Introduction à la recherche : l' Apprentissage par Renforcement Relationnel

Cédric HERPSON

Janvier 2008

Table des matières

Introduction

L'apprentissage par renforcement relationnel (ARR) a émergé comme une alternative aux limites de l'apprentissage par renforcement (AR). Alors que la représentation de l'espace d'état est classiquement basée sur une représentation propositionnelle, l'ARR offre une représentation plus riche de ce dernier par sa description sous forme relationnelle.

La combinaison de la programmation logique inductive et de l'apprentissage par renforcement offre ainsi la possibilité de s'abstraire du but poursuivi à l'instant courant. Elle permet par conséquent d'utiliser des algorithmes s'appuyant sur la structure relationnelle de l'espace d'état pour généraliser les résultats appris.

Avant de présenter les méthodes et avantages associés à cette approche ainsi que ses propres limites, il nous a tout d'abord semblé nécessaire de rappeller les raisons de son développement par un rapide retour sur l'apprentissage par renforcement et les idées fondamentales associées.

Chapitre 1

Apprentissage par Renforcement

1.1 Principe général

L'idée maitresse de l'AR est d'apprendre une politique visant à maximiser la récompense globale de l'agent vis à vis d'un but à atteindre en considérant uniquement la récompense locale associée à chaque action. Chaque décision prise influant sur les décisions suivantes, l'apprentissage par renforcement peut être vu comme une méthode permettant de résoudre un problème de décision séquentielle.

Les Processus Décisionnels de Markov (PDM) définissent le cadre formel de l'apprentissage par renforcement.L' univers y est fini, stochastique, totalement observable et discret.

Processus Décisionnels Markoviens

Plus formellement, un processus de décision Markovien est défini par :

- S, un ensemble fini d'états. s $\in$ S
- A, un ensemble fini d'actions pour l'état s. a $\in$ A(s)
- r, une fonction de récompense. r(s,a) $\in \Re$
- P, la probabilité de transition d'un état à un autre en fonction de l'action sélectionnée. $P(s'|s,a) = P_a(s,s')$

Le fait de se placer dans un univers où l'on considère que l'état suivant ne dépend que de l'état courant et de l'action retenue revient à se placer dans un environnement Markovien. Cette hypothese est communément appelée la "propriété de Markov".

1.1.1 Deux grandes approches

Les algorithmes d'apprentissage par renforcement peuvent être catégorisés selon deux axes :
- L'apprentissage de la fonction d'utilité liée au états.
- L'apprentissage de la fonction d'utilité liée aux actions.

Dans le premier cas, il est nécéssaire de disposer d'un niveau d'information élevé sur l'environnement ou l'on évolue. L'action sélectionnée dépend de l'utilité associée à l'état supposé atteint par l'exécution de la-dite action (algorithme TD learning).
Dans le second cas, seule la récompense associée à l'action est nécéssaire. Notre connaissance de l'environnement et des probabilités de transitions entre les états peut donc être plus parcélaire.

L'algorithme Q-Learning appartenant à cette dernière classe est l'un des plus couramment utilisé aujourd'hui.

1.1.2 Un algorithme type : Q-learning

$Q(\vec{s}, a) \leftarrow \vec{0}$
$s_0 \leftarrow$etat initial
$t \leftarrow 0$
repeter
  choisir l'action a_t
  observer r_t et s_{t+1}
  $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma max_{a \in A(s_{t+1})} Q(s_{t+1}, a) - Q(s_t, a_t)]$
  $t \leftarrow t + 1$
jusqu'a $s_t \in Etatsfinaux$

Cet algorithme permet à l'agent d'apprendre de manière continue en interagissant avec l'environnement. La première étape dite de "sélection de l'action" est un compromis entre l'utilisation des connaissances acquises (on choisit l'action qui maximise la récompense) et l'exploration de l'espace d'état en optant pour une action à priori moins intéressante en terme de récompense.

L'un des principaux intérets de cet algorithme est qu'il peut être exprimé comme une reformulation de l'algorithme d'itération de la valeur. Or il a été prouvé que ce dernier, pour un PDM fixé, converge vers une politique optimale Q^* sous réserve qu'il vérifie les conditions suivantes :

1. Chaque paire (s,a) est visitée une infinité de fois (avec S et A finis).
2. $\gamma < 1$
3. $\sum_t \alpha_t(s) = +\infty$ et $\sum_t \alpha_t^2(s) < +\infty \forall s \in S$

Malgré sa forte utilisation et les avantages évoqués ci-dessus, l'algorithme de Q-learning n'est pas exempt d'inconvénients. Ceux-ci, présentés en partie 1.3 page 4, sont représentatifs des limitations actuelles de l'apprentissage par renforcement.

