Un algorithme Décentralisé d'Apprentissage par Renforcement Multi-Agents Coopératifs: le Q-Learning Hystérétique.

Laëtitia Matignon, Guillaume J. Laurent et Nadine Le Fort-Piat

Laboratoire d'Automatique de Besançon UMR CNRS 6596
24, rue Alain Savary, 25000 Besançon
{laetitia.matignon, guillaume.laurent,
nadine.piat}@ens2m.fr

Résumé : Nous nous intéressons aux techniques d'apprentissage par renforcement dans les systèmes multi-agents coopératifs. Nous présentons un nouvel algorithme pour agents indépendants qui permet d'apprendre l'action jointe optimale dans des jeux où la coordination est difficile. Nous motivons notre approche par le caractère décentralisé de cet algorithme qui ne nécessite aucune communication entre agents et des tables Q de taille indépendante du nombre d'agents. Des tests concluants sont de plus effectués sur des jeux coopératifs répétés, ainsi que sur un jeu de poursuite.

Mots-clés : Apprentissage par renforcement multi-agents, jeux matriciels répétés, DEC-POMPD, Q-Learning.

1 Introduction

Ces dernières années ont vu se développer un intérêt grandissant pour l'extension de l'apprentissage par renforcement (AR) aux systèmes multi-agents (SMA) (Busoniu *et al.*, 2006). La principale limitation réside dans le caractère non-markovien des SMA. Nous nous intéressons ici à des *agents apprenant indépendants* (AAI) - définis par (Claus & Boutilier, 1998) comme des agents n'ayant accès qu'à leurs propres actions - placés dans le cadre de *SMA coopératifs*, *i.e.* que la fonction de renforcement est la même pour tous les agents. L'objectif est de maximiser le renforcement commun. Dans le cas d'AAI, il faut s'assurer que tous les agents vont individuellement choisir leur action de sorte que l'action jointe résultante soit optimale : c'est le problème de la *coordination*.
Dans le cas d'une approche centralisée, le processus central doit disposer de l'ensemble des informations de renforcements, d'états et d'actions. Cette approche se heurte donc au problème d'*explosion combinatoire* du nombre d'états et d'actions jointes à considérer en fonction du nombre d'agents. Une approche décentralisée permet par contre d'appréhender des problèmes avec un nombre d'agents élevé sans explosion combina-

toire puisque chaque stratégie individuelle est construite à partir de Q-valeurs individuelles. Ainsi, si n est le nombre d'agents, S l'ensemble des états, A_i l'ensemble des actions pour un agent, le nombre de Q-valeurs à actualiser dans le cas d'un apprentissage centralisé est $|S| \times |A_1| ... \times |A_n|$, alors qu'il est de $n \times |S| \times |A_i|$ dans le cas décentralisé. Beaucoup de travaux s'orientent donc vers cette dernière approche. Ainsi, malgré ses limitations théoriques dans les SMA, le Q-Learning décentralisé a déjà été appliqué avec succès (Crites & Barto, 1998; Wang & de Silva, 2006).
Plusieurs algorithmes dérivés du Q-Learning sont aussi proposés pour résoudre la coordination entre AAI. Lauer & Riedmiller (2000) présentent le *Distributed Q-Learning* qui ignore dans sa mise à jour les pénalités dues à une non-coordination des agents. La convergence vers l'unique action jointe optimale est validée dans le cas de SMA coopératifs déterministes. La méthode *Frequency Maximum Q-value* (FMQ) (Kapetanakis & Kudenko, 2004) enregistre la fréquence d'occurrence de la récompense maximale obtenue pour chaque action. Cette fréquence influence ensuite la sélection d'action. Cet algorithme, proposé uniquement dans le cadre des jeux répétés, règle le problème de l'incertitude sur les récompenses due aux actions des autres, mais ne surmonte pas la difficulté des jeux fortement bruités.

Nous proposons dans cet article un algorithme pour AAI coopératifs, appelé *Q-Learning hystérétique*. Nous étudions tout d'abord le cas des jeux matriciels coopératifs répétés (section 2) puis celui des DEC-POMDPs (section 3).

