Chapitre 4

Une introduction aux jeux stochastiques

4.1. Introduction

La théorie des jeux [AUM 02] est un formalisme qui vise à étudier les interactions entre agents sachant que de telles interactions peuvent s'étendre de la coopération au conflit. Originellement conçue comme un outil mathématique pour les sciences économiques, son utilité a été montrée au travers des années en logique et en théorie des ensembles [GRÄ 02, MAR 75], en biologie et en théorie de l'évolution [SMI 02], en informatique [PAP 95, PAR 02, GIE 06], en analyse des conflits [MYE 97], etc.

L'interaction est la pierre angulaire des études sous-tendues par la théorie des jeux. Pour modéliser les interactions, il faut tout d'abord étendre la notion de théorie des jeux à des *jeux dynamiques*, généralement utilisés pour rendre compte de la compétition entre processus évoluant dans le temps. Des transitions stochastiques sont ensuite utilisées pour formaliser l'incertain. Les *jeux stochastiques (ou markoviens)* sont des jeux dynamiques avec des transitions stochastiques. Ils forment actuellement une théorie mathématique mature et riche, utilisée dans beaucoup d'applications comme l'économie, la biologie et tout ce qui tourne autour de l'évolution et des populations, les files d'attentes, les télécommunications, le *model checking*, etc. Ces derniers temps, les jeux stochastiques ont pris beaucoup d'importance en informatique au travers plus particulièrement des systèmes multi-agents spécialement pour la décision, la planification et l'apprentissage dans un environnement où interviennent plusieurs agents autonomes [STO 00].

Chapitre rédigé par Andriy BURKOV et Brahim CHAIB-DRAA.

Dans ce chapitre, nous considérons les jeux stochastiques comme étant le modèle multi-agent le plus général. Nous présentons de façon détaillée plusieurs algorithmes pour résoudre les problèmes qui peuvent être modélisés par des jeux stochastiques, même si la plupart de ces algorithmes peuvent aussi résoudre des modèles plus simples. Les systèmes multi-agents « purement collarobatifs » – tous les agents partageant les mêmes objectifs – sont présentés de manière plus approfondie dans le chapitre **??** du volume 2.

Le reste du chapitre est organisé comme suit. La section 4.2 présente les principales notions de la théorie des jeux classique, telles que les jeux en forme stratégique, les jeux dynamiques, les équilibres de Nash et autres. Au paragraphe 4.3, le modèle des jeux stochastiques (aussi appelés « jeux de Markov ») est défini et différents algorithmes pour le résoudre sont présentés et discutés. La section 4.4 conclut ce chapitre.

4.2. Rappel sur la théorie des jeux

4.2.1. *Quelques définitions de base*

Une partie de poker, la formation d'une équipe ou une négociation entre agents pour la prise de rendez-vous sont autant de jeux différents obéissant à des règles spécifiques. Dans ces jeux, chaque participant ne peut être totalement maître de son sort ; on dit alors que tous les intervenants se trouvent en situation d'interaction stratégique [THI 04]. La théorie des jeux vise à étudier formellement ce type de jeux où le sort de chaque agent participant dans le jeu dépend non seulement des décisions qu'il prend mais également des décisions prises par les autres agents intervenant dans le jeu. Dès lors, le « meilleur » choix pour un agent dépend généralement de ce que font les autres.

Les agents participant à un jeu sont appelés joueurs. Ainsi un joueur est un agent qui pourrait représenter une entreprise, un robot, un consommateur, etc. et qui agit pour son propre compte selon le principe de la rationalité qui vise à agir au mieux par rapport à ses buts, par exemple en maximisant une mesure de performance donnée qui mesure le succès de l'agent, comme le précisent Russell et Norvig [RUS 95]. Ainsi, chaque agent cherche à prendre les « meilleures décisions » pour lui-même et ne fait pas référence à un quelconque « sacrifice » pour autrui. Bien entendu, ceci n'est plus valable si on s'intéresse à des équipes d'agents où les participants poursuivent un objectif commun.

En théorie des jeux, il est important de garder à l'esprit que les agents participant au jeu (appelés dorénavant agents ou joueurs) se doivent de choisir leurs propres actions, en tenant compte des actions des autres participants. Ils doivent raisonner sur autrui et se faire une idée aussi précise que possible du comportement possible

des autres joueurs. A cet effet, la théorie des jeux admet : i) que chaque joueur s'efforce de prendre les meilleures décisions pour lui-même et sait que les autres font de même ; ii) que le précédent fait, c'est-à-dire i), est une connaissance commune à tous les joueurs.

