Rapport d'avancement de thèse :

Apprentissage par Renforcement ▷ pour la Conception ◁ de Systèmes Multi-Agents

Olivier Buffet

Comité de thèse :

Frédéric Alexandre
François Charpillet
Alain Dutech
Frédéric Garcia
Claude Kirchner
Manuel Samuelides
Olivier Sigaud

Table des matières

Chapitre I

Introduction

1 Le sujet

L'idée qui a lancé cette thèse est d'utiliser un outil, *l'Apprentissage par Renforcement*, pour en concevoir d'autres, ici des *Systèmes Multi-Agents* (SMA). Ces systèmes multi-agents sont issus de l'Intelligence Artificielle Distribuée. Leur principe est que, de l'interaction d'entités relativement simples (les agents), peut émerger un comportement de groupe complexe et dans lequel on peut reconnaître une certaine forme d'intelligence.

Si l'on sait qu'il existe d'intéressants phénomènes émergents de ce type, il s'avère par contre difficile de les provoquer. Aucune théorie efficace n'existe qui permette, étant donné un comportement global souhaité, de définir des agents dont l'interaction soit génératrice du dit comportement. En l'absence de méthode directe de conception, on se propose d'utiliser des méthodes d'optimisation, c'est à dire des algorithmes cherchant dans l'espace des systèmes multi-agents possibles (que l'on devra restreindre) celui qui répond le mieux à nos attentes, attentes qui s'expriment à travers une mesure de performance à maximiser.

Nous allons faire usage d'un principe d'optimisation assez général qu'est l'apprentissage par renforcement. L'idée est d'améliorer un comportement en cherchant à maximiser une récompense reçue (aussi appelée "signal de renforcement"). Il faut donc au début que l'agent tâtonne pour voir quelles situations et quelles actions sont associées à des récompenses positives (bonus) ou négatives (malus), et en déduire au fur et à mesure l'attitude à adopter pour maximiser sa récompense moyenne.

Plutôt que de chercher directement un système multi-agent, le choix a été fait de distribuer l'apprentissage sur chaque agent, chacun recevant un signal de renforcement propre. Ce choix correspond à des situations plus réalistes. On espère alors que de ces apprentissages concurrents va émerger le comportement souhaité, mais nous verrons que la tâche pose un certain nombre de difficultés.

2 Retour sur quelques notions

Avant de développer les notions *d'Apprentissage par Renforcement* (chapitre II) et de *Syst èmes Multi-Agents* (chapitre III), ainsi que les travaux effectués jusqu'à présent lors de cette thèse (chapitre IV), nous allons revenir sur quelques notions utiles.

2.1 Intelligence (Artificielle)

Pour commencer, un point commun aux notions d'apprentissage par renforcement, d'agent et de système multi-agents est qu'elles représentent toutes trois des approches du domaine de *l'Intelligence Artificielle* (dans l'acception choisie ici pour ces différents termes du moins). Créer ainsi une intelligence ne revient pas à faire une tête bien pleine dotée de toutes les réponses aux questions relatives à un domaine de prédilection. Définir l'intelligence est délicat, mais on retiendra ici que les capacités à raisonner, à apprendre, à s'adapter face à de nouveaux problèmes en sont des éléments principaux que nous recherchons.

2.2 Agent (Rationnel)

La notion *d'agent* pose aussi des problèmes de définition à cause, principalement, des différents usages qui en sont faits. Souvent, comme l'illustre la figure I.1, on parle d'un agent comme étant une entité (matérielle ou logicielle) située dans un environnement E qu'elle perçoit (perception p) et sur lequel elle peut agir (action a). Mais cela ne dit pas comment un agent agit.

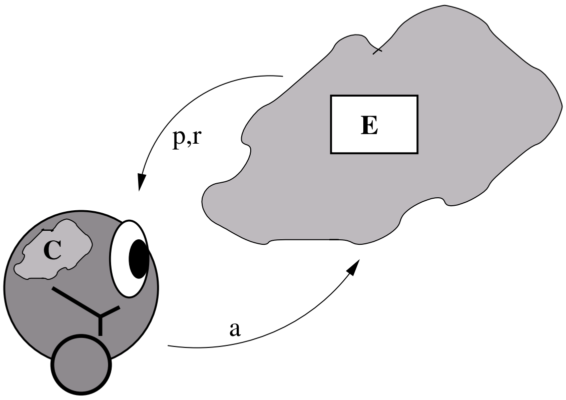


FIG. I.1 – Schéma présentant un agent en interaction avec son environnement

Dans leur chapitre sur les agents intelligents dans [RN95], Russell et Norvig définissent la notion plus précise *d'agent rationnel*[1], lequel agit en cherchant le "succès". Autrement dit, l'agent est guidé par une motivation propre et agit de lui-même (est pro-actif), ce qui le différencie par exemple de la notion informatique d'objet :

"Les objets travaillent gratuitement ; les agents travaillent pour de l'argent.[JSW98]"

Mais cette idée qu'un agent rationnel agit au mieux amène à mesurer la performance du comportement adopté (à travers les récompenses r). Cette mesure est un choix essentiel dans la conception d'un agent, d'autant que si elle est mal définie, on risque d'aboutir à un agent "astucieux" tel qu'un aspirateur qui répandrait de la poussière afin de pouvoir en ramasser plus.

Une définition complète utilement celles déjà fournies jusqu'ici : celle *d'agent rationnel id éal*, terme qui désigne un agent se comportant de façon à maximiser son espérance de gain étant données ses connaissances du monde, son expérience passée, et l'historique de ses perceptions. Cette description plus détaillée montre comment peut fonctionner un agent rationnel. Il ne manque que des considérations sur les contraintes matérielles de l'agent (mémoire disponible, puissance de calcul...) pour que l'agent soit tout à fait réaliste.

On supposera souvent ici qu'un agent est rationnel, idéal et réaliste. Cette description d'un agent peut être rapprochée de l'idée d'intelligence en tant que capacité d'adaption, de raisonnement motivé par un but. Comme cela a déjà été évoqué, l'agent est à l'intersection des deux domaines qui nous intéressent puisque c'est en travaillant au niveau de l'architecture des agents que nous cherchons à en faire un groupe coopérant (chapitre III), et que l'approche choisie pour concevoir un agent est d'utiliser l'apprentissage par renforcement (chapitre II).

[1]Rationnel : conforme à la raison.

Chapitre II

Apprentissage par Renforcement

Ce chapitre présente le principe de l'apprentissage par renforcement (section 1), puis (section 2) le formalisme choisi pour mettre en œuvre ce type d'apprentissage dans le cadre de cette thèse : les Processus de Décision Markoviens (MDP). Ce modèle n'est que rarement utilisable pour répondre à des problèmes concrets, ce qui amène à une extension aux cas d'Observabilité Partielle (section 3) et à un certain nombre d'approches pour répondre aux difficultés rencontrées (section 4).

1 Principe

Comme l'explique [KLM96], l'apprentissage par renforcement se base sur la plaisante idée d'une méthode de programmation ne nécessitant que de spécifier des moments auxquels punir ou récompenser l'agent[1]. Il n'est nul besoin de lui indiquer que faire dans telle ou telle situation, l'agent se charge de l'apprendre par lui même en renforçant les actions qui s'avèrent les meilleures.

On considère donc un agent situé dans un environnement et avec lequel il peut interagir comme nous l'avons déjà décrit à l'aide de la figure I.1. L'agent perçoit d'une part cet environnement (*perceptions* p), en reçoit une récompense (r), et peut d'autre part agir (*actions* a). Agent et environnement doivent être décrits non seulement par leurs moyens d'interaction, mais aussi par leurs fonctionnements propres : les lois régissant l'évolution de l'environnement (lois E), et le comportement de l'agent (noté C).

L'agent ne connaît la situation dans laquelle il se trouve que par l'intermédiaire de l'historique de ses perceptions. Son but, dans le cadre de l'apprentissage par renforcement, est de trouver le comportement le plus efficace, c'est à dire savoir, dans chaque situation possible, quelle action accomplir pour maximiser l'espérance de ses gains.

2 Processus de Décision Markoviens

Après être passés de la notion d'agent à l'idée d'apprentissage par renforcement, nous allons voir que l'on arrive assez naturellement au formalisme des Processus de Décision Markoviens (MDP) et aux algorithmes associés (pour une vue plus large du sujet, voir [SB98]). Il faut noter néanmoins que les hypothèses suivies par ce modèle sont parfois très simplificatrices (fonctionnement de l'environnement qui reste toujours le même, observabilité totale) et que l'apprentissage reste difficile.

2.1 Formalisme

Un *Processus de Décision Markovien (MDP)* [Put94] [SB98] est décrit par un uplet $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ défini comme suit :

- $\mathcal{S} = \{s\}$ un ensemble d'*états* (On fait l'hypothèse forte que l'on a accès à l'état complet du système.).
- $\mathcal{A} = \{a\}$ un ensemble d'*actions*.

[1] Nous adopterons en général le point de vue "agent", même si l'apprentissage par renforcement n'y est pas lié au départ.

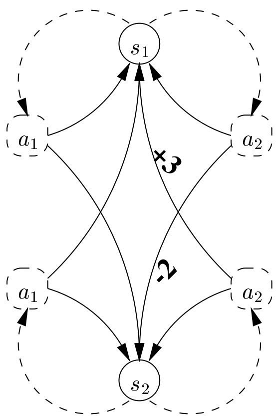
– $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$ une *fonction de transition* d'états, dans laquelle un élément de $\Pi(\mathcal{S})$ est une distribution de probabilités sur l'ensemble $\mathcal{S}$ (c'est à dire une fonction qui, à un état s, fait correspondre une probabilité). On écrira souvent la probabilité de passer entre les instants t et $t+1$ de l'état s à l'état s' sachant l'action choisie a :

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

– $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \Pi(\mathbb{R})$ une *fonction de récompense* qui, à une transition, associe une distribution de probabilité sur $\mathbb{R}$.

Dans la définition choisie, $\mathcal{T}$ et $\mathcal{R}$ sont *stationnaires* (invariants dans le temps) : ces deux fonctions ne dépendent pas du temps. De plus, on travaillera en général sur des Processus de Décision Markoviens *finis*, c'est à dire dont les deux ensembles $\mathcal{S}$ et $\mathcal{A}$ sont finis (ce qui n'est pas le cas par exemple si le problème posé est situé dans un espace continu).

La figure II.1 illustre la définition d'un Processus de Décision Markovien sur un exemple assez simple (les probabilités de transition $T(s, a, s')$ ont été omises pour des raisons de lisibilité).



Fonctionnement de ce MDP :
– Depuis un état (s_1 ou s_2), l'agent peut choisir l'une des deux actions (a_1 ou a_2).
– En fonction de ce choix (indiqué par une flêche en pointillés) s'effectue la transition (indiquée par une flêche continue) vers le prochain état (en suivant la loi $\mathcal{T}$).
– L'agent reçoit alors une récompense (ici déterministe et représentée par une valuation de la "flêche-transition").

FIG. II.1 – Un exemple de MDP à 2 états et 2 actions.