2 Jeux matriciels coopératifs répétés

2.1 Définition

Un jeu matriciel[1] est un tuple $< n, A_i, R_i >$ où
- n est le nombre d'agents
- A_i est l'ensemble des actions pour l'agent i ($A = \prod A_i$ l'ensemble des actions jointes)
- $R_i : A \rightarrow \Re$ est la fonction de renforcement de l'agent i qui dépend de l'action conjointe des agents.

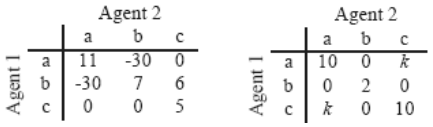
Si $R_1 = ... = R_n$, le jeu matriciel est dit *coopératif*. Les jeux Fig. 1 en sont des exemples. Nous nous intéressons plus particulièrement aux *jeux répétés* qui consistent en la répétition d'un même jeu par les mêmes agents. Claus & Boutilier (1998) proposent une reformulation de l'algorithme du Q-Learning pour ces jeux. Chaque agent i maintient une estimation Q_i de l'utilité de chaque action a_i :

$$Q_i(a_i) \leftarrow Q_i(a_i) + \alpha(r - Q_i(a_i)) \quad (1)$$

2.2 Q-Learning Hystérétique pour des jeux répétés

Dans un SMA, le renforcement perçu par un agent dépend des actions choisies par le groupe ; donc un agent effectuant une commande optimale peut tout de même être puni

[1] appelé aussi *jeu stratégique* dans la littérature



Climbing game Penalty game

FIG. 1 – Jeux matriciels coopératifs.

à cause d'un mauvais choix du groupe. Il est donc préférable pour un agent d'accorder peu d'importance à une punition reçue après avoir choisi une action lui ayant amené satisfaction dans le passé. Néanmoins, l'agent ne doit pas être totalement aveugle face aux sanctions au risque de rester dans des équilibres sous-optimaux ou de ne pas se coordonner sur une action jointe optimale. L'idée du Q-Learning hystérétique est d'utiliser deux coefficients d'apprentissage selon le résultat d'une action jointe. Cette idée a été appliquée à des méthodes de montée de gradient par (Bowling & Veloso, 2002).
Nous nous intéressons ici à l'utilisation de deux coefficients d'apprentissage dans le Q-Learning. Ainsi, l'équation 1 de mise à jour est modifiée :

$$\begin{aligned} d &\leftarrow r - Q_i(a_i) \\ Q_i(a_i) &\leftarrow \begin{cases} Q_i(a_i) + \alpha d & \text{si } d \geq 0 \\ Q_i(a_i) + \beta d & \text{sinon} \end{cases} \end{aligned} \tag{2}$$

où α et β sont, *resp.*, les coefficients de montée et de descente des valeurs de Q. Le Q-Learning hystérétique est décentralisé ; chaque AAI construit sa propre table Q dont la taille est indépendante du nombre d'agents et linéaire en fonction de ses propres commandes.

2.3 Expérimentations

Les performances du Q-Learning hystérétique sont évaluées dans des jeux matriciels coopératifs répétés à deux agents (Fig. 1), dont les difficultés résident dans de fortes pénalités dans le cas de non-coordination et des actions jointes optimales multiples (Claus & Boutilier, 1998). Nous testons différents algorithmes : Q-Learning décentralisé, FMQ, Distributed Q-Learning et Q-Learning hystérétique (Tab. 1). Un épisode consiste en 7500 choix successifs des actions par les deux agents. Au début d'un épisode, les tables Q des agents sont initialisées à 0. A la fin d'un épisode, on regarde si la dernière action jointe est optimale.
Nous prenons $\alpha = 0.1$, $\beta = 0.01$ et $T = T \times 0.99$ avec $T_{init} = 5000$[2]. Tout d'abord, le Q-Learning est inefficace à atteindre l'action jointe optimale. Sur le *Climbing game*, les trois autres algorithmes surpassent la difficulté des fortes pénalités associées à une non-coordination de leurs actions. Notamment les meilleurs résultats son obtenus avec les algorithmes *FMQ* et *Q-Learning Hystérétique*. Par contre, dans le cas de plusieurs

[2]choix de l'action selon une distribution de Bolztmann où T est un paramètre dit de température

TAB. 1 – Pourcentage d'épisodes qui convergent vers l'action jointe optimale (pourcentages calculés sur 3000 épisodes).

