Apprentissage par renforcement pour la conception de systèmes multi-agents réactifs

Alain Dutech — Olivier Buffet — François Charpillet

LORIA - MAIA
BP 239 - F-54506 Vandoeuvre les Nancy
{dutech,buffet,charp}@loria.fr

RÉSUMÉ. *Nous proposons une nouvelle méthodologie d'apprentissage par renforcement (AR) pour la conception de systèmes multi-agents réactifs. Bien que le cadre réaliste d'agents situés avec des perceptions locales sorte du cadre théorique de convergence des algorithmes classiques d'apprentissage par renforcement, notre méthode permet à chaque agent d'apprendre individuellement et localement son comportement. L'aspect progressif de notre algorithme qui met les agents en présence de sous-tâches de plus en plus complexes permet de dépasser les limitations classiques de l'AR dans ce contexte. Notre méthodologie, qui se veut générale, est validée en simulation sur un problème où les agents doivent se coordonner pour atteindre un but global.*

ABSTRACT. *A new reinforcement learning (RL) methodology for the design of reactive multi-agent systems is presented. Although dealing with realistic situated agents with local perception does not belong to the framework where convergence of RL algorithm is guaranted, in our method each agent learns individually its local behavior. The progressive aspect of learning, which pits the agents against more and more complex sub-tasks, allows to go beyond the classical limitation of RL in this context. Our general framework is validated on a simulated environment where the agents have to coordinate themselves to reach a global goal.*

MOTS-CLÉS : *processus décisionnels de Markov, agents réactifs, apprentissage par renforcement, apprentissage progressif, coordination, coopération*

KEYWORDS: *Markov decision processes, reactive agents, reinforcement learning, shaping, coordination, cooperation*

1. Introduction

Le problème de la conception automatique d'un système multi-agents (SMA) est au cœur de nombreuses recherches dans le domaine. C'est un enjeu important qui renvoie en particulier à une question cruciale : comment faire le lien entre une description *globale* d'une tâche et des agents dont les comportements dépendent d'une vue qui est forcément *partielle* et *locale* de cette même tâche.

Ce problème de conception est encore plus délicat quand on s'intéresse aux SMA "réactifs" et "coopératifs" où les agents qui doivent coopérer sont dotés de comportements simples et le plus souvent réactifs. Ces systèmes s'appuient sur les interactions entres les agents, souvent présents en grand nombre, pour produire un comportement collectif complexe en s'appuyant notamment sur des phénomènes d'auto-organisation. Le principal avantage de ces systèmes est la simplicité des agents, ce qui rend leur conception plus aisée. Le gros point noir de ce domaine reste le réglage et la mise au point des différents paramètres des comportements réactifs pour produire le comportement collectif voulu.

Pour automatiser le processus de conception de SMA réactif, nous proposons d'utiliser des algorithmes d'*apprentissage par renforcement* (AR), et ce de manière décentralisée. Chaque agent va ainsi apprendre localement son comportement de manière à optimiser une performance globale. Les avantages de cette approche sont nombreux :

– d'une part, un simple signal scalaire évaluant le comportement du système (la récompense) est suffisant pour apprendre. Il n'est pas nécessaire de disposer d'un professeur connaissant à l'avance la solution au problème. Il ne s'agit pas d'apprentissage supervisé.

– d'autre part, comme chaque agent d'un SMA réactif est confronté à une tâche assez simple, il est plus facile pour chaque agent d'apprendre son comportement réactif localement que d'essayer d'apprendre globalement un comportement collectif du système. On évite ainsi un écueil important de l'AR : une explosion combinatoire d'autant plus probable que le problème à résoudre est plus complexe.

– enfin, les méthodes d'apprentissage par renforcement s'appuient sur le formalisme mathématique des *processus décisionnels de Markov* (PDM) qui précise les conditions de convergence des algorithmes. Ce cadre mathématique permet aussi de proposer un formalisme pour l'étude du comportement collectif du SMA.

En fait, l'utilisation de l'apprentissage par renforcement décentralisé dans le cadre des SMA pose quelques problèmes délicats. En particulier, un agent n'ayant qu'une vue locale et partielle de son environnement, il ne connaît généralement pas l'état global du système dans lequel il se trouve. Dès lors, l'apprentissage ne se fait plus dans le cadre formel des PDM garantissant une convergence des algorithmes vers un comportement optimal.