4.2.1.1. *Critères distinguant différentes formes de jeux*

Les jeux à un seul joueur (jeux d'adresse, casses-têtes, énigmes, etc.). Ils se ramènent souvent à des problèmes d'optimisation classique, de planification dans un espace de grande taille par exemple pour les casses-tête ou de planification dans l'incertain (MDP) pour les jeux d'adresse. On s'intéresse donc ici à des jeux à plusieurs joueurs et, la plupart du temps, à *information complète*, c'est-à-dire que chaque joueur connaît les règles du jeu (dont les coups que peuvent choisir lui-même et les autres joueurs selon le moment du jeu) ainsi que les objectifs de tous les joueurs. Dans le cas contraire, dans le cas d'un jeu à information incomplète, il faudrait soit agir de manière prudente, soit essayer de comprendre les objectifs des autres joueurs.

Avec deux joueurs ou plus, le mot « jeu » peut faire en premier lieu penser aux échecs ou aux dames, des jeux *séquentiels* – parce que chaque joueur joue à son tour – et à *information parfaite* – parce que l'on agit en connaissance de la situation exacte du jeu.

On se trouve dans une situation différente dans un jeu de cartes classique, les tours de jeu étant en général aussi alternés, mais le jeu étant souvent à *information imparfaite* : on ne sait pas quelles cartes les autres joueurs ont ou l'ordre des cartes dans la botte. Une autre différence est qu'ici apparaît la notion de hasard : il faut que les cartes aient été battues pour que l'on ne sache pas comment elles ont été distribuées. Un jeu peut aussi être à information imparfaite si un joueur n'observe pas tous les coups effectués par les autres joueurs, par exemple quelles cartes sont passées d'une main à une autre ou ont été remises dans le tas.

Dans certains autres jeux, plusieurs joueurs (voire tous) peuvent agir en même temps. On parle alors de jeu *simultané*. C'est le cas par exemple dans « diplomatie », un jeu de stratégie dans lequel les prochains mouvements des troupes de chaque joueur sont à chaque tour : 1) discutés entre joueurs (qui peuvent mentir), 2) notés sur des feuilles de papier et 3) exécutés simultanément (où l'on voit qui a menti à qui). C'est le cas aussi dans le cas plus simple d'un penalty ou des tirs au but en football : tireur et gardien doivent souvent choisir leurs mouvements simultanément en espérant avoir deviné quel côté l'adversaire aura choisi.

La plupart des jeux peuvent se ramener à des jeux à somme nulle, c'est-à-dire des jeux dans lesquels la somme des gains des joueurs vaut toujours zéro. Dans le cas d'un jeu à deux joueurs, cela signifie trivialement que, quand l'un gagne, l'autre perd.

Mais, comme dit plus haut, la théorie des jeux est beaucoup étudiée dans le cadre de problèmes d'économie, où plusieurs acteurs agissent dans un même environnement – donc « interagissent » – en ayant des objectifs propres. Toutefois ces objectifs ne sont pas nécessairement en complète opposition. Il peut y avoir des situations de compromis voire de coopération. Un problème délicat est par exemple celui des transports publics : chaque joueur a le choix de se déplacer en bus ou en voiture, les temps de parcours de chacun (ce que chacun cherche à minimiser) dépendant de son mode de déplacement (voiture>bus) et de l'encombrement (plus il y a de voitures, moins ça roule) ; le résultat est qu'il faudrait idéalement que tous prennent le bus, mais qu'en fin de compte chacun tend égoïstement à préférer la voiture.

Pour terminer, on citera une problématique particulière, celle des « jeux coopératifs », qui s'intéresse à la formation de coalitions entre joueurs dont l'intérêt devient alors commun face aux autres joueurs ou coalitions. Nous n'approfondirons pas ce sujet, mais il est au cœur d'un certain nombre de travaux dans le domaine des systèmes multi-agents.

Dans ce chapitre, nous nous intéressons essentiellement aux jeux à information complète, cette section débutant par la présentation de jeux simples : les jeux statiques qui se jouent en un seul coup.

4.2.2. *Jeux statiques à information complète*

De nombreux problèmes de théorie des jeux peuvent se formuler et s'étudier sous la forme de jeux dits statiques (aussi appelés jeux simultanés). Il s'agit de jeux sans hasard et se jouant en un seul tour, tous les joueurs choisissant leur unique coup simultanément.