1.2 Introduction au monde des blocs

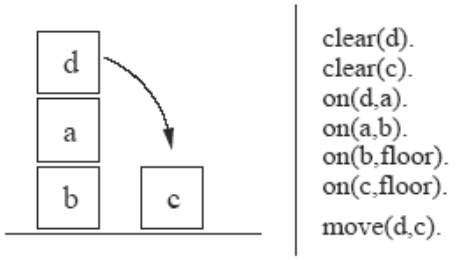

FIG. 1.1 – Exemple d'état dans le monde des blocs

Le monde des blocs est un exemple couramment utilisé pour illustrer les limites de l' apprentissage par renforcement et introduire l' apprentissage par renforcement relationnel et ses spécificités. Les états et actions de ce monde sont en effet tres facilement représentable par les relations qui existent entre les objets qui le compose. Dans ce monde, les blocs peuvent être sur le sol ou les uns sur les autres.
Les prédicats existants pour décrire l'état du monde sont *clear(X)* et *on(A,B)*,ils renvoient respectivement "'vrai"' si aucun cube n'est posé sur X et si A est bien sur B. La seule action possible est *move(Z,B)*. Chaque état peut donc être représenté part un ensemble de faits "'vrai"' dont la taille varie en fonction de l'état courant. Un exemple d'état pour un univers à 4 blocs est présenté sur la figure 1.1. Cet univers sera utilisé comme support dans la suite de ce document.

1.3 Les limites de l'apprentissage par renforcement

1.3.1 Une représentation tabulaire

Les principaux algorithmes d'AR dont le Q-learning présenté en 1.1.2 ont été développés pour travailler avec une représentation tabulaire de l'espace d'états (Matrice état/action pour le Q-learning). Celle-ci fait émerger deux problèmes : l'espace mémoire nécessaire pour représenter l'espace de recherche, et le temps nécessaire au remplissage de ce dernier. Si ces deux limitations sont supportables pour des univers à faible dimension et/ou complexité, il n'est pas possible d'utiliser en l'état ces algorithmes pour des espaces de recherches importants (voire infini).

Bien que l'association aux algorithmes d'AR classiques d'un réseau neuronal (TD-Gammon) ou d'un arbre de décision basé sur une structure propositionnelle ai en partie put résoudre le problème de la taille de l'espace d'état pour une tache d'apprentissage donnée, les limites du langage propositionnel n'ont pas permis de résoudre le problème de la généralisation entre différentes tâches d'apprentissage

1.3.2 Une généralisation à des problèmes proches impossible

L'exemple du monde des blocs dont un état possible apparait sur la figure 1.1 est représentatif des incovénients d'une structure reposant sur un langage propositionnel. Si chaque état possible s peut être décrit par une conjonction de proposition (ici, s={*clear(c)*∧ *clear(d)*∧ *on(d, a)*∧ *on(a, b)*∧ *on(b, floor)*∧*on(c, floor)*}), l'absence de variables nécéssite d' exécuter à nouveau l'algorithme lorsque l'on change le but à atteindre.
L'apprentissage du prédicat *on(a,b)* en lieu et place de *on(b,c)* requiert ainsi, malgrés une variation minime dans la structure du problème, le réapprentissage de la fonction Q.

1.3.3 Une représentation pour un modéle stochastique

L'apprentissage par renforcement repose sur le cadre théorique des processus de décisions markoviens. Si cette approche permet d'obtenir de bons résultats dans des environnement représentables de manière numérique et/ou probabiliste, de nombreux problèmes auxquels nous sommes confrontés ne se prêtent pas, ou difficilement à ce type de représentation (ex : langage naturel). Ces problèmes se représentent plus «facilement» par les relations qui lient les différents objets.

Or, ce que nous connaissons du fonctionnement humain nous laisse à penser que ce type de représentation, plus riche, offre de meilleures possibilités d'apprentissage et facilite la généralisation et/ou l'analogie.

L ' ARR a pour but de s'affranchir des limites de l'AR en basant la représentation de l'espace de recherche du problème concerné sur sa structure relationnelle.

