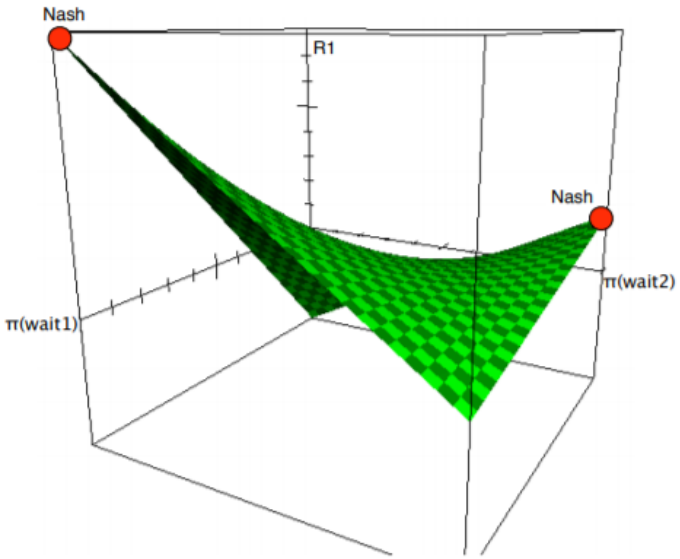
(a) Reward function for player 1



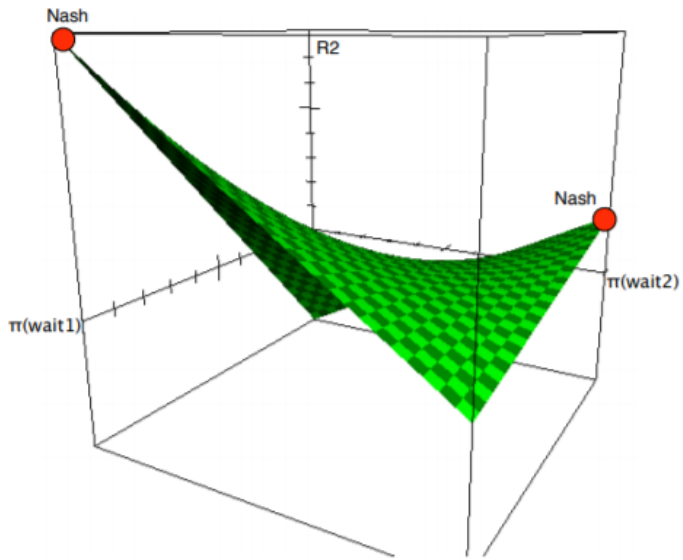
(b) Reward function for player 2

- Les récompenses sont symétriques par rapport au plan horizontal $R = 0$
- Il peut y avoir plusieurs points d'équilibres
- L'équilibre de Nash correspond au pire scénario
- L'équilibre de Nash est un point selle