Notre proposition est alors de faire un apprentissage par renforcement **décentralisé progressif** qui, s'appuyant sur des techniques classiques d'AR, permet de doter les agents de comportements réactifs stochastiques. En effet, comme le montre [SIN 94], des agents avec des comportements stochastiques seront plus performants dans ce cadre de perceptions partielles du système global. L'aspect progressif de notre

algorithme se fera selon deux grands axes, d'une part en confrontant les agents à des tâches de complexité croissante, d'autre part en augmentant progressivement le nombre d'agents.

La première partie de l'article (Section 2) présente notre problématique. L'algorithme décrit en Section 3 est testé expérimentalement (cf. Section 4) avant d'être discuté et mis en relation avec des travaux similaire en Section 5. Nous concluons en Section 6.

2. Problématique

2.1. *Conception de Systèmes Multi-Agents*

Les systèmes multi-agents coopératifs dont nous voulons automatiser la conception sont composés d'agents qui apprennent individuellement leur propre comportement. Ces agents sont des agents réactifs très simples, ce qui permet de conserver une complexité abordable à la tâche d'apprentissage individuelle.

Les caractéristiques principales des agents que nous allons utiliser sont :

– **réactifs** : les agents ont des comportements "réflexes.

– **situés avec des perceptions partielles** : bien que l'aspect "partiel" des observations soit un handicap pour l'apprentissage (cf.Section 2.3), leur aspect "local" permet de diminuer les risques d'explosion combinatoire car peu d'éléments sont perçus.

– **potentiellement hétérogènes** : bien que disposant des mêmes capacités d'action et de perception, chaque agent apprend individuellement. Ainsi, chaque agent peut acquérir un comportement différent des autres.

– **coopératifs** : tous les agents partagent le même but et *devront* se coordonner pour y parvenir.

La question qui est alors posée est la suivante : étant donnée une tâche globale (c'est-à-dire qui nécessite a priori une vision globale du système pour être résolue), comment concevoir les comportements individuels des agents n'ayant qu'une vue locale et partielle de leur environnement ? Pour les raisons que nous avons rappelée en introduction (cadre formel, approche descendante sans supervision), nous proposons d'approcher cette problématique par le biais de l'apprentissage par renforcement.

2.2. *Apprentissage par Renforcement*

Commençons par détailler le cadre théorique *idéal* de l'apprentissage par renforcement, à savoir les processus décisionnels de Markov (PDM) (cf. [PUT 94]).

Soit $(\mathcal{S}, \mathcal{A}, T, r)$ un **P**rocessus **D**écisionnel de **M**arkov (noté PDM) où $\mathcal{S}$ est un ensemble fini d'états, $\mathcal{A}$ un ensemble fini d'actions. Ce processus est markovien car les transitions d'états sont gouvernées par la fonction de transition $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \longrightarrow$

$[0,1]$ avec $P\{S_{t+1} = s' | A_t = a, S_t = s\} = T(s,a,s')$. r est une fonction de $\mathcal{S} \times \mathcal{A}$ dans $\mathbb{R}$ qui indique la **récompense** reçue par le système après chaque transition d'état. Une **politique** d'action π est une fonction des états vers les actions. Le problème est alors de trouver une politique d'action qui optimise une certaine fonction de la récompense, qu'on appelle l'**utilité** et que l'on note V. Typiquement, l'utilité peut être la somme des récompenses sur un horizon fini, ou la somme pondérée de cette récompense sur un horizon infini ($\sum_{t=0}^{\infty} \gamma^t r_t$ où $\gamma \in [0,1[$) ou encore la récompense reçue en moyenne à chaque transition.

On parle d'apprentissage par renforcement quand on cherche à trouver la politique optimale du PDM sans en connaître le modèle (T et r), voir [SUT 98]. Le principe général de ces méthodes est de faire évoluer l'agent apprenant dans son environnement et d'y récolter des expériences (dans l'état s_t, l'action a_t conduit à l'état s_{t+1} et à la récompense r_t). Par des méthodes d'approximation stochastiques, on peut alors apprendre une fonction de valeur de chacun des états et en déduire une politique optimale (*Q-Learning* de [WAT 89]) ou utiliser des méthodes de descente de gradient pour apprendre directement une politique optimale (cf.[BAX 99]).

2.3. *Apprentissage par Renforcement et SMA*

Ainsi que le montrent les travaux de Boutilier [BOU 96], un SMA se modélise facilement sous la forme d'un PDM où l'état du système dépend de *tous* les agents et où une action est la composée des actions de *tous* les agents. Cette approche, outre sa complexité exponentielle en nombre d'agent (et donc rapidement irréaliste), n'est pas en accord avec l'aspect décentralisé des systèmes multi-agents.