Propriété de Markov

Une propriété importante que respecte tout MDP est que la probabilité de transition d'un état s à un état s' en effectuant l'action a ne dépend pas des états et actions précédents. Il n'est alors nul besoin de mémoire pour prendre des décisions au mieux : seule la connaissance de l'état courant est utile. La seule trace d'une mémoire réside dans l'apprentissage de comportement effectué par l'agent.

Politique

Ce formalisme fournit toutes les données d'un problème d'apprentissage par renforcement, lequel consiste alors à trouver un comportement qui soit le plus "performant" possible. Dans le formalisme des MDPs, ce comportement est décrit par une fonction $\pi : \mathcal{S} \rightarrow \Pi(\mathcal{A})$ appelée politique (a priori stochastique), pour laquelle on utilisera souvent la notation $\pi(s, a) = P(a_t = a | s_t = s)$.

Dans le cadre des Processus de Décision Markoviens, il n'y a pas nécessairement unicité de la solution (la politique optimale cherchée). Mais il faut noter qu'au moins l'une d'entre elles est déterministe, ce qui permet de réduire l'espace de recherche comme nous le verrons ultérieurement.

Fonction de valeurs

Mais une question est restée en suspend : comment mesurer la performance d'un comportement donné ? La mesure de performance qui semble le plus naturellement associée à un problème de la forme d'un MDP est l'*espérance de gain* si l'on suit la politique π à partir de l'état courant s. Néanmoins, on préfère souvent une *espérance de gain escompté* qui fait apparaître dans le calcul du gain un coefficient de décroissance $\gamma \in]0;1[$:

$$V_\pi(s) = E_\pi \left(\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s \right)$$

Cette formule calcule donc une espérance de gain si l'on part de l'état initial $s_0 = s$, que l'on suit la politique π, et en notant r_t la récompense obtenue au pas de temps t (récompense pondérée par un coefficient γ^t).

Il faut toutefois noter que ce critère n'est pas la meilleure évaluation de la politique, puisqu'il défavorise d'autant plus les gains à long terme que γ est petit. Mais il reste le plus pratique à mettre en œuvre et se retrouve dans de nombreux algorithmes classiques dans le cas de MDPs à *horizon infini* (dans lesquels le système fonctionne indéfiniment).

Mais que sait l'agent ?

Avant de passer à la présentation d'un algorithme de résolution de MDPs proprement dit, précisons que, désormais, nous considérons le cas où l'agent ne connaît pas de modèle du monde : ni $\mathcal{T}$ ni $\mathcal{R}$ ne lui sont connus. De plus, les algorithmes utilisés ne chercheront pas à apprendre de tels modèles avant de chercher une politique optimale.

2.2 Apprendre un comportement sans modèle : le Q-learning

Un algorithme classique pour les MDPs sans modèle est le Q-learning, lequel fait un apprentissage par essai-erreur en profitant de la propriété de Markov et surtout en évaluant l'utilité de sa politique.

Principe

Il se trouve que, si un comportement π est optimal, la fonction V_π est maximale pour tout s. On peut ainsi évaluer V^* (l'utilité commune à toutes les politiques optimales) en cherchant à optimiser la politique localement. Pour mieux lier la politique et son évaluation, on va même introduire une fonction $Q_\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ qui donne l'espérance de gain si de l'état s on effectue l'action a avant de suivre la politique π.

Le principe du Q-learning est de faire co-évoluer une politique et une table de Q-valeurs associée. D'une part la politique π est déduite des Q-valeurs sachant que la meilleure action à choisir dans un état s est celle qui maximise $Q(s,a)$. D'autre part l'algorithme se sert de chaque nouvelle expérience $(s,a) \rightarrow (s',r)$ (liée à la politique π) pour mettre à jour l'évaluation de Q (voir en ligne 6 de l'algorithme 1 la formule dérivée de l'équation de Bellman [Bel57]). Une fois que l'algorithme a convergé, on obtient une politique déterministe optimale π^* et la table associée Q^*.

Algorithme

Il reste à préciser que la mise à jour tient de moins en moins compte des nouvelles expériences, et ce grâce au paramètre α, lequel doit vérifier $\sum_{t=0}^{\infty} \alpha_t > \infty$, et $\sum_{t=0}^{\infty} {\alpha_t}^2 < \infty$ pour que l'algorithme converge presque sûrement [JJS94]. On obtient alors l'algorithme 1 :

Pour favoriser un comportement exploratoire en début d'apprentissage, nous avons choisi d'utiliser une loi de Boltzmann, telle qu'utilisée en physique statistique, pour associer Q-valeurs et politique. La probabilité d'effectuer l'action a dans l'état s s'écrit alors :

$$\pi(s,a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_{b \in \mathcal{A}} e^{\frac{Q(s,b)}{T}}}.$$

Pour des valeurs du paramètre "température" $T > 0$ grandes, toutes les actions seront tirées de manière équiprobable, alors que, quand T tend vers 0, la politique suivie devient quasi-déterministe (l'action associée à la plus grande Q-valeur ayant les plus grandes chances d'être choisie).

Algorithme 1 Q-learning

1: $Q(s, a) \in \mathbb{R}$ arbitrairement initialisé pour tout $(s, a) \in \mathcal{S} \times \mathcal{A}$
2: Initialiser s
3: **pour tout** pas de temps **faire**
4: Choisir a pour s en utilisant une politique dérivée de Q
5: Effectuer l'action a ; observer r, s'
6: $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'}[Q(s', a')] - Q(s, a)]$
7: $s \leftarrow s'$
8: **fin pour**

Limitations

Le Q-learning est assez simple dans son principe comme dans sa mise en œuvre. Néanmoins il souffre assez rapidement d'une explosion combinatoire quand l'espace d'états augmente (par l'ajout d'une variable d'état par exemple). Une solution possible est parfois de ne plus travailler sur les états du système, mais sur des observations de ces dits états, comme nous allons le voir dans la section 3 suivante.

3 Observabilité Partielle (POMDP)

Si l'on propose parfois de s'intéresser à un espace d'observations partielles et non plus d'états, c'est en général une contrainte imposée par les limitations des capteurs (localité, imprécision), ou d'autres capacités de l'agent (mémoire). On se retrouve alors dans une nouvelle classe de problèmes, celle de Processus de Décision Markoviens Partiellement Observables (POMDP définis en 3.1). On perd ainsi certaines propriétés des MDPs, ce qui amène à utiliser de nouveaux algorithmes comme nous allons le voir ici à travers un Q-learning modifié pour ce cadre (3.2), puis une montée de gradient mieux adaptée (3.3).

3.1 Processus de Décision Markoviens Partiellement Observables

Définition

Si à la base on travaille toujours sur un MDP, la définition d'un *Processus de Décision Markovien Partiellement Observable (POMDP)* doit être complêtée, et s'écrit alors :

- $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ un Processus de Décision Markovien.
- Ω un ensemble d'*observations*.
- $\mathcal{O} : \mathcal{S} \to \Pi(\Omega)$ une *fonction d'observation*. La probabilité de l'observation o sachant qu'on est dans l'état s sera ici notée :
$$\mathcal{O}(s, o) = P(o_t = o | s_t = s)$$

Aspect non-markovien

Il faut noter qu'on perd une propriété qui était largement utilisée pour la résolution de MDPs : le processus d'observation obtienu ici n'est plus markovien. La probabilité d'observer o_{t+1} au prochain pas de temps peut dépendre d'observations et d'actions antérieures à o_t et a_t :

$$P(o_{t+1} | a_t, o_t) \neq P(o_{t+1} | a_t, o_t, a_{t-1}, o_{t-1}, \dots)$$

On peut illustrer ce phénomène par l'observation de l'historique des couleurs d'un feu tricolore. Cette observation améliore l'estimation de la couleur à venir après l'orange, puisqu'elle permet de différencier un fonctionnement "normal" (cycle vert-orange-rouge) d'un fonctionnement "orange clignotant".

3.2 Recherche directe d'une politique : Q-learning

Nous allons voir ici en quelques points comment on peut adapter le Q-learning pour fonctionner dans une situation d'observabilité partielle, et ce, même si cette approche n'est que sous-optimale.

Espace d'entrée

En décrivant les agent rationnels idéaux est apparue l'idée que le mieux était d'agir en fonction de toute l'historique des observations (ce qui se justifie en l'absence de la propriété de Markov). Or l'ensemble de ces historiques est infini (puisqu'elles peuvent être de n'importe quelle longueur). On va donc devoir se restreindre à un sous-ensemble fini. Couramment, comme ici, on fait le choix de ne tenir compte que de l'observation courante ($\pi : \Omega \rightarrow \Pi(\mathcal{A})$).

Politique stochastique

Une riche discussion sur l'apprentissage sans estimation d'état dans les POMDPs est développée dans [SJJ94]. Un point important y apparaissant est qu'il n'existe plus nécessairement de politique déterministe qui soit optimale. On va donc devoir faire une recherche de politique dans le plus vaste espace des politiques stochastiques.

Ainsi, si l'on veut s'inspirer du Q-learning boltzmannien, il va falloir jouer sur la température limite T_{lim}. Cette température permet de régler un "degré de déterminisme" de la politique obtenue à partir des Q-valeurs. On peut au choix essayer de faire un apprentissage d'une température limite optimale, ou en choisir une expérimentalement (ce qui sera notre cas).

Il faut noter que l'on ne fait ici que rajouter un "degré de liberté" à notre sous-espace de politiques, et que la politique qu'on obtiendra n'est pas nécessairement optimale dans l'espace de toutes les politiques stochastiques possibles.

Estimation des Q-valeurs

Un problème supplémentaire est que la formule de mise à jour de Q n'est plus valable et doit être modifiée. Une première cause est que l'on travaille sur des observations : le calcul des Q-valeurs suppose que l'on travaille sur des états et que la propriété de Markov est respectée, ce qui n'est plus le cas comme on l'a dit en 3.1.

Cela tient d'autre part au fait que le "$\max$" de la formule de mise à jour de Q suppose une politique déterministe. Mais ce deuxième point peut être corrigé en tenant compte des différentes décisions possibles. La formule de mise à jour des Q-valeurs d'un MDP (observabilité totale) pour une politique stochastique π est ainsi :

$$Q(s,a) \leftarrow (1-\alpha) * Q(s,a) + \alpha * \left(r + \gamma \sum_{a'} [\pi(s,a') * Q(s',a')] \right).$$

On trouvera aussi une description de ces points dans [SJJ94].

Résultats

En adaptant ainsi le Q-learning, on obtient une simple heuristique sans garanties de convergence comme c'était le cas pour les MDPs. L'algorithme reste néanmoins assez efficace dans bon nombre de situations. On l'utilisera souvent par la suite puisqu'il sera l'un de nos deux algorithmes de référence avec la montée de gradient décrite ci-après (section 3.3) : c'est autour de ces deux algorithmes que sont développés nos travaux.

3.3 Recherche directe d'une politique : montée de gradient

Comme il reste préférable de faire une recherche directement dans l'espace des politiques stochastiques, nous avons décidé d'utiliser comme deuxième algorithme de référence une méthode de montée de gradient due à Baxter et Bartlett [BB01],[BBW01] dont les grandes lignes vont être présentées ici.