Deux entraîneurs d'équipes de football rencontrent une situation très similaire quand leurs équipes respectives vont se confronter et qu'ils doivent chacun choisir en même temps (avant le match) : les joueurs qu'ils vont faire jouer, leurs rôles respectifs sur le terrain et d'autres aspects tactiques qui resteront figés pendant le match.

4.2.2.1. *Jeux en forme stratégique, stratégie pure, stratégie mixte*

Un jeu G[1] en forme stratégique (ou jeu en forme normale) est un tuple $\langle Ag, \{A_i : i = 1 \ldots |Ag|\}, \{R_i : i = 1 \ldots |Ag|\}\rangle$. Les éléments constitutifs de ce jeu sont les suivants [THI 04] :

– $Ag = 1 \ldots |Ag|$ est l'ensemble fini des joueurs. Pour éviter toute confusion, un joueur quelconque est noté i. Bien entendu $i \in Ag$;

1. Pour l'anglais *game*.

– a_i désigne la stratégie du joueur i. Une telle stratégie décrit de manière précise ce qu'un joueur fait. Par extension, l'ensemble A_i décrit toutes les stratégies disponibles pour le joueur i. Bien entendu, $a_i \in A_i$. Un jeu en forme stratégique est fini si l'ensemble des actions de chaque joueur, A_i, est fini.

En outre, tous les joueurs agissent simultanément (ou du moins sans savoir la stratégie choisie par les autres joueurs) ;

– dès lors, $\mathbf{a} = (a_1, \ldots, a_i, \ldots, a_{|Ag|}) \in A_1 \times \ldots \times A_i \times \ldots \times A_{|Ag|} \equiv \mathbf{A}$ est une issue du jeu ; autrement dit une combinaison de stratégies où a_i est la stratégie pour l'agent i. Dans le reste du chapitre, $\mathbf{a}_{-i} \in \mathbf{A}_{-i}$ désigne l'ensemble de toutes les stratégies choisies par les joueurs sauf celle du joueur i ;

– $R_i(\mathbf{a}) \in \mathbb{R}$ est la fonction de récompense du joueur i. On voit bien que la fonction de récompense du joueur i dépend non seulement de sa stratégie a_i, mais aussi de celles des autres joueurs reflétées dans $\mathbf{a}$. Bien entendu, le joueur i préfère strictement l'issue $\mathbf{a}$ à l'issue $\mathbf{a}'$ si $R_i(\mathbf{a}) > R_i(\mathbf{a}')$. Dans le cas, où $R_i(\mathbf{a}) = R_i(\mathbf{a}')$, i est dit indifférent aux deux issues $\mathbf{a}$ et $\mathbf{a}'$;

– chaque joueur connaît, outre les siens, les ensembles de stratégies et les fonctions de gain de tous les autres joueurs.

Cette dernière hypothèse caractérise le jeu comme étant en *information complète*. A l'inverse, un jeu est dit en *information incomplète* si les joueurs ne connaissent certains éléments du jeu au mieux qu'en termes de probabilités.

La figure 4.1 représente sous forme matricielle un exemple de jeu en forme stratégique pour deux joueurs, ENTREPRISE1 et ENTREPRISE2, ayant à leur disposition les actions *produit* et *ne produit pas*. Par convention les gains sont reportés sous la forme (x, y) pour une combinaison d'actions $(action_{Entreprise1}, action_{Entreprise2})$ et où x est le gain du joueur ligne (ici ENTREPRISE1) et y est le gain du joueur colonne (ici ENTREPRISE2).

		ENTREPRISE2	
		produit	*ne produit pas*
ENTREPRISE1	*produit*	$-3, -2$	$10, 0$
	ne produit pas	$0, 8$	$0, 0$

Figure 4.1. *Exemple de jeu en forme stratégique. Des valeurs dans chacune des cases indiquent les gains (en termes d'utilité) de chaque joueur pour chacune des actions jouées conjointement par les joueurs.*

Une *stratégie pure* reflète une action ou une suite d'actions choisies par chaque joueur de façon déterministe (par opposition à stochastique). Dans certains cas, il est préférable d'avoir recours à une *stratégie mixte* définie comme une distribution de probabilité sur l'ensemble des stratégies pures. Soit A_i l'ensemble des stratégies à

la disposition d'un joueur i. Désignons alors la stratégie pure d'un agent i comme une simple action, soit $a_i \in A_i$, et la stratégie mixte comme une distribution de probabilité[2] sur ces stratégies pures, soit $\pi_i = \Delta A_i$. Conformément à cette notation, on va aussi noter par $\pi_i^{a_i}$ la probabilité de l'agent i de jouer une action $a_i \in A_i$ et $\pi_{-i} = \Delta \mathbf{A}_{-i}$ la politique conjointe des autres agents par rapport à l'agent i.