Si l'on veut résoudre le problème de manière décentralisée, où chaque agent n'a qu'une vue partielle de l'état du système, on se place alors dans le cadre de ce que Bernstein *et al.* [BER 00] appellent les Processus Décisionnel de Markov Partiellement Observés Décentralisés, qui n'admettent pas d'algorithme exact de résolution en temps polynômial. Les principales difficultés qui rendent les méthodes classiques inefficaces dans ce cadre sont :

– **des transitions non-stationnaires** : Du point de vue d'un agent, les autres agents sont des éléments imprévisibles de l'environnement, qui peuvent eux-mêmes apprendre. Dès lors, la fonction de transition, du point de vue d'un agent, est non-stationnaire : le résultat de son action dépend des autres agents, et eux-même évoluent avec le temps.

– **des perceptions partielles** : un agent ne peut percevoir l'état global du système, ce qui est pourtant une hypothèse nécessaire pour assurer la convergence des algorithmes classiques vers un comportement optimal. Même quand les transitions sont stationnaires, les méthodes actuelles sont très limitées (systèmes composés de quelques dizaines de situations seulement), voir par exemple [DUT 00].

3. Apprentissage progressif par renforcement

Puisqu'il n'existe pas d'algorithmes exacts permettant de trouver des comportements optimum pour les agents que nous considérons, nous allons utiliser des méthodes approchées. Notre but est de trouver les meilleurs comportements réactifs et stochastiques possibles pour les agents. Pour cela, nous avons utilisé un algorithme d'apprentissage par renforcement dérivé des travaux de [BAX 99] qui permet d'optimiser par une descente de gradient des politiques stochastiques parametrées.

Pour accélérer l'apprentissage et améliorer l'efficacité des comportements obtenus en évitant principalement que l'apprentissage ne s'arrête au premier optimum local rencontré, nous proposons une méthodologie d'apprentissage progressif. Cette approche, s'inspirant des travaux sur le "shaping" en psychologie, a été étudiée en apprentissage par renforcement mono-agent, mais uniquement dans le cadre des PDM (voir [AND 99], [RAN 98]). L'idée principale de cet approche est de guider l'apprentissage en plaçant les agents dans des situations proches des sources (positives ou négatives) de renforcement, puis dans des situations de plus en plus éloignée de ces sources de renforcement.

Cette approche accélère l'apprentissage puisque les agents reçoivent fréquemment des récompenses non-nulles qui leur permettent de modifier leur comportement. Cela devrait se vérifier aussi dans un cadre non-markovien. De plus, quand on se place dans un cadre multi-agents, le fait de contraindre les agents à des situations simples permet de les faire se coordonner plus rapidement, ce qui réduit sensiblement le caractère non-markovien des situations associées à de fortes récompense. Petit à petit, en "éloignant" les agents de ces situations particulières, l'apprentissage devrait être favorisé d'une part grâce à ce que les agents ont déjà appris, et d'autre part grâce à cette "réduction" du caractère non-markovien du problème pour les situations déjà connues.

Les études précédentes ont montré l'importance de la progression (cf. [ASA 96]), qui se traduit généralement par une séquence de situations auxquelles sont confrontés les agents. Ces situations doivent être de plus en plus éloignées des sources non-nulles de récompense. Le problème, dans le cadre des problèmes partiellement observable, c'est qu'on ne peut définir une telle fonction d'ordre sur les situations. Nous avons néanmoins définit la méthode de progression suivante, qui se fait suivant deux axes :

– **complexité de la tâche**. Les agents sont confrontés à des tâches de plus en plus complexes. Les premières tâches sont assez "proches" du but (en terme de récompense), puis de plus en plus éloignées. Au fur et à mesure de l'apprentissage, les agents disposent de plus en plus de liberté de mouvement et peuvent explorer plus avant leur environnement.

– **nombre d'agents**. C'est la méthode la plus intuitive. L'apprentissage commence avec un faible nombre d'agents, chacun apprenant son propre comportement. Ensuite, d'autres agents sont ajoutés, avec un comportement initial déduit des agents qui on déjà appris. Les comportements de tous les agents sont encore améliorés par apprentissage.