Chapitre 2

Apprentissage par Renforcement Relationnel

2.1 Principe général

L'introduction de la logique du premier ordre (et donc de variables) rend, contrairement à l'apprentissage par renforcement propositionnel, théoriquement possible l'abstraction du but courant pour en déduire une propriété générale à même d'etre réutilisée dans un problème proche. Par sa capacité d'apprentissage de lois générales, cette approche permet également d'étendre des connaissances simples à des mondes plus complexes, réutilisant ainsi les connaissaces préalablement acquises et offrant dans le même temps la possibilité de capitaliser les acquis pour s'en servir comme élements de solutions.

2.2 Une approche descendante

Le premier objectif a été de montrer que l' ARR permettait bien de généraliser. Dzeroski et Driessen ont, les premiers, combiné l'algorithme d'apprentissage par renforcement Q-learning présenté en 1.1.2 avec une représentation relationnelle du problème en s'appuyant sur un arbre logique de décision afin d'obtenir une approximation de la fonction Q (Q-RRL).

2.2.1 Arbres de décisions et arbres logiques de décisions

Le terme "arbre de décision" est classiquement associé aux taches de classification et de régression. On partitionne l'espace d'état en travaillant sur la valeur des attributs des exemples (principe de l'algorithme ID3 de Quinlan) afin de discriminer ces derniers selon un critère donné, généralement l'appartenance à une classe.

La différence entre les arbres de décisions et les arbres logiques de décision est que pour ces derniers, les exemples considérés en entrée de l'arbre correspondent à la description d'un état du monde. Cette caractéristique, couplée à la plus grande expressivité de la logique du premier ordre, permet d'employer sur chaque noeud de décision de l'arbre des prédicats possédants des variables.

Les figures 2.1 et 2.2 montrent pour ces deux types d'arbres la manière d'aborder le même problème. Dans le cas présent l'objectif est de savoir, dans un univers réduit à 3 blocs, si au moins 1 bloc est empilé sur un autre.

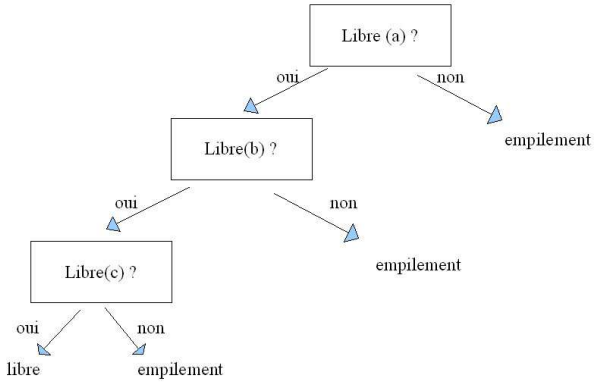

FIG. 2.1 – Exemple d'apprentissage avec un arbre de décision classique

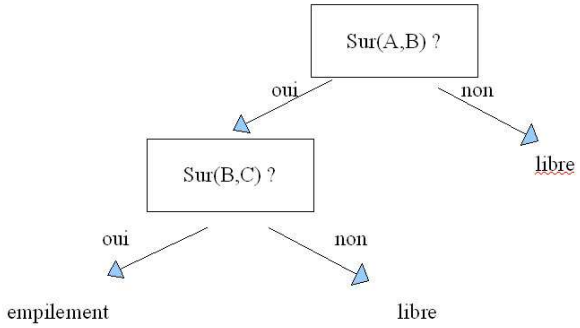

FIG. 2.2 – Exemple d'apprentissage par un arbre logique de décision

Alors que dans la figure 2.1 l'arbre obtenu indique explicitement l'ordre dans lequel il procéde aux différents tests, l'arbre logique de décision de la figure 2.2 reste indépendant des objets considérés (seul le nombre d'objets de l'univers a influé sur la structure de ce dernier). Si l'on change simplement les nom des 3 cubes concernés, l'arbre de décision "classique" devra, contrairement à l'arbre logique de décision, être réappris.

On remarquera également dans le cas de l'arbre logique de décision qu'une même variable (B) peut être partagée entre différents noeuds de profondeur différentes.

Les algorithmes TILDE et TILDE-RT

TILDE [1] est le pendant de l'algorithme de classification C4.5 de Quinlan pour les arbres logiques de décision. Il emprunte aussi bien à ce dernier la méthode de sélection des différents tests à associer aux noeuds internes de l'arbre que son mécanisme d'élagage. Le principal point qui différencie TILDE et TILDE-RT (son extension aux taches de régression) des arbres de décision "classiques" porte sur le fait que l'évaluation d'un prédicat sur un noeud peut dépendre, du fait de l'utilisation de variables, de prédicats présents en amont de l'arbre (comme le montre l'exemple de la figure 2.2 et de la variable partagée B).