Chaque joueur cherche la stratégie qui va lui permettre de maximiser son gain espéré étant donné les stratégies adoptées par les autres joueurs. Le terme « gain espéré » (ou « utilité espérée ») d'un joueur est la récompense moyenne qu'il s'attend à obtenir étant donné sa politique (stratégie mixte) et les politiques des autres joueurs. Désignons le gain espéré de l'agent i par u_i, alors :

$$\begin{aligned} u_i^{(\pi_i,\pi_{-i})} &= E_{\mathbf{a}\in\mathbf{A}} R_i(\mathbf{a}) \\ &= \sum_{a_i \in \mathcal{A}_i} \sum_{\mathbf{a}_{-i} \in \mathbf{A}_{-i}} R_i(a_i, \mathbf{a}_{-i}) \pi_i^{a_i} \pi_{-i}^{\mathbf{a}_{-i}}. \end{aligned} \tag{4.1}$$

Dans le cas d'un jeu à deux joueurs, on peut utiliser des notations matricielles, R_1 et R_2 étant des matrices, et π_1 et π_2 étant des vecteurs :

$$u_i^{(\pi_1,\pi_2)} = {\pi_1}^\top R_i \pi_2, \ \forall i \in \{1,2\}.$$

4.2.2.2. *Jeu à somme nulle et minimax*

Un jeu à est à somme nulle si, pour tout $\mathbf{a} \in \mathbf{A}$, $\sum_i R_i(\mathbf{a}) = 0$. On s'intéressera ici au cas de deux joueurs (notés 1 et 2). Les gains de l'un sont alors les pertes de l'autre et les joueurs sont donc des adversaires « au sens strict ». On peut par exemple modéliser un problème de prise de décision robuste par un jeu à somme nulle, l'objectif étant de trouver les décisions les plus sûres face à un environnement incertain (voir section 5.3). Ce genre de jeu particulier peut être approché (mais pas nécessairement résolu) par le principe de *minimax* (ou *maximin*).

Minimax est une technique de recherche d'une solution dans les jeux à somme nulle. L'algorithme Minimax a été élaboré en 1928 par John von Neumann [NEU 28]. Il s'applique pour les jeux à 2 joueurs à somme nulle. Les agents sont assignés à un des deux rôles : soit MAX soit MIN. Le joueur MAX est censé maximiser sa récompense,

2. On notre ΔE une distribution de probabilité sur un ensemble E.

alors que le joueur MIN est censé minimiser la récompense de MAX (ce qui revient à maximiser sa propre récompense).

Pour illustrer cela, supposons que le joueur i est celui qui cherche à maximiser. Il dispose de m stratégies a_{ik} avec $k = 1, 2, \ldots, m$. Le joueur j est celui qui cherche à minimiser. Il dispose de n stratégies $a_{jk'}$ avec $k' = 1, 2, \ldots, n$. L'ensemble de tous les gains possibles que i peut obtenir est représenté par une matrice R_i de dimensions $m \times n$ avec l'entrée $R_i(k, k')$. Dès lors, dans cette matrice R_i, l'agent i sélectionne les lignes de R_i, tandis que l'agent j sélectionne les colonnes. Dans ce cas, si le joueur i choisit la stratégie k tandis que j choisit la stratégie k', alors j doit payer à i le gain $R_i(k, k')$. Il convient de noter que les paiements négatifs sont permis car on est dans un jeu à somme nulle. On pourrait aussi dire que i reçoit la quantité $R_i(k, k')$ alors que j reçoit la quantité $-R_i(k, k')$.

Considérons alors l'exemple montré en figure 4.2 ci-dessous. Bien entendu, un tel jeu peut être représenté simplement par la matrice de la figure 4.3.