4. Validation expérimentale

4.1. *Fusion de cubes*

La tâche choisie met en œuvre des agents de deux types, jaunes ou bleus, dans un monde pavé plan. Leur but est de pousser des cubes jaunes contre des cubes bleus[1]. Quand deux agents coordonnent leurs mouvements pour atteindre ce but (pousser ensemble une paire de cubes), les deux cubes disparaissent temporairement pour réapparaître de manière aléatoire ailleurs sur le pavage. Simultanément, les agents responsables de cette "fusion" reçoivent alors une récompense positive. L'objectif des agents est ainsi de provoquer de telles fusions le plus souvent possible.

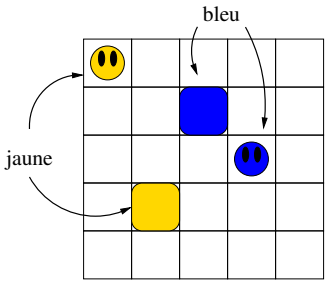
Description des agents

– **actions** : Les agents ne disposent que de quatre actions possibles, correspondant à des déplacements d'une case vers le nord, l'ouest, le sud et l'est (un agent cherche toujours à se déplacer). Les agents peuvent pousser d'autres agents et d'autres cubes, ce qui rend aléatoire les conséquences de leurs actions. En effet, quand plusieurs agents ont une influence sur le déplacement d'un cube (par exemple), une seule de ces influences - choisie au hasard - va avoir un réel effet.

– **perceptions** : Comme le montre la figure 1, les perceptions d'un agent sont constituées des informations suivantes :

- `dir(aco)` : La direction de l'agent de couleur opposée le plus proche parmi les quatre directions cardinales `(N-O-S-E)`.

- Pour le cube le plus proche de chaque couleur : la direction `dir(.)` `(N-NO-O-SO-S-SE-E-NE)` du cube et un drapeau `proche(.)` sur sa proximité (`vrai` si sur une case adjacente).



agent	jaune	bleu
dir(aco)	SE	NO
dir(cj)	S	O
dir(cb)	E	NO
proche(cj)	non	non
proche(cb)	non	oui

aco : agent de couleur opposée
cj : cube jaune
cb : cube bleu

Figure 1. *Exemples de perceptions (deux agents dans un monde simple)*

En combinant ces perceptions, on arrive à un maximum de 1024 observations. En tenant compte des situations impossibles et des symétries on peut ramener ce nombre à 256. C'est à comparer avec les 15.249.024 états du problème centralisé et complètement observé dans un environnement de taille 8×8, d'autant que cet ensemble d'observations est *indépendant de la taille du monde*.

1. La couleur des agents est ici sans lien avec celle des cubes.

– **récompenses** : Dans le cas d'une fusion de deux cube, chaque agent participant à la fusion reçoit une récompense positive ($+5$). Il s'agit bien ici, comme souhaité, d'une récompense dépendant du point de vue local de l'agent.

4.2. *Progression en complexité*

Si le problème de fusion de cubes présenté peut mettre en jeu un nombre a priori variable de cubes et d'agents, l'apprentissage progressif en complexité qui nous intéresse ne sera utilisé que sur un dispositif réduit au minimum utile : un agent et un cube de chaque couleur.

La table 4.2 montre une séquence d'expériences utilisées pour aider les agents dans leur entraînement. La première *situation initiale*, dans un monde 6×3, n'a besoin que d'un mouvement pour que le but soit atteint, chaque agent poussant dans la direction des cubes. Toutefois, ils ont en tout 6 pas de temps pour accomplir cette tâche, ce qui leur permet d'explorer différents mouvements et leurs conséquences. Un tel essai est ici répété 150 fois.

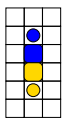
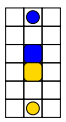
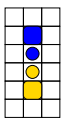
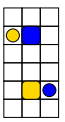
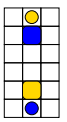
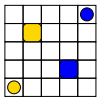
Situation initiale							
n (pas)	6	6	10	20	20	100	100
N (essais)	150	100	150	150	150	15	15

Tableau 1. *La séquence d'expériences utilisée pour l'apprentissage progressif*

Pour pouvoir juger de l'efficacité de l'approche adoptée, un apprentissage standard est comparé à l'apprentissage progressif étudié. Dans les deux cas, on compte le nombre de fusions de cubes effectuées pendant 1000 pas de temps pour évaluer la qualité des politiques adoptées par les agents, ceux-ci étant placés dans un environnement de taille 8×8 une fois l'éventuel entraînement terminé. Nous allons maintenant observer et analyser les résultats obtenus.