	Climbing game	Penalty game ($k = -100$)
Q-Learning décentralisé	12.1%	64%
Distributed Q-Learning	95.8%	50.6%
FMQ	99.8%	99.9%
Q-Learning hystérétique	99.5%	99.8%

actions jointes optimales (*Penalty Game*), les agents *Distributed Q-Learning* ne se coordonnent pas à tous les coups sur la même politique conjointe optimale. La coordination sur une des actions jointes optimales est réalisée avec les algorithmes *FMQ* et *Q-Learning hystérétique*.

3 DEC-POMDP

3.1 Définition

Ce formalisme permet de représenter le problème de prise de décision collective avec des observabilités partielles.
Un DEC-POMDP (Bernstein *et al.*, 2002) est un tuple $< n, S, A_i, T, \Gamma_i, O, R >$ où

- n désigne le nombre d'agents dans le système
- S l'ensemble des états possibles du monde
- $A = \prod A_i$ l'espace des actions jointes
- $T : S \times A \times S \rightarrow [0, 1]$ la matrice de transition
- Γ_i l'ensemble des observations possibles pour l'agent i ($\Gamma = \prod \Gamma_i$)
- $O : S \times A \times \Gamma \rightarrow [0, 1]$ définit les probabilités d'observations
- $R : S \times A \rightarrow \Re$ la fonction de renforcement

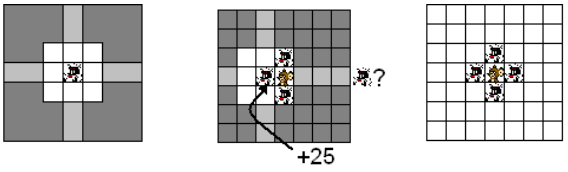
3.2 Q-Learning Hystérétique à plusieurs états

L'équation de mise à jour du Q-Learning hystérétique pour un agent i ayant reçu la récompense r pour la transition de l'observation s à la nouvelle observation s' après exécution de l'action a_i est :

$$d \leftarrow r + \gamma \max_{a'} Q_i(s', a') - Q_i(s, a_i)$$
$$Q_i(s, a_i) \leftarrow \begin{cases} Q_i(s, a_i) + \alpha d & \text{si } d \geq 0 \\ Q_i(s, a_i) + \beta d & \text{sinon} \end{cases} \quad (3)$$

3.3 Expérimentations

Dans cette section, nous testons notre algorithme sur un jeu de poursuite (Buffet, 2003). Ce jeu consiste en 4 agents placés dans un labyrinthe toroïdal 7×7 qui doivent



(a) Perceptions (b) Renforcement (c) Capture

FIG. 2 – a) $(2 \times 8)^4$ états par agent. b) Le renforcement est attribué de manière individuelle et ne dépend que des perceptions locales. c) Proie capturée

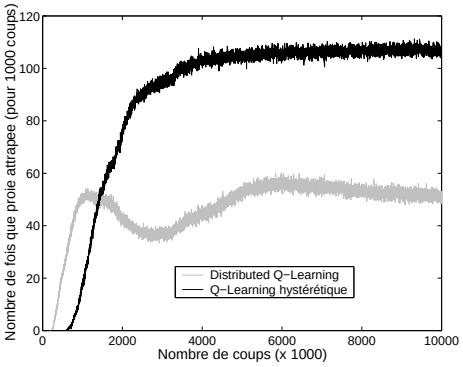


FIG. 3 – Nombre de captures pour 1000 coups (moyenne sur 20 essais) avec $\alpha = 1$ $\beta = 0.1$ $\gamma = 0.9$ $\epsilon = 0.1$.