Rappel sur l'opérateur "gradient"

Le gradient d'une fonction continue et dérivable f définie sur plusieurs variables $x_1, \ldots, x_n$ est la fonction qui à $x_1, \ldots, x_n$ associe le vecteur :

$$\overrightarrow{\nabla} f(x_1, \ldots, x_n) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x_1, \ldots, x_n) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x_1, \ldots, x_n) \end{pmatrix}$$

Si les variables sont toutes définies sur $\mathbb{R}$ et que la fonction f est à valeurs dans $\mathbb{R}$, le gradient de f en un point $\overrightarrow{x}$ est un vecteur dont une propriété est d'indiquer la direction dans laquelle f croît le plus vite (direction de plus grande pente).

Principe de la montée de gradient

La méthode classique de *montée de gradient* a pour but de trouver un point en lequel une fonction continue et dérivable $f : \mathbb{R}^n \rightarrow \mathbb{R}$ est maximale. L'idée est de parcourir une suite de points $(\overrightarrow{x_k})_{k \in \mathbb{N}}$ se rapprochant du sommet de la surface.

A l'étape k, on part du point $\overrightarrow{x_k} \in \mathbb{R}^n$, et on suit le vecteur donné par le gradient de f en $\overrightarrow{x_k}$: $\overrightarrow{\nabla} f(\overrightarrow{x_k})$, lequel indique la direction de plus grande pente de f. Ainsi on passe à $\overrightarrow{x_{k+1}}$ par :

$$\overrightarrow{x_{k+1}} = \overrightarrow{x_k} + \lambda_k * \overrightarrow{\nabla} f(\overrightarrow{x_k}),$$

où $(\lambda_k)_{k \in \mathbb{N}}$ est une suite réelle positive décroissante définie de manière à progresser rapidement aux premières étapes (en sortant si possible de zones d'optima locaux) de l'algorithmes, puis à affiner le résultat par des pas de plus en plus petits.

Il ne s'agit que d'une heuristique puisqu'on risque de tomber dans un extremum local, et que la vitesse de convergence va dépendre du choix de la suite λ pour le problème courant. Si des variantes améliorant cette méthode existent, cette rapide description suffira pour introduire son adaptation à la recherche d'une politique.

Adaptation de notre problème à la montée de gradient

Pour appliquer le principe de montée de gradient en apprentissage par renforcement, il faut en premier lieu choisir une paramétrisation $\Theta = \{\theta\}$ de la politique de l'agent. C'est dans l'espace de ces paramètres que se fera la recherche. Une mesure de performance $V(\Theta)$ de la politique obtenue tient alors le rôle de la fonction à maximiser. Dans l'hypothèse que, étant donnée la politique choisie, le système tend vers une distribution de probabilité d'occupation des états unique et stationnaire (un seul "attracteur"), [BB01] propose de mesurer la performance d'une politique comme suit :

$$V(\Theta) = \sum_{s \in \mathcal{S}} P_\Theta(s) * r(s),$$

où $P_\Theta(s)$ est la probabilité d'être dans l'état s une fois le système à l'état stationnaire, et où la récompense r ne dépend que de l'état s. Ce calcul évalue (de manière fort logique) le gain moyen à l'état stationnaire.

On peut noter ici qu'on ne cherche pas, comme c'était le cas pour les MDPs, à connaître une espérance de gain étant donnée la situation (observation ou état). Si l'on a déjà dit en section 3.2 que les formules utilisées n'étaient plus vérifiées dans le cas de POMDPs, la principale raison est qu'un comportement optimal n'est plus nécessairement localement optimal partout. En effet il n'est plus vrai qu'une politique optimale maximise $V_\pi(s)$ pour tout $s \in \mathcal{S}$, propriété largement utilisée dans les algorithmes de résolution de MDPs. Ne pouvant plus utiliser cette propriété, on revient à un critère d'évaluation "global" de la politique tel que celui qui vient d'être décrit.

Avant d'entrer dans plus de détails, il faut préciser que la politique paramétrée par Θ est définie à travers une fonction μ qui, étant donné Θ et l'observation courante o, y associe une distribution de probabilité sur les actions. La forme que nous avons choisie est précisée ci-après.

Evaluation du gradient

Dans la situation où nous nous trouvons (sans connaissance d'un modèle du monde), on ne peut calculer le gradient de $V(\Theta)$ de manière formelle. Connaissant μ, l'évaluation du gradient (que l'on notera simplement $\vec{\nabla}$ ici) peut être faite par itération des deux calculs suivants au fur et à mesure des expérimentations :

$$\overrightarrow{z_{t+1}} = \beta \overrightarrow{z_t} + \frac{\vec{\nabla} \mu_{a_t}(\Theta, o_t)}{\mu_{a_t}(\Theta, o_t)} \tag{II.1}$$

$$\overrightarrow{\nabla_{t+1}} = \overrightarrow{\nabla_t} + r_{t+1} \overrightarrow{z_{t+1}}, \tag{II.2}$$

où o_t est l'observation à l'instant t, a_t l'action choisie et r_{t+1} la récompense reçue.

Choix de la paramétrisation

La paramétrisation est choisie pour avoir tous les degrés de liberté utiles et pour que la formulation de divers calculs (tels que le gradient) soit simple. Elle est définie par la probabilité de choisir l'action a suivante (pour une observation o donnée) :

$$\mu_a(\Theta, o) = \frac{e^{\theta_{a,o}}}{\sum_{b \in \mathcal{A}} e^{\theta_{b,o}}}$$

En remarquant, pour la formule II.1, la propriété :

$$\frac{\vec{\nabla} \mu_a(\Theta, o)}{\mu_a(\Theta, o)} = \vec{\nabla} \ln(\mu_a(\Theta, o)) \ , \tag{II.3}$$

on peut développer le calcul d'une des composantes de ce vecteur II.3 :

$$\begin{aligned} \frac{\partial \ln(\mu_a(\Theta, o))}{\partial \theta_{a',o'}} &= \frac{\partial \ln\left(\frac{e^{\theta_{a,o}}}{\sum_{b \in \mathcal{A}} e^{\theta_{b,o}}}\right)}{\partial \theta_{a',o'}} \\ &= \delta_{o,o'} * (\delta_{a,a'} - \mu_{a'}(\Theta, o')) \end{aligned} ,$$

où la fonction δ est définie par :

$$\forall (a, b) \in \mathbb{N}^2, \ \delta_{a,b} = \begin{cases} 1 \text{ si } a = b \\ 0 \text{ sinon} \end{cases} .$$

On se retrouve donc bien avec des calculs réalisables et assez légers, si ce n'est qu'il faut les effectuer pour chaque paire observation-action.

Algorithme

Sachant désormais comment évaluer le gradient et comment s'en servir pour faire une montée de gradient, on a tous les éléments nécessaires à la mise en application. On peut ainsi écrire l'algorithme 2 :

Comme la présentation qui vient d'être faite de la montée de gradient que nous utilisons, nous laissons au lecteur intéressé le soin de compléter sa lecture par les deux articles précédemment cités [BB01] et [BBW01], sachant que l'algorithme que nous utilisons, OLPOMDP (OnLine POMDP), y est décrit en détail.

4 Difficultés / diversité des approches

Un des principaux risques en apprentissage par renforcement est celui de l'explosion combinatoire : la moindre capacité perceptive supplémentaire peut faire croître l'espace des observations de manière catastrophique. Mais cette "malédiction de la dimensionalité" peut être aussi simplement due à un espace de

Algorithme 2 OLPOMDP$(\beta, T, \Theta_0) \rightarrow \mathbb{R}^K$

Entrées:

- $\beta \in [0, 1)$.
- $T > 0$.
- Des valeurs initiales du paramètre $\Theta_0 \in \mathbb{R}^K$.
- Des politiques paramétrées randomisées $\{\mu(\Theta, .) : \Theta \in \mathbb{R}^K\}$.
- Un POMDP.
- Des tailles de pas $\alpha_t, t = 0, 1, \ldots$ satisfaisant $\sum \alpha_t = \infty$ et $\sum \alpha_t^2 < \infty$.
- Un état de départ arbitraire (inconnu) s_0.

Soit $z_0 = 0$ $(z_0 \in \mathbb{R}^K)$.

pour $t = 0$ à $T - 1$ **faire**

  Observer o_t (généré d'après $v(s_t)$).

  Générer une commande a_t d'après $\mu(\Theta, o_t)$

  Observer $r(s_{t+1})$ (où le prochain état s_{t+1} est généré d'après $T(s_t, a_t, s_{t+1})$)

  $z_{t+1} \leftarrow \beta z_t + \frac{\nabla \mu_{a_t}(\Theta, o_t)}{\mu_{a_t}(\Theta, o_t)}$

  $\Theta_{t+1} \leftarrow \Theta_t + \alpha_t r(s_{t+1}) z_{t+1}$

fin pour

Sorties: Θ_T

travail continu que l'on va devoir discrétiser avec plus ou moins de détails. Mais l'on se bat toujours dans un compromis entre trop et trop peu d'information : trop et la complexité de l'apprentissage est déraisonnable, trop peu et l'on ne peut trouver de solution satisfaisante.

Vu le grand nombre de travaux visant à améliorer la recherche d'une politique optimale, nous allons nous contenter d'une rapide présentation de quelques grandes idées (un aperçu plus vaste est présenté dans [KLM96]) :

Hiérarchisation du contrôleur :

Apprendre directement un comportement complet est rarement réaliste. Faire une conception hiérarchique peut alors être une solution, en apprenant par exemple à résoudre des sous-tâches avant de les récomposer en une tâche globale. ([MPR+98][Die00])

Utilisation d'une mémoire :

Si les seules perceptions immédiates s'avèrent insuffisantes pour apprendre un comportement efficace, on peut essayer d'utiliser une mémoire des quelques dernières perceptions (une historique partielle) pour lever ce qui sont en générales des ambiguïtés (deux états cachés par une même perception). L'utilisation de contextes fait aussi partie de ces travaux sur l'emploi de mémoire. ([Dut99][McC95a][McC95b])

Discrétisation de l'espace :

Dans certains cas, les états du système peuvent être accessible, mais l'espace de ces états est alors souvent trop grand, car discret et trop détaillé, voire même continu. Quelques algorithmes ont été proposés dont le but est de discrétiser l'espace des états en peu d'états, mais de façon à avoir la politique la plus efficace possible (distinguant deux états voisins s'ils sont associés à des actions différentes). ([MA95][MM02])

Epilogue

Le chapitre suivant ouvre sur le domaine de la conception de systèmes multi-agents, auquel nous souhaitons appliquer des méthodes d'apprentissage par renforcement. Les approches qui nous intéressent seront donc orientées vers des problèmes spécifiques aux SMA.

Les deux algorithmes "de référence" présentés dans le présent chapitre (Q-learning modifié et montée de gradient) vont s'avérer utiles par la suite, puisque les observations partielles et l'aspect multi-agents vont faire perdre l'aspect markovien des classiques MDPs comme nous allons le voir assez rapidement.