Un exemple d'arbre obtenu par l'utilisation de l'algorithme TILDE-RT est présenté en page 8 conjointement avec l'exemple d'exécution de l'algorithme Q-RRL[2] qui l'utilise pour représenter la fonction Q.

[1] Top down Induction of Logical DEcision trees
[2] Relationnal Reinforcement Learning

2.2.2 L'exemple de Q-RRL

L'algorithme Q-RRL a été obtenu en associant l'algorithme d'apprentissage par renforcement Q-learning et l'algorithme TILDE-RT. Au lieu de travailler explicitement avec une matrice contenant les différentes valeurs de la fonction Q, celle-ci est implicitement représentée par le "Q-arbre" obtenu par l'algorithme TILDE.

Algorithme Q-RRL

Initialize $\hat{Q}_0$ to assign 0 to all (s,a) pairs
Initialize Examples to the empty set.
e :=0
do forever
    e :=e+1
    i :=0
    generate a random state s0
    while not goal(s_i) do
        select an action a_i stochastically
            using the Q-exploration strategy
            using the current hypothesis for $\hat{Q}_e$
        perform action a_i
        receive an immediate reward $r_i := r(s_i, a_i)$
        observe the new state s_{i+1}
        i :=i+1
    endwhile
    for j=i-1 to 0 do
        generate example $x = (s_j, a_j, \hat{q_j})$
            where $\hat{q_j} := r_j + \gamma max_{a'} \hat{Q}_e(s_{j+1}, a')$
        if an example $(s_j, a_j, \hat{Q_{old}})$ exists in Examples, replace it with x,
            else add x to Examples.
        update $\hat{Q}_e$ using TILDE-RT to produce $\hat{Q_{e+1}}$ using Examples.
    endfor
enddo

Le principal changement par rapport à l'algorithme Q-learning se situe dans la boucle "for", où la mise à jour de la fonction $\hat{Q}$ est effectuée. Celle-ci étant maintenant réalisée par l'algorithme TILDE-RT introduit en 2.2.1, il faut donc construire pour chaque action réalisée entre l'instant initial et l'instant final (arrivée sur l'état but) le vecteur exemple associé. Une fois la base d'exemple constituée, l'algorithme TILDE-RT peut mettre à jour le Q-arbre.

Afin de mieux visualiser le fonctionnement de cet algorithme, un exemple simple d'exécution de celui-ci est présenté ci-après.

2.2.3 Application au monde des blocs

L'exemple ci-dessous nous place dans un monde des blocs réduit à 3 éléments. Le but est d'arriver dans un état où le cube a est situé sur le cube b.

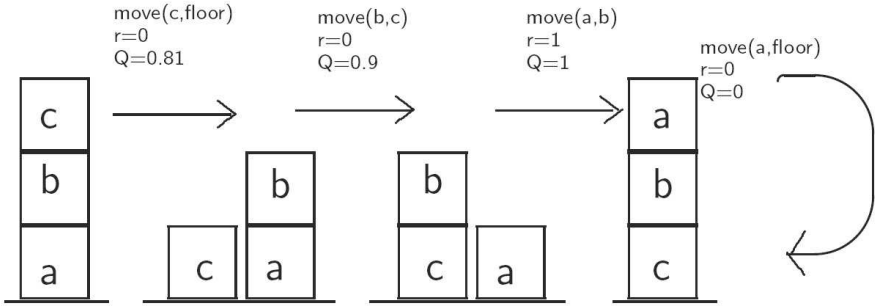

FIG. 2.3 – Exemple d'apprentissage par Q-RRL

La figure 2.3 montre les étapes par lesquelles est passé l'algorithme afin d'atteindre le but cherché. On a représenté sur les différentes transitions la valeur de la fonction Q, la décision retenue ainsi que la récompense immédiate associée (nulle si l'action ne débouche pas sur l'état cible ou si l'on était déja sur celui-ci). La fonction Q a été calculée avec γ=0,9.