se coordonner afin d'attraper une proie (Fig. 2c). Chaque agent peut se déplacer selon 4 directions ou rester sur place ; de même pour la proie qui se déplace de manière aléatoire avec une vitesse inférieure à celle des agents. Chaque agent perçoit ses congénères et la proie selon les 8 directions cardinales ainsi que selon un critère proche ou lointain (Fig. 2a). Le Q-Learning hystérétique et le Distributed Q-Learning utilisent ici une table Q de dimension $16^4 \times 5$ par agent. Des tests ont été effectués avec des AAI apprenant avec l'algorithme du Q-Learning et montrent que même après 10.10^6 coups, les agents n'arrivent pas à apprendre une politique optimale. Par contre, avec l'algorithme du *Q-Learning hystérétique*, les agents apprennent à se coordonner pour encercler la proie beaucoup plus souvent qu'avec l'algorithme du *Distributed Q-Learning* (Fig. 3). L'apprentissage de la coordination se stabilise autour de 5.10^6 coups.

4 Conclusion

Nous détaillons dans cet article un algorithme décentralisé basé sur le Q-Learning et destiné à l'AR dans les *systèmes multi-agents coopératifs pour des agents apprenant*

Début
Initialiser $Q_i(s, a_i)$ à 0 pour tout a_i de A_i et s de S.
Tant que s n'est pas l'état absorbant **faire**
Choisir une action a_i selon une méthode de décision
Exécuter l'action a_i et observer la récompense r et le nouvel état s
$d \leftarrow r - Q_i(s, a_i)$
Si $d \geq 0$ *alors* $Q_i(s, a_i) \leftarrow Q_i(s, a_i) + \alpha d$
sinon $Q_i(s, a_i) \leftarrow Q_i(s, a_i) + \beta d$
Fin tant que
Fin

FIG. 4 – Algorithme du Q-Learning Hystérétique pour un agent i.

indépendants (Fig. 4). Grâce à deux coefficients d'apprentissage, cet algorithme donne la possibilité d'influencer différemment les vitesses de croissance et de décroissance des valeurs de Q. Il permet notamment aux agents de se coordonner sur la même action jointe optimale. Comparé à d'autres algorithmes décentralisés ne nécessitant pas de communication, il est aussi performant pour une taille mémoire inférieure. En effet, seule la table Q est nécessaire (contrairement à l'algorithme du FMQ par exemple). De plus, sur un jeu de poursuite, la coordination des agents est effective. Néanmoins, des travaux sont à poursuivre, notamment dans le cadre des SMA à récompenses stochastiques, où la difficulté fondamentale est de distinguer si l'incertitude sur les récompenses provient des actions des autres ou du bruit.

Références

BERNSTEIN D. S., GIVAN R., IMMERMAN N. & ZILBERSTEIN S. (2002). The complexity of decentralized control of markov decision processes. *Math. Oper. Res.*, **27**(4), 819–840.

BOWLING M. & VELOSO M. (2002). Multiagent learning using a variable learning rate. *Artificial Intelligence*, **136**, 215–250.

BUFFET O. (2003). *Une double approche modulaire de l'apprentissage par renforcement pour des agents intelligents adaptatifs.* PhD thesis, Université Henri Poincaré, Nancy 1, France. paper.

BUSONIU L., BABUSKA R. & SCHUTTER B. D. (2006). Multi-agent reinforcement learning : A survey. In *Int. Conf. Control, Automation, Robotics and Vision*, p. 527–532.

CLAUS C. & BOUTILIER C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, p. 746–752.

CRITES R. H. & BARTO A. G. (1998). Elevator group control using multiple reinforcement learning agents. *Machine Learning*, **33**(2-3), 235–262.

KAPETANAKIS S. & KUDENKO D. (2004). Reinforcement learning of coordination in heterogeneous cooperative multi-agent systems. In *Proc. of AAMAS '04*, p. 1258–1259, Washington, DC, USA : IEEE Computer Society.

LAUER M. & RIEDMILLER M. (2000). An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proc. 17th International Conf. on Machine Learning*, p. 535–542 : Morgan Kaufmann, San Francisco, CA.

WANG Y. & DE SILVA C. W. (2006). Multi-robot box-pushing : Single-agent q-learning vs. team q-learning. In *Proc. of IROS*, p. 3694–3699.