Résultats

On peut voir sur la figure 2 les évolutions des mesures de performance du groupe de deux agents avec et sans apprentissage progressif (moyenne sur 10 simulations indépendantes, et courbe lissée correspondante). La courbe montrant l'efficacité de l'apprentissage progressif ne commence qu'une fois passés les 12000 pas de temps (on parle aussi de "coups") que dure le script du tableau 4.2.

Dans les deux cas, les agents tendent vers des comportements leur permettant d'effectuer près de 90 fusions sur 1000 pas de temps. Par contre, une fois le délai de l'entraînement passé, la convergence vers de très bonnes performances se fait notablement plus vite que par un apprentissage "standard". Si les agents n'ont pas tout appris via

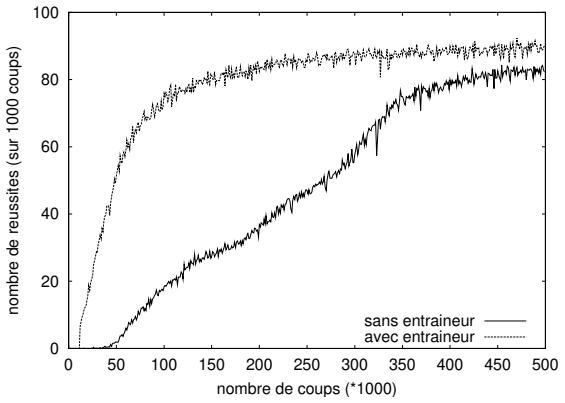


Figure 2. *2 agents, 2 cubes : apprentissage standard vs. apprentissage progressif*

le script qui les guide pendant les premiers temps, au moins en ont-ils bénéficié pour efficacement apprendre leurs politiques.

4.3. *Progression en nombre*

Comme précédemment, l'apprentissage avec progression va être ici comparé à l'apprentissage sans progression (standard). On va toutefois commencer par regarder l'efficacité des politiques apprises dans le cas des expérimentations précédentes (à 2 agents et 2 cubes - que l'on notera *2a2c*) quand elles sont réutilisées dans des mondes plus peuplés.

Que ce soit dans le cas d'une simple réutilisation ou pour procéder à un apprentissage progressif, les agents étudiés sont initialisés avec les politiques apprises en $2a2c$.

Résultats : simple réutilisation

On commence donc par utiliser des agents dont le comportement fixé a été appris dans le cas $2a2c$, ce qui n'est pas supposé être particulièrement efficace avec plus de cubes. Néanmoins, cela leur donne d'assez bonnes réactions pour obtenir quelques bons résultats.

Différents tests ont été menés avec 2, 4, 6, 8 ou 10 cubes et 2, 4, 8, 16 ou 20 agents (ici dans un monde de taille 10×10). Nous avons utilisé des séries de 1000 pas de temps pour évaluer les performances des agents, chacune commençant dans des configurations aléatoires. Comme des blocages peuvent apparaître, 100 séries sont effectuées à chaque fois pour en déduire une efficacité moyenne.

La table 2 donne l'efficacité moyenne dans chacun des 25 cas (l'efficacité étant le nombre de fusions effectuées en 1000 coups). On peut comparer ces résultats avec ceux de la table 3 qui évalue l'efficacité d'agents ayant un comportement totalement aléatoire et placés dans les mêmes environnements. Il apparaît alors clairement que des comportements appris par des agents ayant à gérer 2 cubes sont encore efficaces dans des environnements plus complexes. Au travers de ces résultats, une tendance semble se dessiner : sauf pour 2 cubes, un nombre croissant d'agents améliore les résultats jusqu'à ce que les agents se gênent les uns les autres et amènent plus de problèmes de coordination qu'ils n'en résolvent.

Pour un nombre donné d'agents, avec un nombre croissant de cubes, il semble exister un seuil au delà duquel l'ajout d'une cube fait baisser les performances des agents. Une analyse plus qualitative des comportements observés nous a montré que les agents tombent facilement dans des séquences de mouvements oscillantes. Les agents, à cause de leur perception partielle et de leur absence de communication, semble avoir des difficultés pour travailler ensemble sur le même sous-ensemble de cubes et hésitent sans cesse entre deux options qui leut paraissent tour à tour valables.

cubes	agents				
↓	2	4	8	16	20
2	40.4	30.0	20.0	12.7	11.0
4	7.6	17.1	17.5	13.9	12.9
6	3.4	11.2	14.7	15.7	16.5
8	1.9	8.6	13.5	15.9	18.0
10	1.6	6.7	11.0	17.7	20.6