		MIN	
MAX	$3, -3$	$1, -1$	$8, -8$
	$4, -4$	$10, -10$	$0, 0$

Figure 4.2. *Jeu à somme nulle. Dans la matrice, les valeurs de la forme $\cdot, \cdot$ représentent respectivement les utilités des agents* MAX *et* MIN *pour chacune des actions conjointes*

		MIN	
MAX	3	1	8
	4	10	0

Figure 4.3. *Autre représentation matricielle d'un jeu à somme nulle. Les valeurs dans la matrice représentent seulement les utilités de l'agent* MAX

Dans ce jeu, la question est de savoir quelle option devra choisir un agent rationnel. Pour cela, on peut considérer les *niveaux de sécurité* de chacun des agents [HAU 98]. Il est en effet facile de voir que, si MAX choisit la première ligne, quoi que fasse MIN, il aura au moins un gain de 1. En choisissant la deuxième ligne, il risque de faire un gain nul. De façon similaire, en choisissant la première colonne, MIN n'aura pas à payer plus que 4, tandis que s'il choisit la seconde ou la troisième colonne, il risque de perdre respectivement 10 ou 8. On dit alors que le niveau de sécurité de MAX est 1 et qu'il est assuré par le choix de la première ligne, tandis que le niveau de sécurité de l'agent MIN est 4 et qu'il est assuré par le choix de la première colonne. Il convient de noter que : $\max_k \min_{k'} R_1(k, k') = 1$ et $\min_{k'} \max_k R_1(k, k') = 4$. Ceci montre que

la stratégie qui assure à l'agent MAX son niveau de sécurité est la *stratégie maximin*. Symétriquement, la stratégie qui assure à l'agent MIN son niveau de sécurité est la *stratégie minimax*.

PROPOSITION 4.1 [HAU 98].– *Dans une matrice représentant un jeu à deux joueurs à somme nulle, on a l'inégalité suivante :*

$$\max_k \min_{k'} R_1(k,k') \leq \min_{k'} \max_k R_1(k,k')$$

Il convient de noter que la solution maximin (ou minimax) est acceptable (optimale) si les joueurs jouent chacun à leur tour, par exemple, le joueur i commence par choisir une action puis le joueur j fait son choix en fonction de l'action jouée par j. Cependant, si les deux joueurs ont à jouer simultanément, une étude approfondie du jeu précédent illustré en figure 4.2 montrerait que, dans ce cas, les stratégies maximin et minimax ne sont pas des solutions satisfaisantes pour ce jeu. Dans certains cas, toutefois, le jeu pourrait converger vers une solution stable. Considérons un autre exemple, celui de la figure 4.4, emprunté à [HAU 98].

		MIN		
MAX				
	10	−15	20	
	20	−30	40	
	30	−45	60	

Figure 4.4. *Jeu où* Maximin = Minimax

Il est facile ici de voir que :

$$\begin{aligned} \max\{-15,-30,-45\} &= -15 \quad \text{et} \\ \min\{30,-15,60\} &= -15. \end{aligned}$$

Ainsi, la paire de gains correspondant aux stratégies maximin and minimax est donnée par $R_i(k,k') = (-15,15)$ et elle correspond à $(k,k') = (1,2)$, c'est-à-dire à la première ligne et à la deuxième colonne.

Si dans une matrice de jeu à somme nulle il y a une paire (k^*,k'^*) telle que :

$$\forall k,k' \quad R_1(k,k'^*) \leq R_1(k^*,k'^*) \leq R_1(k^*,k'),$$

alors on dit que la paire (k^*,k'^*) est un point selle.

PROPOSITION 4.2 [HAU 98].– *Si dans une matrice de jeu à somme nulle, on a :*

$$\max_{k} \min_{k'} R_1(k, k') = \min_{k'} \max_{k} R_1(k, k') = v$$

alors le jeu admet un point selle en stratégies pures.

Si (k^*, k'^*) est un point selle pour une matrice de jeu, alors les joueurs MAX et MIN ne peuvent améliorer leur gain unilatéralement en déviant de k^* et k'^* respectivement. On dit alors que (k^*, k'^*) est un *équilibre*, car tous les joueurs ont intérêt à s'y tenir.

S'il n'existe pas toujours d'équilibre en stratégies pures dans un jeu à deux joueurs et à somme nulle, il en existe toujours un en stratégies mixtes (appelé équilibre minimax mixte). En effet, si l'on raisonne sur des stratégies mixtes $\pi_i \in \Delta A_i$, on a :

$$\max_{\pi_1 \in \Delta A_1} \min_{\pi_2 \in \Delta A_2} {\pi_1}^\top R_1 \pi_2 \quad = \quad \min_{\pi_2 \in \Delta A_2} \max_{\pi_1 \in \Delta A_1} {\pi_1}^\top R_1 \pi_2.$$