Chapitre III

Systèmes Multi-Agents

1 Généralités

1.1 La résolution distribuée

Nombre de méthodes monolithiques de résolution de problèmes ont été développées, méthodes dans lesquelles une seule "entité" intervient et cherche à maîtriser toutes les données du problème posé. Cette approche devient facilement ingérable si plusieurs domaines de connaissances interviennent. De même certaines situations telles que la robotique mobile peuvent contraindre à l'utilisation conjointe de plusieurs entités (qu'on appellera désormais "agents"). L'idée est donc apparue de distribuer l'intelligence artificielle et, par exemple, de faire collaborer des agents.

1.2 Problème de méthodologie

Sans chercher à donner de définition précise, on peut considérer l'Intelligence Artificielle comme un domaine de l'informatique qui cherche à concevoir des outils de résolution de problèmes. Dans ce cadre, l'idée des Systèmes Multi-Agents (SMA) est que la résolution d'un problème peut émerger de l'interaction d'un certain nombre d'agents relativement simples.

Malheureusement on maîtrise très mal la théorie de ces phénomènes émergents. Il n'est de ce fait pas possible, étant donné le comportement de groupe escompté, d'en déduire le comportement que doit adopter chaque agent.

L'étude de systèmes biologiques peut inspirer la conception des agents dans certaines situations. Ces situations sont encore assez limitées et correspondent assez souvent à des imitations de systèmes biologiques. Nous préfèrerons une autre idée aux domaines d'application a priori plus ouverts, et qui cherche à être la plus générique possible : l'idée d'utiliser des méthodes d'apprentissage par renforcement (voir chapitre II). Il n'est alors besoin que de spécifier les situations bonnes ou mauvaises qui permettent de définir le but à atteindre, et de laisser l'apprentissage se faire tout seul.

Aparté : Si "inspiration biologique" et "apprentissage par renforcement" ont été ici distinguées, il faut noter que ces deux approches ne sont aucunement en opposition. L'article de Meuleau et Dorigo [MD00] donne d'ailleurs un très bon exemple en faisant le lien entre algorithmes de colonies de fourmis (figure III.1) et descente de gradient stochastique.

2 Conception de Systèmes Multi-Agents

2.1 Que (dé)centraliser ?

Une première remarque est que, si l'on veut au final que chaque agent se comporte de façon autonome, l'idée qui vient d'être décrite d'utiliser l'apprentissage par renforcement n'exclue pas une phase

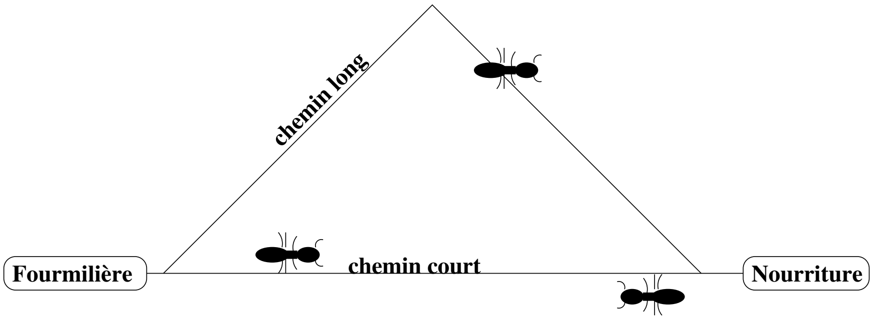


FIG. III.1 – Illustration de la recherche d'un plus court chemin par "algorithme fourmi".

d'apprentissage qui soit centralisée. On pourrait ainsi user d'un algorithme apprenant en même temps le comportement de tous les agents d'un système. Seule la mise en application serait réellement décentralisée.

L'usage normal de l'apprentissage par renforcement est d'avoir une phase d'utilisation du comportement acquis une fois la phase d'apprentissage terminée. On peut toutefois détourner cet usage pour qu'un agent sache s'adapter continuellement à de possibles évolutions de son environnement. Nous préférons ainsi utiliser l'apprentissage par renforcement aussi pendant le fonctionnement des agents, donc de manière indépendante sur chacun. De cette manière, aussi bien l'exploration que l'exploitation des comportements seront décentralisées, ces deux phases n'étant pas clairement distinctes dans le fonctionnement du système.

2.2 Perte de la stationnarité

Une première raison pour perdre la stationnarité du système est le fait que l'on ne suppose les observations que partiellement observables (voir la présentation des POMDPs en 3).

Supposons maintenant que chaque agent ait accès à l'état du système (observabilité totale) et que la loi de transition de l'environnement soit stationnaire. Un apprentissage centralisé correspondrait alors au cas d'un MDP classique, l'ensemble des agents pouvant être considéré comme une unique entité dont seuls les effecteurs sont distribués. Par contre, si chaque agent apprend son comportement de manière autonome, un agent A quelconque verra tous ses congénères comme faisant partie de l'environnement (voir figure III.2). Or les dits congénères sont aussi en phase d'apprentissage, donc en phase d'évolution de leur comportement, ce qui amène à une loi de transition de "l'environnement vu par l'agent A" qui n'est pas stationnaire.

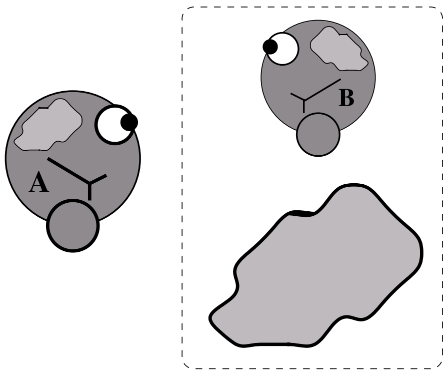


FIG. III.2 – Une situation classique : l'agent (A) considère les autres agents (ici B) comme faisant partie de l'environnement.

Avec une loi de transition non stationnaire, nous perdons la propriété de Markov et sortons du cadre des MDPs. Le problème ne peut même pas être modélisé par un POMDP, à moins que l'on considère les comportements des autres agents comme faisant partie de l'état caché de l'environnement en tant que variables internes qui évoluent au cours du temps. Quoi qu'il en soit, on ne peut avoir aisément accès à un quelconque modèle du système, et il va falloir par conséquent utiliser des algorithmes de recherche de politique sans modèle qui soient adaptés aux cas non-markoviens (voir section 3).

2.3 Comment répartir la récompense

Un autre point délicat qui a déjà été évoqué est la manière de récompenser les agents. En apprentissage par renforcement, l'idéal est que toute la difficulté de conception qui puisse restée soit concentrée dans la tâche de bien définir une fonction de récompense qui "guide" vers le but recherché, c'est à dire qui définisse ce but. Dans le cadre multi-agents où nous nous trouvons, une difficulté vient s'ajouter à ce problème : une récompense doit être assignée à différents agents.

Différentes approches sont possibles. On peut supposer tout d'abord que tous les agents reçoivent la même récompense quelle qu'ait été la contribution de chacun au résultat obtenu. Dans cette première hypothèse, tous les agents ont accès à la même information sur la récompense commune (ce qui sous-entend un phénomène centralisé), mais on élude alors le problème de répartition "raisonnée" des gains.

Une deuxième solution (parmi d'autres encore) est que chaque agent sache associer une récompense à ce qu'il perçoit de l'environnement. Cette manière de faire respecte le point de vue "agent localisé", l'aspect décentralisé du problème. Par contre, il s'avère plus délicat avec cette méthode de définir les récompenses locales associées à un but global. Les travaux de Wolpert, Wheeler et Tumer sur les *COIN* mettent en évidence ce problème. Si cette approche est liée à un tel risque de perdre la complète collaboration des agents, nous la suivrons toutefois dans nos travaux, pour respecter l'aspect réaliste d'une récompense liée à ce que fait et perçoit chaque agent.

3 Quelques approches du problème

Un certains nombre de travaux ont déjà abordé l'idée de concevoir des systèmes multi-agents par des méthodes d'apprentissage par renforcement. Nous allons présenter ici quelques approches pour pouvoir situer nos travaux.

3.1 Des problèmes pratiques

Dans un premier temps, nous allons voir d'abord un cas pratique qui permet de mieux appréhender la problématique qui nous intéresse.

Robots fourrageants [Mat97]

L'approche de Maja Matarić [Mat97] a été construite en travaillant sur un problème de fouragement. Dans un environnement clos, quatre robots doivent aller chercher des palets et les ramener dans une zone prévue à cet effet, les robots devant aussi parfois se reposer dans leur "maison".

S'attaquer à une tâche aussi complexe avec des agents-robots bruts s'avère trop difficile, même s'il n'est pas ici nécessaire d'avoir une coordination forte entre agents. Le premier point de l'approche présentée dans [Mat97] est de définir un certain nombre de comportements parmi lesquels chaque robot va choisir son comportement courant. En plus sont définies des conditions qui peuvent déclencher des changements de comportements. L'apprentissage va alors consister à chercher sous quelles conditions provoquer le lancement d'un comportement donné.

On vient en fait de décrire les actions (comportements) et les observations (conditions) d'un problème comparable à un POMDP. C'est dans la partie "récompenses" que se trouve le deuxième point important de ces travaux. Si l'on retrouve les habituelles récompenses ponctuelles (pour une bonne ou une mauvaise action), une idée supplémentaire est de donner une mesure de progression pour certains critères. Un exemple est la distance à un obstacle qui peut être considérée comme une récompense négative alors qu'il s'agit de ce que l'on vient d'appeler une "mesure de progression".

Le fait d'utiliser des comportements prédéfinis et d'avoir des récompenses très précisément définies pour le problème posé font que cette approche est assez spécifique et donc pas aussi générique que la notion d'apprentissage par renforcement pourrait le laisser penser. Toutefois, il faut voir que l'on s'attaque ici à un travail dans des conditions réelles, et non sur simulateur comme c'est souvent le cas.

3.2 Quelques points de vue théoriques

Dans les travaux théoriques, on peut distinguer ceux qui permettent une analyse de la conception de systèmes multi-agents par apprentissage, et ceux qui proposent des solutions adaptées à au moins certains cas. Nous allons commencer ici par ces premiers travaux "d'analyse".

Jeux de Markov à somme nulle [Lit94]

Pour revenir donc à des approches plus théoriques, le premier exemple que nous donnons ici est un cas assez particulier, mais dont l'étude permet de bien comprendre les situations de conflits quand des agents ont des motivations concurrentes. C'est Littman qui propose dans [Lit94] d'utiliser les jeux de Markov dans des situations où deux agents sont en complète opposition (le gain de l'un est la perte de l'autre).

L'idée considérée part de l'hypothèse que l'agent dont on prend le point de vue connaît tout : le modèle du monde, ainsi que les récompenses de l'autre agent comme évidemment les siennes. Seul le comportement de son adversaire lui est inconnu. La théorie des jeux de Markov, parfois appelés jeux stochastiques [Owe82], permet alors de chercher le meilleur comportement à adopter sachant que l'adversaire a des motivations strictement contraires[1].