Tableau 2. *Réutilisation agents 2a2c :Efficacités moyennes*
(nombre de fusions pour 1000 coups)

cubes	agents				
↓	2	4	8	16	20
2	0	0	0.1	0.3	0.4
4	0	0.1	0.2	1.2	1.8
6	0	0	0.5	1.9	3.6
8	0.1	0.2	1.0	4.1	6.0
10	0.1	0.3	1.1	6.1	7.3

Tableau 3. *Agents aléatoires : Efficacités moyennes*
(nombre de fusions pour 1000 coups)

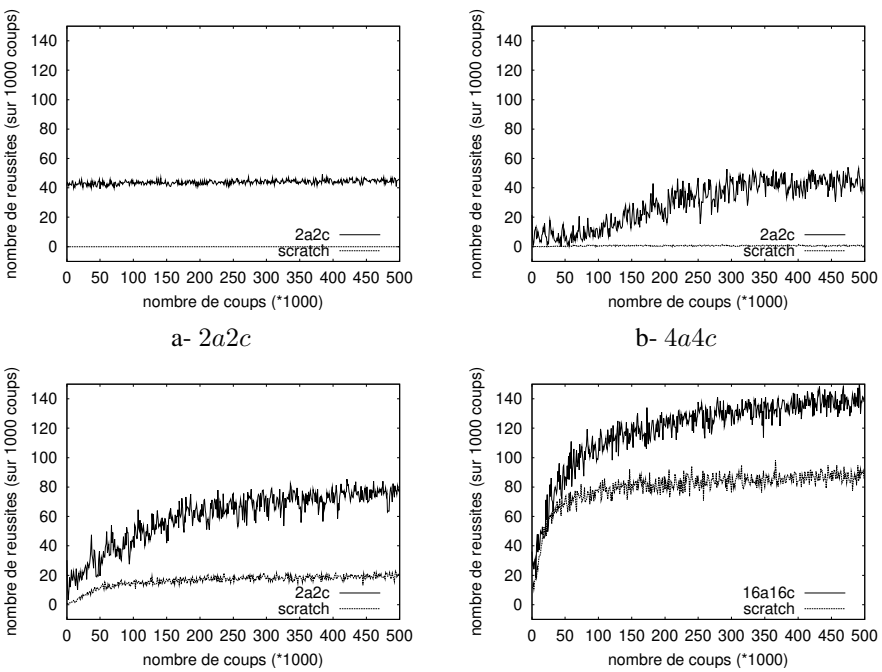
Résultats : apprentissage progressif

Si, comme nous venons de le voir, des comportements appris pour une situation à 2 agents et 2 cubes peuvent être réutilisés dans des environnements plus encombrés, il peut être intéressant de s'en servir de base pour adapter les agents à cette nouvelle situation en prolongeant l'apprentissage, ce qui constitue notre proposition d'apprentissage progressif en nombre.

Pour évaluer cette méthode, des apprentissages d'agents dans différentes situations ont été comparés selon qu'ils étaient accomplis à partir d'agents néophytes ou à partir d'agents dotés d'une politique $2a2c$. Les figures 3 a, b, c et d montrent les résultats dans quatre situations distinctes.

L'apprentissage pour des agents néophytes est ici difficile, surtout dans les cas a et b car la taille de l'environnement (qui est plus grand que l'environnement de la section 4.3) fait que les agents reçoivent beaucoup moins fréquement des récompenses non-nulles. En fait, dans les cas a et b, les agents néophyte n'apprennent rien et leur comportement est le même que celui d'agents aléatoires. En effet, dans chaque cas, les

agents néophytes évalué à l'instant 0 sont en faire des agents aléatoires ayant évolué pendant 1.000 pas de temps dans l'environnement. Quand il y a plus d'agents, mêmes les agents néophytes peuvent apprendrent car la fréquence des récompenses est plus grande.



a- $2a2c$ b- $4a4c$

c- $8a8c$ d- $16a16c$

Figure 3. *Comparaison entre apprentissage standard (à partir de zéro) et apprentissage progressif pour différents environnements*

Pour revenir au problème de départ de cette étude, ces essais confirment clairement que des agents ayant déjà l'expérience de la politique à adopter dans un problème du même type ont une progression beaucoup plus rapide que s'ils étaient de réels débutants. Dans le cas a, on est même dès le départ à un niveau d'efficacité optimal (ce qui n'a rien d'étonnant puisque ce sont quasiment les conditions de l'apprentissage initial). Outre la vitesse d'évolution du groupe, l'efficacité maximale atteinte par les agents via la méthode progressive est de loin meilleure que les performances obtenues par un apprentissage en partant de zéro. Comme on pouvait le prévoir, des agents utilisant des comportements $2a2c$ viennent avec des connaissances qui leur permettent de trouver un meilleur optimum local.