Equilibre de Nash dans un jeu d'équipe



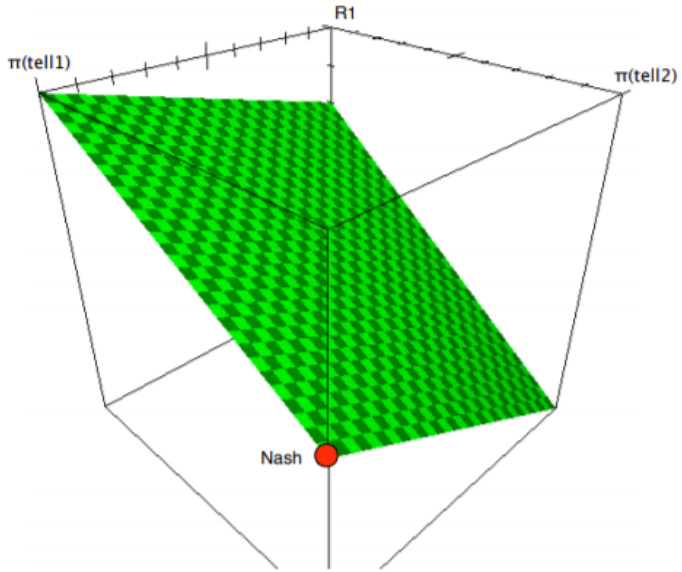
(a) Reward function for player 1



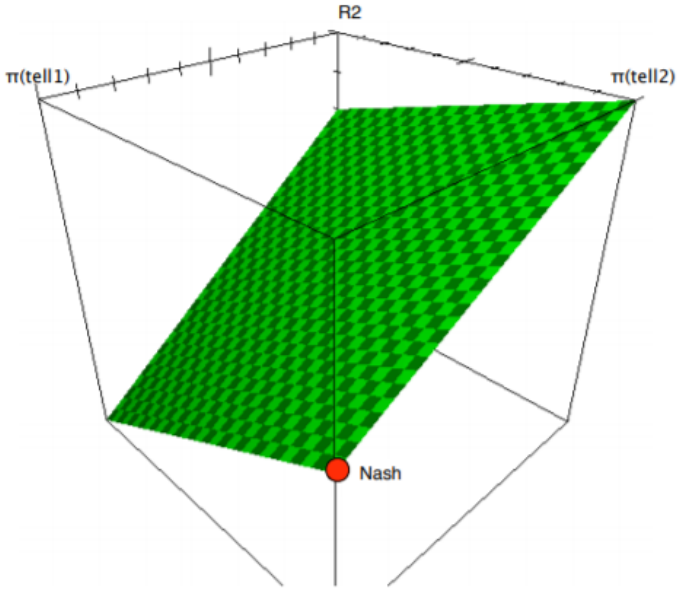
(b) Reward function for player 2

- Les points d'équilibres sont des les maximums locaux de la récompense commune.
- Politique gloutonne (comme les MDP).

Equilibre de Nash dans un jeu à somme générale



(a) Reward function for player 1



(b) Reward function for player 1

- Les points d'équilibres sont plus difficile à trouver
- Pour $n = 2$, les points d'équilibres sont les solution d'un problème d'optimisation quadratique.

Politiques optimales des jeux stochastiques

- Contre une politique stationnaire $\rightsquigarrow$ MDP
- Contre une politique non-stationnaire:
  - Best-response Learners
  - Equilibrium Learners

Best-response Learners

Définition

L'agent essaye d'apprendre une politique *d'attaque* optimale face à celles de ses adversaires

+ Si les adversaires adoptent des politiques d'équilibre, l'agent peut obtenir des meilleurs récompenses que celles que l'équilibres propose.
– Si les adversaires n'ont pas une politique stationnaire ou ne convergent pas vers une, l'agent risque de ne pas savoir s'adapter à ses adversaires.

Exemples:

- Joint-action learners
- Opponent modeling

Equilibrium Learners

Définition

L'agent essaye d'apprendre une politique optimale qui converge vers un équilibre de Nash.

\+ La politique obtenue est une politique de défense contre les pires adversaires.

– La politique peut être battue par un adversaire adapté.

Exemples:

- Minimax-Q
- Nash-Q
- Mean-Field MARL

Mean Field Multi-Agent Reinforcement Learning

Problématique

Problème

- *Recherche d'une politique d'équilibre pour une très grande population d'agents*
- *Approximation des interactions des agents (amis-ennemies)*

Proposition

Les interactions dans la population d'agents peuvent être considérées comme celle entre un agent simple et l'effet moyen de la population total ou les agents voisins [Yang et al., 2018].

Factorisation

La fonction Q est exprimée est fonction par ses interactions locales deux-à-deux.

$$Q_i(s,\mathbf{a}) = \frac{1}{n_i} \sum_{k \in \mathcal{N}(i)} Q_i(s,a_i,a_k)$$

où $\mathcal{N}(i)$ est l'ensemble d'agents voisin de l'agent i de cardinalité $n_i = |\mathcal{N}(i)|$.

Approximation Mean-Field

Les interactions des $Q_i(s,a_i,a_k)$ peuvent être approchées à l'aide de la théorie Mean Field (théorie du champ moléculaire).

- L'action *moyenne* $\bar{a}_i$ se calcule $\bar{a}_i = \frac{1}{n_i} \sum_{k \in \mathcal{N}(i)} a_k$
- L'action a_k s'exprime alors $a_k = \bar{a}_i + \delta a_{i,k}$ où $\delta a_{i,k}$ est une petite fluctuation.
- La fonction Q peut être approchée par un développement de Taylor

$$Q_i(s,\mathbf{a}) = \frac{1}{n_i} \sum_{k \in \mathcal{N}(i)} Q_i(s,a_i,a_k) \approx Q_i(s,a_i,\bar{a}_i)$$

- L'apprentissage de la fonction Q est donc défini par:

$$Q_i(s,a_i,\bar{a}_i) \leftarrow (1-\alpha) Q_i(s,a_i,\bar{a}_i) + \alpha \left(r_i + \gamma V_i(s')\right)$$

- Le problème MARL se réduit à la recherche de politique optimale de l'agent π_i par rapport à l'action moyenne $\bar{a}_i$ et de ses voisins.