Une fois l'état but atteint, l'algorithme sort de la boucle "while" et génère les exemples à fournir à l'algorithme TILDE-RT. Ceux-ci sont consitués de la valeur courante de la fonction $\hat{q}$, de l'action sélectionnée à l'étape t, du but à atteindre et de la description de l'état courant (représenté par un nombre variable de prédicats). Les exemples fournis en parametre de TILDE-RT sont détaillés dans la figure 2.4.

Example 1	Example 2	Example 3	Example 4
qvalue(0.81).	qvalue(0.9).	qvalue(1.0).	qvalue(0.0).
move(c,floor).	move(b,c).	move(a,b).	move(a,floor).
goal(on(a,b)).	goal(on(a,b)).	goal(on(a,b)).	goal(on(a,b)).
clear(c).	clear(b).	clear(a).	clear(a).
on(c,b).	clear(c).	clear(b).	on(a,b).
on(b,a).	on(b,a).	on(b,c).	on(b,c).
on(a,floor).	on(a,floor).	on(a,floor).	on(c,floor).
	on(c,floor).	on(c,floor).	

FIG. 2.4 – Ensemble des exemples fournis à TILDE

A partir de cette liste d'exemple, l'algorithme TILDE-RT génère le Q-arbre représenté sur la figure 2.5. La racine de l'arbre se charge d'introduire le couple état-action tandis que le reste de l'arbre ce charge de son évaluation, soit le calcul de valeur à associer à la fonction Q pour cette instance de parametres. Le but positionné en noeud initial satisfait la condition de récompense nulle si l'état courant est l'état but. Si le fait que l'arbre prédise une récompense de 1 si le bloc A est libre pourrait paraitre surprenant/optimiste, cette propriété valide le fait que A doit nécéssairement être libre pour pouvoir être déposé sur B.

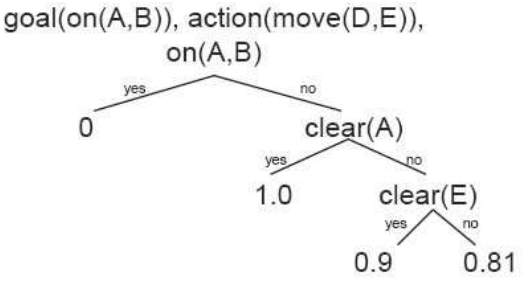

FIG. 2.5 – Arbre de régression généré par TILDE après 4 étapes

L'exécution de l'algorithme Q-RRL sur dix itérations (10 exemples fournis à TILDE-RT) permet d'obtenir le Q-arbre optimal pour le monde des 3-blocs. Et de la même maniére que dans l'arbre de la figure 2.5, les variables définissant les blocs ne sont pas instanciées, ce qui rend l'arbre obtenu optimal pour le but *on(A,B)* quelles que soient les valeurs des variables A et B. C'est l' avantage majeur de cette représentation.

L'algorithme Q-RRL a depuis sa premiére publication connu plusieurs modifications visant à améliorer ses capacités. On a nottament remplacé l'algorithme TILDE-RT (non incrémental), utilisé ici, par l'algorithme TG. D'autres algorithmes ont également été proposés afin d'apprendre directement la politique optimale sans passer par l'apprentissage de Q (P-RRL). Si ces différents travaux ont bien fait la preuve de leurs capacités de généralisation sur des exemples simples (le monde des blocs par exemple avec nb blocs <=5), le passage à des univers plus complexes s'est avéré problématique.

2.3 Les limites de l' Apprentissage par Renforcement Relationnel

2.3.1 Le point de vue influe sur l'apprentissage

Rejoignant ce que l'on connaît du raisonnement humain, la forme de la représentation et les connaissances préalables introduites influent, à supposer que toutes les données du problème soient intégrées, sur ce qui est éffectivement généralisé et/ou déduit par l'algorithme. De plus, si la complexité du problème oblige à réaliser une sélection de l'information avant la représentation, cette sélection introduit un biais d'apprentissage supplémentaire. Par conséquent, exprimer la structure relationnelle d'un domaine de manière à ce qu'elle permette de trouver la politique optimale, et ce quelque soit le but recherché, est très difficile.

2.3.2 La contrepartie d'une plus grande expréssivité

Bien que la représentation de l'espace d'état ne soit plus réalisée sous la forme d'un tableau à 2 dimensions, l'introduction de la logique du premier ordre, et donc d'un langage plus riche, a engendré une augmentation du nombre de composantes utilisables pour la description complète de chaque état. Cet accroissement a donc entraîné l'apparition des mêmes inconvénients que la représentation tabulaire initialement utilisée en AR. Ainsi, si le temps d'apprentissage de la fonction Q pour le but *on(A,B)* par l'algorithme Q-RRL requiert 20 minutes de temps de calcul pour un monde de 3 blocs, le calcul de l'arbre optimal pour un univers à 5 blocs nécéssite 12,4h de calcul[3].