5. Discussion et travaux similaires

Nos expériences ont montré que le problème de la coordination des agents restait le point crucial à résoudre pour améliorer les performances de notre algorithme. En particulier, quand le nombre d'agent augmente, ces derniers ont du mal à se coordonner et il est difficile pour deux agents qui "veulent" travailler ensemble de s'intéresser à la même paire de cube.

Boutilier (dans [BOU 96]) s'intéresse explicitement à ce problème de coordination. Son étude, principalement théorique, définit le cadre des processus décisionnels de Markov Multi-agents. La coordination passe alors par un algorithme centralisé de résolution et demande à ce que les agents aient des capacités cognitives (reconnaissance des autres agents, perceptions plus globales, conventions sociales) beaucoup plus élaborées que celles de nos agents réactifs. De plus, il ne traite pas le cas de l'apprentissage dans lequel nous nous sommes placés.

En fait, pour un agent donné, essayer de reconnaître les autres agents pour tenter d'apprendre leur comportement et d'en tenir compte pour choisir une action peut permettre d'améliorer la coordination et la coopération globale. Il faut cependant être sûr de bien modéliser les autres agents car, comme l'ont montré Hu et Wellman (cf. [HU 98]), une mauvaise modélisation est pire que pas de modélisation. En outre, cette approche peut-être sans fin car si l'agent A modélise l'agent B, ce dernier doit modéliser A comme un agent sachant en modéliser un autre, mais alors A doit modéliser B comme un agent modélisant un agent qui sait modéliser un autre agent, etc... De plus, cette approche nécessite des perceptions complexes qui ne seront plus adaptées à un apprentissage progressif.

Une autre solution consisterait à utiliser des communications entre les agents. Le remède peut être pire que le mal, car cette communication a un coût (cf.[XUA 00]) et il n'est pas facile de déterminer la meilleure sémantique de ces communications : action envisagée, modèle du monde, comportement actuel, utilité des différentes actions... Néanmoins, c'est une question que nous comptons creuser à l'avenir.

Un autre paramètre important de l'apprentissage est la fonction de récompense. Il est possible d'utiliser la récompense pour guider l'apprentissage et, comme dans notre cas, de s'en servir pour doter les agents de comportements plus performants que ceux issus d'un apprentissage classique. Matarić, dans [MAT 97], utilise cette méthode en fournissant aux agents, non plus une fonction de récompense quasi booléenne (positive au but, rien ailleurs) mais une fonction beaucoup plus progressive, donnant de nombreux indices aux agents. Cette méthode nécessite plus de travail de la part du concepteur, elle est de plus très liée à la tâche à laquelle sont confrontés les agents. Pour chaque nouvelle tâche, il faut de nouveau repenser la fonction de récompense.

Il faut de plus noter que la fonction de récompense n'est pas facile à définir. Il est en particulier délicat de savoir comment récompenser individuellement chaque agent alors que la tâche est souvent globale. L'idéal, comme dans notre exemple, est que chaque agent puisse déterminer localement sa récompense, mais ce n'est pas toujours

aisé. Ce problème est peu abordé dans la littérature. L'approche COIN (cf. [WOL 99]) s'intéresse à l'adaptation d'une fonction de récompense globale à chaque agent, mais en fait se limite à séparer les agents en sous-groupes indépendants. Appliquée à notre exemple, cette approche aurait rassemblé les agents en un seul sous-groupe sans nous donner plus d'indications sur la forme de la récompense.

Dans un avenir proche, nous tenons beaucoup à travailler sur le problème de la définition de la récompense. Outre le problème de la répartition de la récompense globale que nous venons d'évoquer, nous voulons automatiser la définition des différentes étapes de progression de notre algorithme d'apprentissage progressif. Comme l'a montré Asada dans [ASA 96], si l'on dispose d'un ordonnancement grossier des étapes de l'apprentissage, ce dernier peut être géré automatiquement. Nous pensons que l'agent pourrait tirer parti de la fonction de récompense pour définir lui-même cet ordre grossier de situations d'apprentissage.