Un algorithme spécifique à ce cadre est minimax-Q. Il est comparable au Q-learning, mais combine les points de vue opposés des deux joueurs, d'où la présence d'un $\min$ et d'un $\max$ dans l'algorithme. Il faut noter que l'on suppose ici l'adversaire le plus intelligent possible. Si au contraire on ne fait pas d'hypothèse sur le jeu de l'adversaire, on peut apprendre avec un simple Q-learning à utiliser ses défauts (en supposant que sa politique est toujours la même). Ainsi, un gardien de but sera plus efficace s'il se rend compte que le buteur auquel il fait face a une large préférence pour le côté droit du but.

Mais les jeux de Markov ont donné lieu à d'autres développement comme nous allons le voir maintenant.

Jeux de Markov à somme non nulle [HW98a][HW98b]

Des travaux tels que ceux présentés dans [HW98a] et [HW98b] sont partis de la même approche "jeux de Markov", mais dans un cas plus orienté "multi-agents". Il s'agit en effet de considérer n agents (pas nécessairement 2), chacun ayant ses propres gains (indépendants des gains des autres agents, à la différence des jeux à somme nulle)[2]. On adopte donc le point de vue d'un agent qui cherche à maximiser ses gains en considérant que ses congénères font de même.

Le but est d'atteindre un *point fixe* du système, c'est à dire une situation d'équilibre dans laquelle aucun agent ne peut améliorer ses résultats sans que les autres changent aussi de comportement. Les agents vont devoir tomber d'accord sur le point fixe (appelé "équilibre de Nash") à atteindre, et ce, sans communication.

Un agent a le choix entre deux voies s'il fait de l'apprentissage par renforcement :

- soit il ne distingue pas les autres agents de l'environnement (on parlera d'agent simplement *comp étitif*),
- soit au contraire il les distingue, mais cela nécessite de modéliser ces autres agents.

Une première possibilité, si l'on modélise explicitement un autre agent, est d'apprendre son comportement par l'observation (on parle de modélisation de niveau 0). Un niveau de modélisation supérieure est de considérer que cet autre agent est compétitif (on dira alors adopter un niveau 1 de modélisation). Ensuite, on peut définir par récurrence un niveau $n \geq 2$ de modélisation comme une situation où un agent considère son congénère comme étant de niveau $n - 1$.

Note : En fait chaque agent fait une hypothèse sur les autres. Très facilement on va se trouver dans des situations où nécessairement l'un des deux agents se trompe. Prenons l'exemple de deux agents, le premier étant de niveau 2. Soit il se trompe, soit effectivement le second est de niveau 1, auquel cas c'est ce dernier qui se trompe.

[1] Les jeux stochastiques sont plus généraux, mais nous nous limiterons à ce cas des jeux à deux joueurs et dits "à somme nulle".

[2] Attention : on se place ici dans un jeu répété, c'est à dire un système à un seul état.

Les travaux de Hu et Wellman mettent l'idée de "modélisation de l'autre" en application dans un problème simple de marché boursier, donc non coopératif. En comparant différentes situations dans leur système à quatre agents apparaît le fait qu'une modélisation mal faite est bien pire qu'une absence de modélisation. L'idée de méta-raisonnement sous-jacente dans cette approche est intéressante, mais demanderait peut-être une analyse dans un cadre clairement coopératif.

De manière générale, l'utilisation de jeux de Markov pose problème dans la mesure où elle requiert la connaissance du modèle de l'environnement, et une observation complète de l'environnement. Les cas pratiques ne supportent pas de telles hypothèses, autant pour des raisons de puissance de calcul que de connaissances a priori ou de capacités des capteurs.

MMDP [Bou96][Bou99]

Une formalisation aussi théorique mais plus proche de réalisations possibles est celle présentée dans [Bou96][Bou99]. Pour replacer les MDPs dans un cadre multi-agents, Boutilier définit la notion de MMDP (Processus de Décision Markovien Multi-agents). Il s'agit d'un uplet $\langle \mathcal{S}, \alpha, \{A_i\}_{i\in\alpha}, Pr, R\rangle$, où : $\mathcal{S}$ et α sont des ensembles finis d'états et d'agents, respectivement ; A_i est un ensemble fini d'actions possibles pour l'agent i ; $Pr : \mathcal{S} \times A_1 \times A_2 \times \ldots A_n \times \mathcal{S} \rightarrow [0,1]$ est une fonction de transition ; et $R : \mathcal{S} \rightarrow \mathbb{R}$ est une fonction de récompense à valeurs réelles. Si on compare cette définition à celle d'un MDP, la principale différence réside dans la distinction des actions des n agents (dont la réunion est appelée *action jointe*).

On est là dans un cadre multi-agents complètement coopératif, au point que la récompense reçue de l'environnement à un instant donné est la même pour tous. Cette mise en commun de la récompense est en fait le moyen d'être sûr que tous les agents cherchent à agir pour atteindre un but commun : des récompenses propres à chacun peuvent à l'inverse provoquer des conflits locaux qui nuisent à une coopération complète. Toutefois, cette récompense partagée confronte chaque agent d'un MMDP à un problème délicat : quelle a été sa contribution dans l'obtention de cette récompense ?

La première restriction qui apparaisse dans les MMDPs (Multiagent Markov Decision Processes) est que l'on suppose une observation totale de l'environnement : chaque agent sait exactement l'état actuel du système dont lui et ses congénères font partie. Mais cette simplification du problème permet de concentrer l'attention sur seulement certaines difficultés de la coopération.

Entre agents qui coopèrent, les actions délicates sont les coordinations, c'est à dire les actions "de groupe", menées conjointement par plusieurs agents ayant chacun un rôle spécifique. Boutilier distingue parmi les réponses possibles à ces problèmes de coordinations : la communication (pour s'entendre sur la répartition des rôles), les conventions (ou lois sociales, qui définissent des règles de résolution de conflit [3]), et enfin l'apprentissage de politiques coordonnées.

On peut noter que l'idée de la communication peut être considérée comme un simple déplacement du problème : "Comment échanger des messages ?" est aussi un problème de coordination (Qui doit parler le premier ?). D'autre part, distinguer conventions et apprentissage revient à distinguer une "loi sociale" fournie à l'avance et une "loi sociale" apprise par essais-erreurs.

Le formalisme des MMDPs permet néanmoins d'appréhender certaines difficultés liées aux SMAs mais, comme c'était déjà le cas avec les jeux de Markov vus précédemment en section 3.2, il ne conduit pas pour l'instant à une solution convaincante.

DEC-POMDP [BZI00]

Une troisième formalisation qui s'approche toujours plus de situations réelles est celle des DEC-POMDPs (DECentralized-POMDPs) décrite dans [BZI00]. Sans entrer dans trop de détails, la principale différence avec les MMDPs déjà présentés en 3.2 est qu'un agent n'a accès qu'à une observation partielle (et personnelle) de l'état du système. Notons que la récompense reçue est toujours commune à tous les agents, comme c'était déjà le cas pour les MMDPs.

La contribution de [BZI00] est de préciser la complexité de la recherche d'un DEC-POMDP optimal. En effet un résultat principal est le premier théorème de l'article :

[3] Ici le conflit porte sur le rôle de chacun.

Le problème de décision pour un DEC-POMDP à horizon fini avec $m \geq 2$ agents est NEXP-complet.

La difficulté apparente de la conception d'un SMA coopératif est ainsi clairement confirmée. Il faut noter qu'une différence avec notre cadre de travail est que nos agents ont des récompenses indépendantes, puisque liées à des perceptions locales, ce qui ne simplifie la tâche en aucune manière.

3.3 Approches théoriques proposant des méthodes

La théorie permet heureusement aussi de mettre au point des méthodes comme les trois suivantes.

COIN [TW00][WWT99]

Pour en venir à des travaux proposant des méthodes pour améliorer la conception de systèmes multi-agents, un premier exemple est le cadre des COIN (COllective INtelligence) présenté, en même temps qu'un algorithme, dans [TW00] et [WWT99].

Un COIN est un système multi-agents sans communication ni commande centralisée, et pour lequel une fonction d'utilité "du monde" est définie (donc commune à tout le groupe d'agent, comme pour les MMDP et POMDP en section 3.2). L'idée de Tumer et Wolpert est de faire en sorte que cette utilité commune soit décomposable en utilités propres à chaque agent (tous ne pouvant avoir accès à l'utilité commune), de façon à ce que les variations des utilités de chacun amènent les mêmes variations à l'utilité commune.

Ce principe assure que l'amélioration de la politique de chaque agent mène à une amélioration globale du comportement de groupe. On peut ainsi être certain d'atteindre un optimum au niveau du groupe tout en n'étant pas obligé d'avoir une récompense commune à tous (ce qui requiert une centralisation). Néanmoins, la mise en application présentée dans [TW00] ne concerne qu'un problème d'optimisation de routage, et ne dit ni dans quels cas on pourra mettre en œuvre cette méthode, ni comment.

C'est dans [WWT99] que l'on peut trouver une discussion sur les problèmes dans lesquels on peut appliquer la "méthode COIN". Le point auquel Wolpert et Tumer en sont arrivés est, pour résumer leur résultat, qu'une condition suffisante est que le système multi-agents puisse être décomposé en systèmes multi-agents indépendants. Sauf nouveaux résultats, il ne semble donc pas encore évident de pouvoir automatiser la conception de COINs.

Empathie [CSC02]

Dans les algorithmes d'apprentissage par renforcement, nous n'avons pas encore parlé des algorithmes dits "de planification". Cette omission s'explique, dans le cadre multi-agents, parce qu'une co-évolution de tous les agents (un apprentissage simultané) semble nécessaire : un agent ne peut faire de planification s'il ne sait pas comment ses congénères vont se comporter.

Comme la planification s'avère souvent plus efficace qu'un apprentissage en ligne, des travaux s'intéressent à la rendre possible dans un cadre multi-agents. Ainsi, [CSC02] propose une méthode qui pallie le besoin de co-évolution. L'idée proposée "d'empathie"[4] est que, tous les agents effectuant les mêmes raisonnements (les mêmes calculs), ils peuvent les faire en connaissance du comportement de leurs congénères.

La méthode proposée commence avec des agents ayant tous un même comportement par défaut au départ. Ainsi, un quelconque agent peut planifier "son" comportement en connaissant le comportement (par défaut inchangé) des autres agents. Chaque agent effectuant le même calcul, tous adopteront la même nouvelle politique obtenue et sauront qu'elle est adoptée par chacun. On peut alors itérer le procédé de planification en connaissance du comportement commun de tous.

Cette idée d'exploiter une empathie des agents est rendue possible par la planification dont le résultat ne dépend pas d'expériences comme dans les algorithmes en ligne. Par contre, il faut avoir connaissance du modèle de l'environnement pour pouvoir mettre en œuvre des algorithmes de planification, ce qui n'est que rarement le cas.

[4]Empathie : Capacité de se mettre à la place de l'autre et de ressentir ses sentiments et ses émotions.

Communication [XLZ00]

Comme nous l'avons vu précédemment en 3.2, le formalisme des MMDP a permis de mettre en évidence le problème clef de la coordination d'un groupe d'agents. La communication entre agents discutée dans [XLZ00] est l'un des trois moyens suggérés pour y répondre.