[3]machines Sun cadencées à 270Mhz et disposant de 128Mo de Ram

2.4 L'approche ascendante : une solution ?

Pour tenter de pallier aux limitations de l'approche descendante introduite par Dzeroski, certains travaux plus recents ont abordé le problème de la généralisation sous l'angle ascendant. Travaillant sur une représentation relationnelle de l'espace d'état sous la forme d'un réseau bayésien naif et non plus sous la forme d'arbre de régression, Scott Sanner a ainsi dévellopé un algorithme offrant la possibilité d'apprendre simultanément la valeur des paramètres et la structure du réseau tout en limitant la complexité opératoire de celui-ci (SVRRL).

2.4.1 Les réseaux bayésiens naifs

Les réseaux bayésiens naifs sont une méthode de catégorisation. On cherche à déterminer l'état d'un système ou la classe d'un objet à partir de plusieurs observations sur le système ou l'objet. On utilise pour cela un réseau bayésien classique auquel on adjoint une hypothese d'indépendance des différentes observations. Ainsi l'expression :

(1) $\mathrm{P}(C \cap O_1 \cap O_2 \cap ... \cap O_n) = P(C) * P(O_1 \cap O_2 \cap ... \cap O_n|C)$

devient par la prise en considération de l'hypothese d'indépendance

(2) $\mathrm{P}(C \cap O_1 \cap O_2 \cap ... \cap O_n) = P(C) * P(O_1|C) * P(O_2|C) * ... * P(O_n|C)) = P(C) * \prod_{i=1}^{n} P(O_i|C)$

Bien que cette hypothèse se revèle généralement fausse (c'est l'hypothèse naive), les réseaux bayesiens naifs donnent en pratique de très bon résultats pour les problemes de classification, et l'utilisation de cette hypothèse permet de simplifier drastiquement les calculs.
Dans le cas de l'équation (1), si les observations O_i et la classe C sont bivaluées, il y aura $2^n * 2$ probabilités à calculer pour la probabilité conditionnelle alors que pour l'équation (2) on se ramène à n*4 probabilités pour évaluer ce même terme.

2.4.2 L'algorithme SVRRL

Apprentissage de la valeur des parametres

En se placant dans une catégorie de problème où la récompense se situe uniquement lors de la transition vers l'état final et où le nombre d'itérations est potentiellement infini, la fonction de valeur a une interprétation probabiliste : elle correspond à la probabilité conditionnelle d'un éventuel succès considérant les états succesivement observés du système.

$\mathrm{V}^*(\mathrm{s})=\mathrm{E}[\sum_{t=0}^{\infty} r^t|s^{t=0} = s_0)$

$\mathrm{V}^*(\mathrm{s})=P(s^{t\rightarrow\infty} \in EtatFinal|s^{t=0} = s_0) = P(succs|s_0)$

C'est cette propriété qui permet d'exprimer et d'exploiter la structure relationnelle du problème par le biais d'un réseau bayesien. En effet, du fait des caractéristiques de ces réseaux, il est possible d'obtenir une approximation des parametrès du réseau en calculant la valeur du maximum de vraisemblance de la probabilité $\hat{P}(succs|s)$ (avec s, description de l'état courant du système par une conjonction de prédicats).

$\hat{P}(succs|s) = \frac{\hat{P}(s|succs)*\hat{P}(succs)}{\hat{P}(s)}$

Ce qui, considerant le fait que l'état s est totalement décrit par une conjonction de prédicats positifs et négatifs et que l'hypothese d'indépendance des observations est posée, peut s'exprimer :

$\hat{P}(succs|s) = \frac{\hat{P}(succs)\prod_{i=1}^{p}\hat{P}(s_i|succs)\prod_{i=p+1}^{n}\hat{P}(\overline{s_i}|succs)}{\sum_{x\in(succs,echec)}\hat{P}(x)\prod_{i=1}^{p}\hat{P}(s_i|x)\prod_{i=p+1}^{n}\hat{P}(\overline{s_i}|x)}$