Algorithme Mean-Field

Une méthode itérative est proposé par [Yang et al., 2018]

1. Calculer l'action moyenne $\bar{a}_i = \frac{1}{n_i} \sum_{k \in \mathcal{N}(i)} a_k$

   Les actions a_k sont déterminées par les politiques π_k et leurs actions moyennes précédentes $\bar{a}_k$.
2. Calculer la politique *Boltzmann* par la formule

$$\pi_i(a_i|s,\bar{a}_i) = \frac{\exp\left(-\beta Q_i(s,a_i,\bar{a}_i)\right)}{\sum_{a'_i \in A_i} \exp\left(-\beta Q_i(s,a'_i,\bar{a}'_i)\right)}$$

L'action moyenne $\bar{a}_i$ converge vers un seul point d'équilibre $\Rightarrow$ la politique π_i converge [Yang et al., 2018]

Apprentissage de l'agent

Calcul de la fonction Q par réseaux de neurones artificielles: paramétrisation par des points ϕ_i.

- **Méthode MF-Q:** Minimiser la fonction de perte

$$\mathcal{L}(\phi_i) = (y_i - Q_{\phi_i}(s, a_i, \bar{a}_i))^2 \quad \text{avec } y_i = r_i + \gamma V_{\phi_i}(s')$$

- **Méthode MF-AC:** Méthode Actor-Critic non-étudiée

Exemple d'application: Jeu de bataille

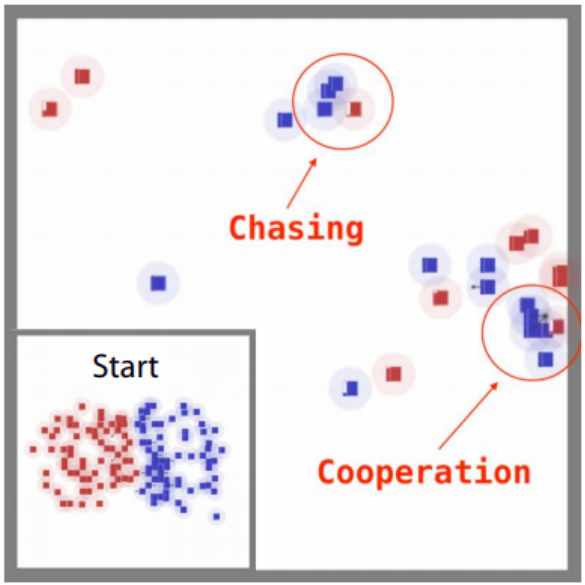
Environnement: plateforme open-source MAgent [Zheng et al., 2017]

- Cas multi-agent *cooperatif-compétitif*
- Deux armées se battent sur une grille $2D$
- Chaque armée est entraînée avec un algorithme RL différent
- L'armée consiste de 64 agents

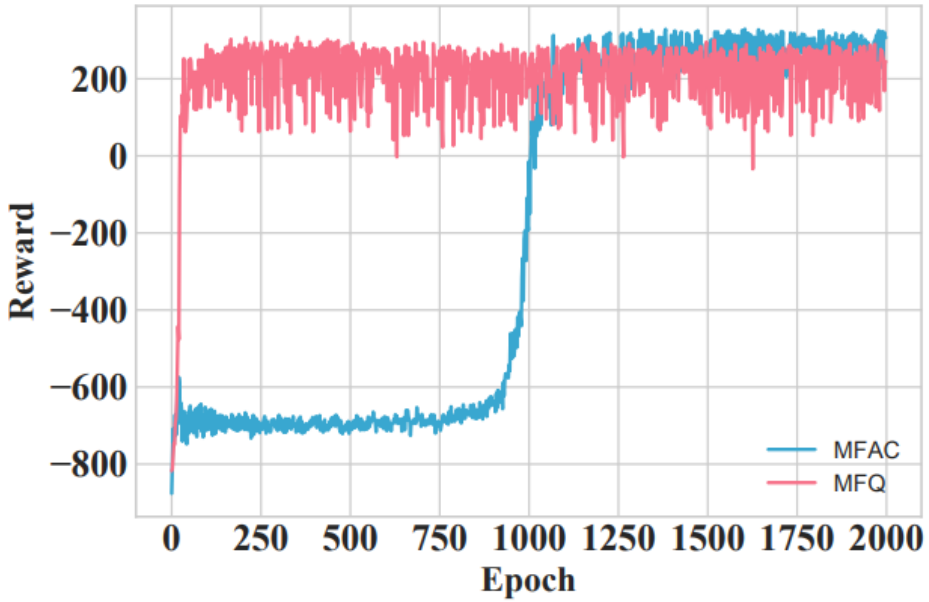
Les récompenses du jeu sont:

- un mouvement: -0.005
- attaque contre un ennemie: 0.2
- tuer un ennemie: 5
- attaque contre une case vide: -0.1
- se faire attaquer/tuer: -0.1

Résultats I

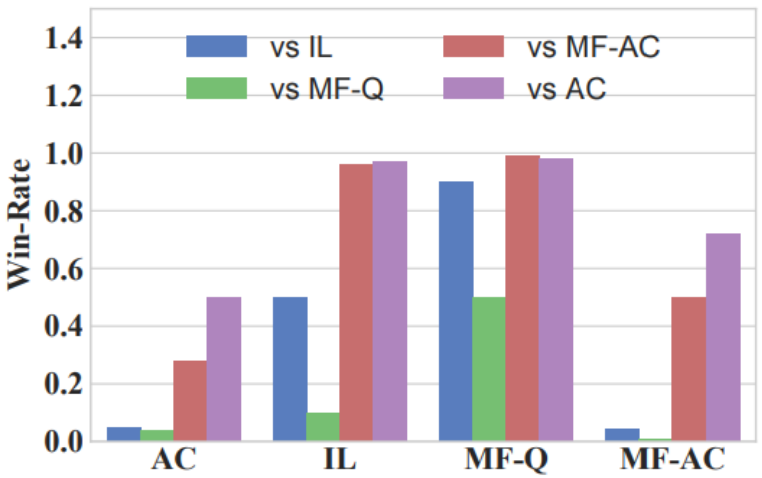


(a) Battle game scene.

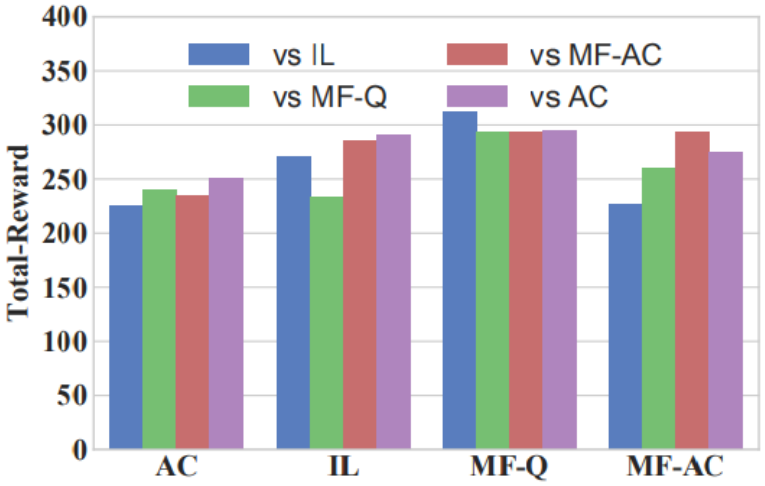


(b) Learning curve.

Résultats II



(a) Average wining rate.



(b) Average total reward.

Merci de votre attention!

Références I

Bellman, R. (1957).

Dynamic programming.

Princeton University Press, Princeton, NJ, USA, 1 edition.

tex.bib2html_rescat: General RL.

Littman, M. L. (1994).

Markov games as a framework for multi-agent reinforcement learning.

In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning*, ICML'94, pages 157–163, New Brunswick, NJ, USA. Morgan Kaufmann Publishers Inc.

Références II


Neto, G. (2005).
From Single-Agent to Multi-Agent Reinforcement Learning: Foundational Concepts and Methods.
Instituto de Sistemas e Robótica, Instituto Superior Técnico.


Shapley, L. (1953).
Stochastic Games.
Proceedings of the National Academy of Sciences of the United States of America.


Watkins, C. J. C. H. (1989).
Learning from delayed rewards.
PhD thesis, King's College, Cambridge, UK.
tex.bib2html_rescat: Parameter.

Références III


Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., and Wang, J. (2018).
Mean Field Multi-Agent Reinforcement Learning.
arXiv:1802.05438 [cs].
arXiv: 1802.05438.


Zheng, L., Yang, J., Cai, H., Zhang, W., Wang, J., and Yu, Y. (2017).
MAgent: A Many-Agent Reinforcement Learning Platform for Artificial Collective Intelligence.