Une première remarque que l'on doit faire sur le terme de communication, est que l'on entend ici qu'elle est intentionnelle et directe. En effet d'autres phénomènes font apparaître une forme de communication indirecte à travers l'environnement. On peut donner l'exemple classique des phéromones déposées par des fourmis et qui leur servent de guide (voir [DG97] par exemple).

Il n'est pas évident de savoir si une communication est directe ou intentionnelle. Ainsi, si des agents "prédateurs" doivent encercler une proie, la position de chacun indique aux autres où aller. On pourrait imaginer qu'un prédateur marque clairement qu'il va aller d'un certain côté de la proie pour inciter les autres à agir en conséquence. On peut définir ici une "communication" comme étant simplement un ensemble d'interactions (actions et perceptions) mettant en relation des agents, mais sans influence sur l'environnement.

Pour en venir au travail décrit dans [XLZ00], la communication n'y est pas présentée comme une action comme une autre. Une phase de communication y est donc distinguée de la phase dite d'action. Cette communication est considérée comme coûteuse, ce qui incite à n'en user qu'avec parcimonie.

Ce cadre étant fixé, [XLZ00] ne propose pas d'apprendre par renforcement quand et quoi communiquer, mais examine deux stratégies assez opposées. Dans les deux cas, un agent transmet l'information de son état local. Mais l'une des stratégies consiste à communiquer dès qu'une révision de plan est suggérée (à cause d'une évolution peu attendue dans le suivi du dit plan). La seconde stratégie, elle, ne communique qu'en dernier recours, quand il devient indispensable de revoir le plan commun.

L'analyse des deux stratégies sur un exemple de deux agents devant se retrouver sur la même case montre surtout l'importance de la certitude des actions (actions à résultat plus ou moins hasardeux), mais ne permet pas de conclure sur "la" solution à adopter. Devant ce résultat mitigé, introduire de l'apprentissage peut sembler tentant, mais ces nouvelles actions et perceptions conduiraient assez rapidement à une explosion combinatoire déraisonnable. Il ne faut donc pas se lancer dans cette voie sans précautions.

3.4 Bilan

Utiliser l'apprentissage par renforcement pour concevoir des agents coopérants s'avère être une tâche très difficile en raison de la très grande taille de l'espace de recherche. Si des méthodes permettent de simplifier la tâche ou d'en améliorer le résultat, elles nécessitent souvent des choix ou des situations qui ne sont pas toujours possible. Il reste ainsi beaucoup à faire dans ce domaine.

Chapitre IV

Travaux développés

Ce dernier chapitre présente rapidement les différents travaux conduits jusqu'ici, ainsi que leurs possibles développements. Des articles joints permettent d'avoir une vision plus détaillée de ces travaux.

1 Point de vue adopté :

1.1 Apprentissage de politique pour un POMDP sans modèle du monde

Pour ne faire qu'un rapide rappel de ce qui a déjà été dit précédemment, le point de vue qui a été choisi pour concevoir des systèmes multi-agents par apprentissage est de laisser chaque agent apprendre son comportement de manière autonome (l'approche est donc décentralisée), tous les agents apprenant simultanément.

De plus, on s'intéresse à des agents sans modèle du monde, et n'ayant accès qu'à une observation partielle de leur environnement. Le cadre multi-agents incite fortement ces choix puisque l'on va facilement se trouver dans des environnements avec de nombreux autres congénères dont on connaît mal le comportement.

1.2 Précisions sur nos exemples

Nos travaux ont été mis en application sur différents problèmes-jeux : une tâche de fusion de blocs (détail en 2), le problème proie-prédateur (dans lequel 4 agents-prédateurs doivent encercler une proie), et le problème du tileworld ("monde des tuiles" présenté en 4.3). Ces trois problèmes ont en commun de mettre en jeu des agents :

- situés dans un environnement discrétisé (un quadrillage),
- dont les actions possibles correspondent à des déplacements (Nord-Ouest-Sud-Est), et
- dont les perceptions indiquent pour un certain nombre d'objets environnants :
  + la direction de cet objet (un point cardinal parmi 4 ou 8), et
  + éventuellement sa distance (par la seule indication "proche" si l'objet est sur l'une des huit cases qui entourent l'agent, ou "éloigné" en cas contraire).

2 Apprentissage incrémental

Un premier problème-jeu mis en œuvre pendant le DEA demandait à ce que des agents apprennent à se coordonner pour pousser des blocs l'un contre l'autre et les faire ainsi disparaître [1] (voir figure IV.1). Au fur et à mesure de la conception est apparue l'idée d'aider l'apprentissage en le rendant progressif, ou plus précisément *incrémental*.

[1] Ne pas chercher de réalisme dans cet exemple.

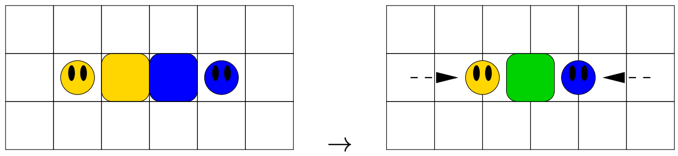


FIG. IV.1 – L'action qui donne une récompense aux agents : fusionner deux blocs.

Nous allons voir ici succinctement que le terme "incrémental" prend en fait deux aspects. Ce thème apparaît dans [1], [2] et [3]. C'est toutefois ce dernier article ([3]) qui approfondit le mieux ce sujet, utilisant une montée de gradient au lieu du Q-learning.

2.1 Complexité de la tâche

Les premières tentatives d'apprentissage s'avérant laborieuses, l'idée est apparue naturellement de commencer par des tâches simples puis de compliquer régulièrement les situations dans lesquelles se trouvent les agents, comme le montre la figure IV.2.

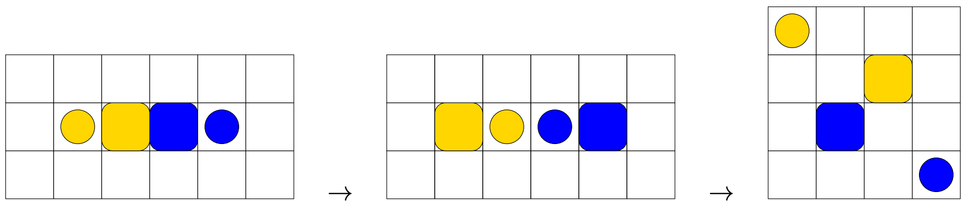


FIG. IV.2 – Exemple d'évolution dans la complexité de la tâche.

On défini un "entraînement" type à travers un script disant de quelles situations de départ lancer l'exécution, pour combien de pas de temps, et combien de fois répéter chaque situation. Cette méthode n'est pas réaliste pour tout problème, et s'avère de plus centralisatrice puisqu'une part de la méthode d'apprentissage est liée au simulateur. Néanmoins, certains cas comme celui-ci montrent qu'il n'est pas raisonnable de vouloir partir de zéro à tout prix et que l'on peut largement bénéficier d'une "supervision" extérieure.

La définition de scripts d'entraînement est un point assez lourd de cet algorithme. Il est toutefois envisageable d'y pallier en l'automatisant, en partant du principe que les premières situations sur lesquelles se concentrer sont les situations proches du but. Mais cet aspect n'a pas été développé pour l'instant et serait probablement assez délicat.

2.2 Nombre d'agents

La tâche choisie nécessite l'intervention d'au moins deux agents manipulant au moins deux cubes. Mais l'utilisation d'un plus grand nombre d'agents autour de plus nombreux cubes nous intéressait. Si les modifications de l'environnement que sont de tels ajouts d'objets seraient génantes dans un vrai MDP (ensemble d'états tout à fait différent), nos perceptions locales peuvent rester les mêmes. Le choix de perceptions *adaptées* à notre méthode est d'ailleurs crucial.

Ainsi, on a pu "cloner" des agents conçus dans un cadre `[deux agents + deux cubes]`, ajouter des cubes, et laisser les agents continuer leur apprentissage pour qu'ils s'adaptent à la nouvelle situation (le comportement optimal ne reste pas le même quand on modifie ainsi l'environnement).

Si les comportements tendent à ne plus être très structurés quand le nombre d'objets dans l'environnement augmente, il est assez satisfaisant de voir que l'architecture des agents leur permet cette souplesse d'utilisation. Seuls des cas avec plus de cubes que d'agents ont posé des problèmes sérieux, certaines "situations de jeu" étant bloquantes.

3 Intérêt / Attention persistante

Les “situations de jeu” bloquantes dont nous venons de parler étaient en général provoquées par un défaut d’attention. En effet, la localité des perceptions a pour effet que, d’un instant à l’autre, les objets perçus par l’agent ne sont plus les mêmes, ce qui peut être assimilé à un phénomène de perte d’attention. Cette thématique a donné lieu à quelques travaux qui mériteraient d’être approfondis et dont voici une description.

3.1 01-interet

Comme il faut avoir une attention mesurée, c’est à dire équilibrée entre “changer d’idée à chaque instant” et “toujours se focaliser sur le même problème”, l’idée est venue d’essayer d’apprendre quand changer de cibles (d’objets perçus).

Quelques essais furent ainsi menés, soit en ajoutant aux actions de déplacement habituelles une action *changer de cible*, soit en ajoutant au contrôleur de base appris par l’agent (celui qui gère donc les déplacements) un contrôleur d’attention qui apprend quand changer de cible. Une remarque importante est qu’ici, quand on *change de cible*, c’est pour choisir la cible `[1 agent et 2 cubes]` la plus proche. Les quelques expérimentations menées ne furent pas très concluantes, et ce n’est que plus tard avec un projet d’élèves que le thème de l’attention a été repris sous une autre forme.

3.2 projet-Esiaux

Le projet d’initiation à la recherche proposé part du même principe que nos agents peuvent nettement bénéficier d’une persistance de leur attention, mais prend une forme différente.

Le principe est ici aussi d’ajouter un contrôleur de l’attention, mais dont le choix de la nouvelle cible est fait en fonction de son *utilité*. En effet à chaque cible possible est associée une perception, et l’agent sait de par son apprentissage associer une utilité à chaque perception. On peut ainsi choisir la nouvelle cible en donnant plus de chances aux cibles de plus grandes utilités.

Dans les développements de ce projets, l’intervalle de temps entre changements de cibles était fixé par l’expérimentateur. Il s’est avéré que les meilleurs résultats sont obtenus en remettant en cause la cible à chaque pas de temps (sur le problème choisi du moins).

Cette thématique offre diverses possibilités de développements et d’études encore à mener. Une remarque que l’on peut faire sur cette notion d’attention est qu’elle se rapproche de l’idée de mémoire contextuelle, et à des travaux tels que ceux de [Dut00] et [McC95a].

4 Décomposition d’un comportement en comportements élémentaires

Comme on vient de l’évoquer en décrivant le projet d’élèves, choisir la “cible” de l’attention de l’agent en fonction d’une mesure d’utilité s’avère être une bonne heuristique utilisable avec tout POMDP tel que le nôtre. Mais cette mesure peut servir à de plus vastes fins qu’un simple choix entre deux cibles potentielles, comme le montrent les travaux en cours présentés ci-dessous.