6. Conclusion

Dans cet article, nous nous sommes intéressés au problème de la conception automatique d'un système multi-agents composé d'agents réactifs et coopérants. Nous proposons d'utiliser des algorithmes d'apprentissage par renforcement pour que chaque agent adapte individuellement son comportement local afin de traiter une tâche globale. Ce problème d'apprentissage est théoriquement sans solution à cause de son aspect décentralisé et de la perception partielle du système par les agents apprenant.

Nous proposons alors une méthodologie d'apprentissage progressif, aussi bien au niveau de la complexité de la tâche que du nombre d'agent, pour pallier ces difficultés. Cette méthodologie est assez générique et peut être adaptée à différentes tâches sans adapter spécifiquement les agents à la tâche donnée.

Notre avons testé notre approche en simulation pour un problème où la coordination des agents est nécessaire pour la réalisation du but global. Ces expériences ont confirmé l'efficacité de notre méthode car les performances issues de notre apprentissage progressif sont bien meilleures que celles obtenues par un apprentissage classique. De plus, nous avons montré qu'il est plus facile d'apprendre une tâche complexe de manière progressive que d'apprendre cette tâche directement.

Néanmoins, plusieurs axes de recherche peuvent permettre d'améliorer notre algorithme. Ainsi, nous voulons étudier les apports possibles de la communication pour permettre une meilleure coordination et la modélisation des autres agents pour doter les agents de comportements plus riches et plus performants. Nous aimerions aussi nous pencher sur le problème plus délicat de la définition automatique des étapes de l'apprentissage progressif : comment l'agent peut-il utiliser les informations contenues dans la récompense qu'il reçoit pour décider de lui-même de la progression de son apprentissage ?

7. Bibliographie

[AND 99] ANDREW NG D. H., RUSSELL S., « Policy invariance under reward transformations : Theory and application to reward shaping », *Proceedings of the Sixteenth International Conference on Machine Learning, ICML-99*, 1999.

[ASA 96] ASADA M., NODA S., TAWARATSUMIDA S., HOSODA K., « Purposive behavior acquisition for a real robot by vision-based reinforcement learning », *Machine Learning*, vol. 23, 1996, p. 279–303.

[BAX 99] BAXTER J., BARTLETT P., WEAVER L., « Infinite horizon gradient-based policy search : II. gradient ascent algorithms and experiments », rapport, 1999, The Austrialian National University, Canberra, Australia.

[BER 00] BERNSTEIN D., ZILBERSTEIN S., IMMERMAN N., « The Complexity of Decentralized Control of Markov Decision Processes », *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, Stanford, California*, 2000.

[BOU 96] BOUTILIER C., « Planning, Learning and Coordination in Multiagent Decision Processes », *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge (TARK '96), De Zeeuwse Stromen, The Netherlands*, 1996.

[DUT 00] DUTECH A., « Solving POMDP using selected past-events », *Proceedings of the 14th European Conference on Artificial Intelligence, ECAI2000*, 2000.

[HU 98] HU J., WELLMAN M., « Online learning about other agents in a dynamic multiagent system », *Second International Conference on Autonomous Agents*, 1998, p. 239–246.

[MAT 97] MATARIC M., « Reinforcement Learning in the Multi-Robot Domain », *Autonomous Robots*, vol. 4(1), 1997, p. 73–83.

[PUT 94] PUTERMAN M., *Markov Decision Processes : discrete stochastic dynamic programming*, John Wiley & Sons, Inc. New York, NY, 1994.

[RAN 98] RANDLØV J., ALSTRØM P., « Learning to Drive a Bicycle using Reinforcement Learning and Shaping », *Proceedings of the 15th International Conference on Machine Learning, (ICML-98)*, 1998.

[SIN 94] SINGH S., JAAKKOLA T., JORDAN M., « Learning without state estimation in partially observable markovian decision processes. », *Proceedings of the Eleventh International Conference on Machine Learning.*, 1994.

[SUT 98] SUTTON R., BARTO G., *Reinforcement Learning*, Bradford Book, MIT Press, Cambridge, MA, 1998.

[WAT 89] WATKINS C., « Learning from delayed rewards. », PhD thesis, King's College of Cambridge, UK., 1989.

[WOL 99] WOLPERT D., WHEELER K., TUMER K., « General principles of learning-based multi-agent systems », *Proceedings of the Third International Conference on Autonomous Agents, Seattle*, 1999, p. 77–83.

[XUA 00] XUAN P., LESSER V., ZILBERSTEIN S., « Communication in multi-agent markov decision processes », *Proc. of ICMAS Workshop on Game Theoretic and Decision Theoretis Agents*, Boston, MA., 2000.