Apprentissage de la structure de la représentation

L'apprentissage de la structure du réseau est quant à lui constitué de 2 principales opérations :l'ajout de noeuds au réseau et la fusion de certains d'entre eux.
La premiére opération peut être vue comme une recherche de l'estimation la plus fine des probabilités conditionnelles associées à chaque état rencontré. A l'inverse, la seconde vise à remettre en cause l'hypothese naive d'indépendance des observations et compare le niveau d'information apporté entre une probabilité jointe de type $P(O_1 \cap O_2|succs)$ et le produit des probabilités $P(O_1|succs) * P(O_2|succs)$. Dans le cas où la probabilité jointe s'avère plus pertinente, l'algorithme fusionne les noeuds concernés (la comparaison des probabilités de deux noeuds n'etant réalisée que lorsque ces derniers sont potentiellement compatibles, par exemple *Expose(1,0)* et *Expose(2,0)*).

Application à l'apprentissage du Backgammon

L'approche utilisée pour l'apprentissage de la fonction de valeur et de la structure du réseau bayésien permet d'utiliser l'algorithme SVRRL sur des problèmes disposant d'un facteur de branchement conséquent. Ainsi, l'application de ce dernier à l'apprentissage du Backgammon (jeu possédant un espace de 10^{18} états) est rendue possible. Les résultats obtenus sont présentés en figure 2.6.

Player	Winning PCT	# Training Games
TD-Gammon 1-Ply (est)	66.0 % ± ???	1,500,000
FAA-SVRRL	51.2 % ± 0.02	5,000
Pubeval	50.0 % ± 0.00	unknown
HC-Gammon	40.0 % ± 3.46	100,000

Fig. 2.6 – Pourcentage de victoire de différents programmes de Backgammon face à Pubeval

En plus d'obtenir un nombre de victoire supérieur à 50% face à Pubeval, il est important de noter que seuls 5000 cycles d'apprentissage, réalisés en 10 mins, ont été nécéssaires pour obtenir ce niveau de jeu[4]. La vitesse de convergence de l'algorithme SVRRL en comparaison des 100000 cycles d'apprentissage nécéssaires au HC-Gammon ou des 1,500 000 cycles de l'algorithme dévellopé par Tesauro est plus que significative.

[4]machine intel Pentium III cadencée à 1Ghz et disposant de 128Mo de Ram

CONCLUSION

Si l'appprentissage par renforcement relationnel, par l'introduction de la logique du premier ordre, a fait montre de capacités de généralisation absentes du domaine de l'apprentissage par renforcement, les résultats obtenus ne sont pas sans être accompagnés de sérieux inconvénients. Ainsi, la forme de la représentation et les connaissances acquises influent sur ce qui est appris. Ce fait rendant chaque action dépendante des interactions passées nous contraint à abandonner la propriété de Markov et les preuves de convergences associées.
De plus, le fait que Dzeroski et Driessens posent l'hypothèse du monde clos et la description complète de chaque état comme un préalable à leurs travaux apparaît comme surprenant. Il semble en effet que cette hypothèse aille à l'encontre des principes qui ont conduits au développement de l'ARR, à savoir l'utilisation d'outils de généralisation permettant de travailler en environnement complexes (au sens de l'espace d'état) et ouverts.

Alors que les avantages de l'ARR du point de vue d'un agent sont étudiés et que certains travaux récents laissent présager d'une possible utilisation des principes de l'ARR dans des univers étendus (Sanner, Guestrin), très peu de travaux ont abordé l'utilisation de l'ARR dans le cadre multi-agents. Dans ce cadre, du fait de la complexité des univers concernés, l'échange de connaissance sur l'espace d'état entre les différents agents pourrait faire bénéficier le systeme de gains de temps non négligeables en phase d'exploration. De même, la mise en place d'une relation enseignant-apprenant entre différents agents pourrait se réveler une alternative pertinente à la sous-traitance de tâches et aux contraintes associées.

Bibliographie

Relational Reinforcement Learning : Kurt Driessens - Saso Dzeroski - Luc de Raedt (2001)

Relational Reinforcement Learning : An Overview : Kurt Driessens - Robert Givan - Prasad Tadepalli (2001)

Relational Reinforcement Learning for Agents in Worlds with Objects : Saso Dzeroski (2003)

Simultaneous Learning of Structure and Value in RRL : Scott Sanner (2005)

Online Feature Discovery in Relationnal Reinforcement Learning : Scoot Sanner (2006)