4.1 La problématique

En fait, un problème qui nous est apparu est qu’un agent (tel qu’un robot mobile) se retrouve facilement face à un environnement complexe, donc posant de très grandes difficultés, alors que cette complexité n’est souvent que le résultat d’une combinaison de situations relativement simples.

L’idée est alors venue de chercher dans la *scène* perçue par l’agent quelles sont les situations simples qui la composent, quels comportements de base ces situations appellent, et enfin comment recombiner ces comportements en un comportement “complexe”.

4.2 Méthode proposée

Un premier point, loin d'être résolu, est pour l'agent de trouver quelles motivations de base le guident dans son monde, et donc quels comportements de base distinguer. Pour l'instant, cette tâche est celle de l'humain qui définit ces comportements de base à la fois par les récompenses et le type de configurations [2] associés. Ceci permet d'apprendre à la fois les politiques associées à ces comportements par la méthode de montée de gradient, et les tables de Q-valeurs associées (dont nous verrons l'usage ci-après).

L'agent étant ainsi préparé, il lui faut trouver dans la scène qu'il perçoit les configurations correspondant à ses comportements possibles. Un point délicat est alors de recombiner les politiques utiles. Des combinaisons linéaires ont été essayées (voir [4], [5] et surtout [6]), en donnant aux politiques des pondérations dépendant des Q-valeurs (qui représentent une espérance de gain), et de coefficients appris (un par comportement). On obtient d'assez bon résultats sur le problème-jeu présenté ci-après (section 4.3).

4.3 Exemple du monde des tuiles

Nous allons illustrer la méthode proposée à l'aide du "tileworld" (monde des tuiles), problème-jeu dont on rencontre un certain nombre de variantes. Dans notre cas, il s'agit d'un environnement bidimensionnel quadrillé dans lequel se déplace un agent (voire plusieurs au besoin) et se trouvent des tuiles et des trous. Le but de l'agent est de pousser les tuiles dans les trous tout en évitant de tomber dans ces derniers.

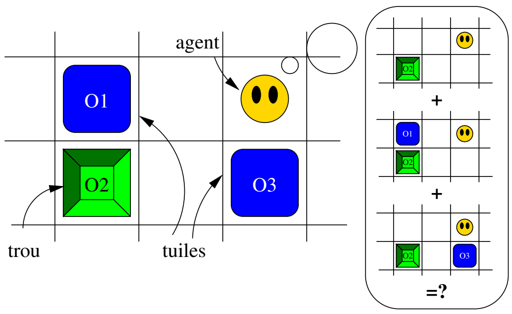


FIG. IV.3 – Ici, un agent dans une situation "complexe" décomposable en 3 configurations simples.

La figure IV.3 montre un cas dans lequel l'agent peut reconnaître trois configurations, dont une incite à éviter le trou perçu, et les deux autres suggèrent de pousser une tuile dans ce trou. On a ainsi trois comportements possibles, associées chacune à quatre probabilités (définissant la politique à suivre) et quatre Q-valeurs (une probabilité et une Q-valeur pour chaque action, c'est à dire pour chaque direction de déplacement possible).

Les différentes méthodes "additives" possibles pour recombiner les comportements de base en un comportement complexe visent en général à calculer quatre nouvelles probabilités de choix des actions possibles. Sans entrer dans le détail, la probabilité d'aller au Nord peut être la somme des probabilités de choisir cette même action (du point de vue de chacun des trois comportements de base), pondérées par l'exponentiel de la Q-valeur associée :

$$P(Nord) = \sum_{c \in Comportements} e^{Q_c(Nord)} * P_c(Nord)$$

4.4 Travaux à venir

Une approche multiplicative et non plus additive a aussi été suivie, donnant avec moins d'efforts de mise au point un comportement complexe plus efficace. Cela nous a incité à distinguer les relations possibles entre comportements : motivations incompatibles (s'excluant mutuellement), additives... Les travaux à venir s'orientent dans cette direction. Si cette problématique n'est pas précisément multi-agents, elle

[2] On appelle *configuration* un ensemble d'objets de la scène, comme on parlait auparavant de *cibles*.

se justifie clairement tant les environnements “complexes” de cette forme sont courants en présence de groupes d’agents. De plus, l’utilisation de configurations pourrait permettre une communication efficace entre agents (par désignation).

Publications

[1] O. Buffet. Apprentissage par renforcement dans un système multi-agents. Master's thesis, Université Henri Poincaré (Nancy), 2000. Mémoire de DEA.

[2] O. Buffet, A. Dutech, and F. Charpillet. Incremental reinforcement learning for designing multi-agent systems. In *Proceedings of the fifth international conference on Autonomous agents (Agents'01)*, pages 31–31, 2001.

[3] A. Dutech, O. Buffet, and F. Charpillet. Multi-agent systems by incremental gradient reinforcement learning. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJ-CAI'01)*, 2001.

[4] O. Buffet and A. Dutech. Looking for scalable agents. In *Proceedings of the fifth European Workshop on Reinforcement Learning (EWRL-5)*, 2001.

[5] O. Buffet, A. Dutech, and F. Charpillet. Learning to weigh basic behaviors in scalable agents. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'02)*, 2002. [poster session].

[6] O. Buffet, A. Dutech, and F. Charpillet. Adaptive combination of behaviors in an agent. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI'02)*, 2002.

Bibliographie

[BB01] J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15 :319–350, 2001.

[BBW01] J. Baxter, P. Bartlett, and Lex Weaver. Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15 :351–381, 2001.

[Bel57] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, New-Jersey, 1957.

[Bou96] C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge (TARK '96), De Zeeuwse Stromen, The Netherlands*, 1996.

[Bou99] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99), Stockholm, Sweden*, 1999.

[BZI00] D.S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'00), Stanford, California*, 2000.

[CSC02] I. Chadès, B. Scherrer, and F. Charpillet. A heuristic approach for solving decentralized-pomdp : Assessment on the pursuit problem. In *Proceedings of the seventeenth ACM Symposium on Applied Computing (SAC-2002)*, 2002.

[DG97] Marco Dorigo and Luca Maria Gambardella. Ant colonies for the traveling salesman problem. *BioSystems*, 43 :73–81, 1997.

[Die00] T.G. Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of Artificial Intelligence Research*, 13 :227–303, 2000.

[Dut99] A. Dutech. *Apprentissage d'environnement : approches cognitives et comportementales*. PhD thesis, ENSAE, Toulouse, 1999.

[Dut00] A. Dutech. Solving pomdp using selected past-events. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI'00)*, 2000.

[HW98a] J. Hu and M. Wellman. Multiagent reinforcement learning : theoretical framework and an algorithm. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 242–250, 1998.

[HW98b] J. Hu and M. Wellman. Online learning about other agents in a dynamic multiagent system. In *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 239–246, 1998.

[JJS94] T. Jaakkola, M. Jordan, and S. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6(6) :1186–1201, 1994.

[JSW98] N.R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1 :7–38, 1998.

[KLM96] L. Kaelbling, M. Littman, and A. Moore. Reinforcement learning : A survey. *Journal of Artificial Intelligence Research*, 4 :237–285, 1996.

[Lit94] M. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML'94), San Fransisco, CA*, 1994.

[MA95] A. Moore and C. Atkeson. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state space. *Machine Learning*, 21, 1995.

[Mat97] M. Matarić. Reinforcement learning in the multi-robot domain. *Autonomous Robots*, 4(1) :73–83, 1997.

[McC95a] R. A. McCallum. Instance-based utile distinctions for reinforcement learning with hidden state. In *Proceedings of the 12fth International Machine Learning Conference (ML'95)*, 1995.

[McC95b] R. A. McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, University of Rochester, 1995.

[MD00] N. Meuleau and M. Dorigo. Ant colony optimization and stochastic gradient descent. Technical Report IRIDIA/2000-36, IRIDIA, Universite Libre de Bruxelles, Belgium, 2000.

[MM02] R. Munos and A. Moore. Variable resolution discretization in optimal control. *Machine Learning*, 2002.

[MPR+98] A. McGovern, D. Precup, B. Ravindran, S. Singh, and R. Sutton. Hierarchical optimal control of mdps. In *Proceedings of the 10th Yale Workshop on Adaptive and Learning systems*, 1998.

[Owe82] G. Owen. *Game Theory : Second Edition.* Academic Press, Orlando, Florida, 1982.

[Put94] M. L. Puterman. *Markov Decision Processes–Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, USA, 1994.

[RN95] S. Russell and P. Norvig. *Artificial Intelligence : A Modern Approach*. Englewood Cliffs, NJ : prentice Hall, 1995.

[SB98] R. Sutton and G. Barto. *Reinforcement Learning : an introduction*. Bradford Book, MIT Press, Cambridge, MA, 1998.

[SJJ94] S. Singh, T. Jaakkola, and M. Jordan. Learning without state estimation in partially observable markovian decision processes. In *Proceedings of the 11th International Conference on Machine Learning (ICML'94)*, 1994.

[TW00] K. Tumer and D. Wolpert. Collective intelligence and braess' paradox. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence, Austin, TX*, 2000.

[WWT99] D. Wolpert, K. Wheeler, and K. Tumer. General principles of learning-based multi-agent systems. In *Proceedings of the 3rd International Conference on Autonomous Agents (Agents'99), Seattle*, pages 77–83, 1999.

[XLZ00] P. Xuan, V. Lesser, and S. Zilberstein. Communication in multi-agent markov decision processes. In *Proceedings of ICMAS Workshop on Game Theoretic and Decision Theoretic Agents, Boston, MA*, 2000.

Annexe A

Rapport sur le Comité de Thèse d'Olivier Buffet (par Alain Dutech)

Avec les participations actives de :
- Frédéric Alexandre, DR INRIA, Equipe Cortex, LORIA, Nancy
- François Charpillet, DR INRIA, Equipe Maia, LORIA, Nancy
- Alain Dutech CR INRIA, Equipe Maia, LORIA, Nancy
- Frédéric Garcia, CR à l'INRA, Dept. BIA, Toulouse
- Manuel Samuelides, Professeur à l'ENSAE, Toulouse
- Olivier Sigaud, MdC LIP6, Paris

(Claude Kirchner, DR INRIA au Loria, ayant été excusé).

Quand c'est possible, des remarques liées à des points précis de la présentation d'Olivier sont associées au numéro du transparent concerné entre parenthèses, comme ceci : (n) [1].

1 Remarques générales

Après un an et demi de thèse, la première impression qui se dégage du travail d'Olivier est globalement très positive. D'une part, l'orientation donnée par Olivier à ses recherches est intéressante et amène à des résultats expérimentaux conséquents, d'autre part ces résultats sont validés et appuyés par des publications de très bon niveau. On ne peut apprécier le sérieux du travail, aussi bien en qualité qu'en quantité, ce qui se traduit par une bonne connaissance et une bonne appréhension du domaine étudié.

Cette connaissance n'est pas toujours bien mise en valeur dans le rapport et la présentation faite par Olivier. En particulier, et nous y reviendrons par la suite, le choix fait de présenter l'apprentissage par renforcement avant les systèmes multi-agents ne semble pas pertinent. C'est en effet un défi que de vouloir présenter d'une manière concise et compréhensible un sujet aussi vaste que l'apprentissage par renforcement. Des raccourcis et des imprécisions sont alors inévitables. L'approche suggérée est de d'abord s'intéresser aux systèmes multi-agents, et plus précisément au sous-ensemble de ces systèmes qui seront étudiés dans la thèse pour en définir les caractéristiques, ce qui permettra par la suite de restreindre l'étude sur l'apprentissage par renforcement et de la rendre plus adaptée et plus claire.

D'ailleurs, si le travail d'Olivier peut se prolonger dans plusieurs directions différentes, il faudra sans doute n'en choisir qu'une pour être à même de l'explorer le plus complètement possible. Parmi les directions possibles, à savoir,
- **Politiques sans-mémoire et POMDP** : outre des questions sur la mesure et l'optimalité (possible ou non) de telles politiques, l'intérêt se porterait aussi et surtout sur les algorithmes pouvant être mis en

[1] Note d'Olivier : Hélas, les transparents sont séparés de ce document

œuvre (gradient, Q-Learning modifié, Monte-Carlo, etc). Une avancée théorique étant évidemment un plus significatif.

– **Apprentissage incrémental** : Replacer les travaux d'Olivier dans la/les communauté(s) et s'inspirer d'autres travaux pour aller encore plus loin. La question essentielle qui se pose ici est de savoir comment un agent peut lui-même organiser (pour lui ou pour un autre) les "étapes" de l'apprentissage. Quels seraient les liens avec la psychologie développementale, l'apprentissage par imitation, etc...
– **Combinaison de comportements "primaires"** : il faut là aussi mieux situer ces travaux dans la communauté, il peut en particulier s'avérer important de les comparer (théoriquement et expérimentalement) avec des travaux similaires. Le point clé de cette direction de recherche est la façon dont sont combinées les actions des différentes politiques primaires, et les travaux préliminaires d'Olivier en ce sont sont plus qu'encourageants. Une autre question essentielle, mais qui peut éventuellement s'avérer trop complexe et avec trop de ramification pour être sérieusement abordée dans cette deuxième moitié de thèse, concerne la détection et l'élaboration par l'agent de ces comportements primaires.

il semble que les travaux sur la combinaison de comportements sont les plus originaux et les plus intéressants. C'est donc dans cette direction qu'Olivier est encouragé à continuer.

La suite de ce rapport apporte des remarques et des suggestions plus détaillées et plus spécifiques, notamment en ce qui concerne l'apprentissage par renforcement, les systèmes multi-agents, l'apprentissage incrémental et la combinaison de comportements.

2 Apprentissage

Malgré un effort conséquent pour présenter le principe de l'apprentissage par renforcement (RL) par le biais du formalisme des Processus Décisionnels de Markov (MDP) en long et en large, cette partie manque en fait de profondeur, parfois de rigueur, et finalement ne fait pas entièrement le tour de la question. C'est bien normal puisque cette partie pourrait faire l'objet d'une thèse à part entière. La proposition mentionnée auparavant de présenter d'abord les SMA puis les aspects du RL adaptés à la problématique devrait résoudre ce problème en grande partie. C'est d'autant plus vrai qu'Olivier a montré qu'il avait suffisamment de recul et de réflexion sur ce domaine de l'apprentissage par renforcement.

Pour aller plus dans les détails, certains points sont cependant à revoir.

Pour illustrer les formalismes et les algorithmes, il serait sans doute plus judicieux de choisir un exemple et de le garder, éventuellement en le développant, tout au long de cette partie sur RL. En plus, l'exemple du feu rouge (12) n'a pas vraiment convaincu.

En ce qui concerne le Q-Learning, il faut sans doute expliquer que dans sa version classique il n'y a pas "co-évolution" mais au contraire d'abord une phase d'exploration (à l'aide d'une stratégie) puis ensuite seulement exploitation. C'est dans la version "Boltzmannienne" du Q-Learning, qu'il faudrait sans doute expliciter un peu plus, que cette notion de co-évolution est présente. Comme la co-évolution, ainsi que les politiques stochastiques, sont d'un intérêt particulier quand on veut traiter un POMDP comme si c'était un MDP, ces notions seront plus faciles à développer une fois le cadre applicatif (c-à-d le SMA) bien exposé. Enfin, mais ce n'est pas forcément possible de tout faire, il serait intéressant de tester le Q-Learning de Boltzmann sur un exemple simple pour mieux comprendre son comportement, l'influence des paramètres, de la décroissance de la température, de T_{min}, etc.

L'algorithme du gradient de Baxter a lui aussi suscité intérêt et commentaires (oscillation, extremum locaux...), prouvant qu'il pourrait être un sujet de thèse à lui tout seul. D'une part, il faudrait l'expliquer avec plus de détails et, comme avec le Q-Learning de Boltzmann, essayer de mieux le cerner en le testant sur des exemples plus élémentaires. Il n'a pas été facile de répondre à la question : "Pourquoi *ces* deux algorithmes ?". Et pourquoi ne pas utiliser REINFORCE ou un algorithme inspiré qui serait plus simple à maîtriser et comprendre ?

Il reste enfin quelques détails :

– La figure du transparent (9) n'est pas très conventionnelle, elle est alors un peu difficile à appréhender.
– La différence entre la planification et l'apprentissage par renforcement (10), il semble préférable de dire qu'il y a RL quand on s'appuie sur l'expérience.

3 Systèmes Multi-Agents (SMA)

Dans le manuscrit final, cette partie pourrait trouver sa place avant de parler d'apprentissage par renforcement afin de restreindre le champ d'étude de ce dernier. Dans cette optique, il faudrait s'attacher à décrire quelle partie des SMA sera étudiée et de préciser s'il y a compétition, coordination, collaboration, etc. Le problème de la récompense du système (globale, décentralisée, accessible à un agent ou nécessitant un agent observateur) est crucial et devra être abordé avec plus de rigueur. Une question qui a été soulevée concerne la prise en compte d'agents qui pourraient se sacrifier au groupe, et qui ne maximiseraient pas une récompense locale “naïve”.

D'un point de vue bibliographique, il ne faut pas oublier que les économistes étudient depuis assez longtemps des domaines qui paraissent connexes : jeux markoviens, équilibre de Nash, économie évolutionnaire.

Une remarque concerne plus spécifiquement le modèle choisi, c'est-à-dire un ensemble d'agents homogènes qui peuvent néanmoins apprendre des politiques différentes en fonction de leur expérience. Dans ce cadre, pourquoi ne pas étudier ce qu'apporte l'échange de politiques, ou de morceaux de politiques, entre les agents ? En tout cas, cela pose des questions intéressantes pour savoir si une politique est efficace en tant que telle ou plutôt parce qu'elle est utilisée conjointement à d'autre politiques (les autres agents).

En ce qui concerne le modèle sous jacent qui correspond au problème posé dans le cadre de l'apprentissage (transparent 16), les causes de la perte de stationnarité du problème de décision sont à revoir. Elle est causée en fait, mea culpa, uniquement par les autres agents qui changent eux-aussi leur politique.

Dans le domaine des détails :

- Transparent (17), ne pas mettre la communication et la mémoire sur le même plan. La mémoire est en fait une mémoire à l'exécution, à différencier de la mémoire utilisée pour mémoriser la politique.

4 Méthode incrémentale

En ce qui concerne cette partie, trois aspects ont été soulevés. Le premier concerne les méthodes existantes dont il semble que ni le rapport ni la présentation ne fassent mention. En particulier, le problème de l'apprentissage avec aide a déjà fait l'objet de plusieurs études. Olivier Sigaud a mentionné quelques travaux sur “apprendre à faire du vélo”, l'apprentissage progressif de Mataric. Plus récemment, Arnaud Revel de Cergy a évoqué des travaux en collaboration avec une psychologue sur une approche développementale de l'apprentissage (revel@ensea.fr).

D'autre part, le contexte des simulations devrait être mieux explicité. Par exemple, est-ce que tous les agents jaunes sont initialisés avec une politique d'agent jaune ou reprend-on une et une seule politique ? Il faut aussi mieux expliquer pourquoi, quand de nombreux agents sont présents, ils ont besoin de “moins” apprendre pour arriver aux mêmes performances (effet de saturation de l'environnement).

Enfin, les perceptions des agents ont soulevé de nombreux commentaires. Dans l'état actuel elles ne paraissent pas plausibles, ou en tout cas pas exactement locales. Est-ce vrai ? Et si on utilisait une perception plus locale (au moins spatialement) quelle serait l'influence sur les performances du système. Plus généralement, quel est le biais induit par les perceptions choisies ?

Dans le domaine des détails et des questions plus spécifiques :

- Les courbes (25) devraient plutôt être des moyennes car le “lissage” est trompeur et pas forcément maîtrisé.
- Dans les courbes de (28), est-ce un hasard si la fin de la courbe “8a8c from scratch” est à la même valeur que le début de la courbe “8a8c from 2a2c” ou est-ce explicable ?
- Transparent (30) : il ne faut pas forcément associer déterministe et Q-Learning, ni stochastique et gradient.

5 Décomposition du comportement, de la scène

Les travaux connexes dans le domaines sont assez nombreux, et l'étude bibliographique qui en a été faite n'apparaît pas vraiment ni dans le rapport ni dans la présentation. Elle gagnerait à être complétée par la lecture des travaux de Wiering, Parr, Sun, Preccup, Toby Tyrrell, JP Muller. Une synthèse/analyse de l'état de l'art sous la forme d'une "grille de lecture" serait un atout considérable.

L'apport essentiel de cette partie concerne évidemment la combinaison de plusieurs politiques en une seule, plus complexe. Les différentes méthodes employées (voir (48) et (51)) devraient être plus explicitées, notamment en ce qui concerne ce qu'elles apportent et ce qu'on en attendait. Plus spécifiquement la formule additive (51), outre un petit problème de normalisation, pose un problème pour les actions liées à un fort Q négatif (et donc une action à éviter) qui va ainsi renforcer la probabilité d'une action qui conduit à une récompense négative. Même si en fait cette probabilité est proche de 0, il paraît bizarre, intuitivement, de la renforcer et donc Olivier devra mieux montrer en quoi ces formulations sont pertinentes. Enfin, plus globalement et comme cela est indiqué dans les perspectives, un travail important et fort intéressant est sans doute à faire sur l'étude des différentes façons de combiner des comportements.

Pour rester dans le même cadre, il semble qu'il existe des travaux connexes en théorie des jeux, sur des optimisation multi-critères que l'on pourrait aussi voir comme une combinaison de plusieurs influence. Cela concerne les "fronts de Pareto" et cette piste, bien qu'assez floue, reste à explorer.

Dans le domaine des détails et des questions plus spécifiques :

- Il faut tester les nouvelles formules (51) sur le problème proie-prédateurs.
- Est-il possible de trouver la "vraie" solution optimale pour le problème évoqué lors du transparent (52) ?