On peut en fait obtenir les stratégies des deux joueurs en résolvant les deux problèmes plus simples suivants (en notant $a_i k$ la stratégie pure k du joueur i) :

$$\begin{aligned} \operatorname*{argmax}_{\pi_1 \in \Delta A_1} \min_{k'=1}^{|A_2|} {\pi_1}^\top R_1 a_2 k' &= ? \\ \operatorname*{argmin}_{\pi_2 \in \Delta A_2} \max_{k=1}^{|A_1|} a_1 k^\top R_1 \pi_2 &= ? \end{aligned}$$

4.2.2.3. *Équilibre en stratégies dominantes*

Le minimax est une approche classique pour les jeux à deux joueurs et à somme nulle. Dans d'autres situations, des algorithmes différents doivent être employés. Ce problème peut s'illustrer par le célèbre jeu du dilemme du prisonnier. Il s'énonce de la façon suivante : deux suspects (SUSPECT1 et SUSPECT2) sont interrogés séparément par un juge pour un délit grave. Le juge ne dispose pas d'éléments de preuve suffisants pour les condamner et l'aveu d'au moins un est indispensable. Dès lors, il propose à chaque accusé la liberté s'il avoue. En revanche, s'il nie et si l'autre avoue, il écope d'une peine de 15 ans. Si les deux avouent, ils peuvent espérer bénéficier de circonstances atténuantes et recevoir une peine de 8 ans. Enfin si les deux nient, ils seront condamnés pour des délits mineurs à 1 an de prison chacun. Avouer revient à

		SUSPECT2 *D*	SUSPECT2 *N*
SUSPECT1	*D*	$-8, -8$	$0, -15$
	N	$-15, 0$	$-1, -1$

Figure 4.5. *La matrice des gains des deux joueurs dans le dilemme du prisonnier*

dénoncer l'autre (en même temps que soi-même). On notera donc *D* comme « dénoncer » et *N* comme « nier » les deux actions. La matrice des gains des deux joueurs correspondante apparaît sur la figure 4.5.

On est donc amené à se poser la question : comment doivent se comporter les deux suspects, à supposer qu'ils soient rationnels ? On peut remarquer qu'*Avouer* est une stratégie qui conduit à une peine moins lourde que la stratégie *Nier* et ce, quel que soit le choix effectué par l'autre joueur. Par conséquent, chacun des suspects a intérêt à opter pour cette stratégie en vue de réduire sa peine. Selon la matrice des gains, chacun écope alors d'une peine de 8 ans de prison, ce qui constitue une condamnation assez lourde. Il faut bien voir que cette stratégie mise sur le fait qu'elle est choisie parce qu'elle donne un gain moindre que l'autre stratégie et ce sans avoir besoin de se faire une idée de ce que va faire l'autre. Une telle stratégie est appelée, en théorie des jeux, une *stratégie dominante*.

DÉFINITION 4.1.– *Dans un jeu en forme stratégique, une stratégie $a_i \in A_i$ est dite dominante pour le joueur i si, quel que soit $\hat{a}_i \in A_i$ et $\hat{a}_i \neq a_i$, on a :*

$$R_i(a_i, \mathbf{a}_{-i}) \geq R_i(\hat{a}_i, \mathbf{a}_{-i}), \qquad \forall\, \mathbf{a}_{-i} \in \mathbf{A}_{-i}.$$

Dans notre exemple, on voit bien que nos deux suspects ont intérêt à jouer leur stratégie dominante tous les deux et à ne pas dévier. La stratégie conjointe (*D*,*D*) est donc un équilibre (sorte de point fixe) pour les deux où chacun n'a pas intérêt à dévier. Plus généralement si, dans un jeu donné, tous les joueurs ont à leur disposition une stratégie dominante, alors ils ont intérêt à la choisir effectivement et, dans ce cas, le résultat du jeu est appelé *équilibre en stratégies dominantes*.

En fait, l'équilibre en stratégies dominantes existe rarement et il faut faire appel à d'autres types de solutions. Pour ces cas, il existe un concept de solution plus faible qui s'appelle l'équilibre de Nash.

4.2.2.4. *Equilibre de Nash*

DÉFINITION 4.2.– *On dit qu'une combinaison de stratégies $\mathbf{a}^*$ est un* équilibre de Nash *(en stratégies pures) si on a l'inégalité suivante pour chaque joueur i :*

$$R_i(a_i^*, \mathbf{a}_{-i}^*) \geq R_i(a_i, \mathbf{a}_{-i}^*) \qquad \forall\, a_i \in A_i.$$

Autrement dit, si le joueur i anticipe que les autres participants au jeu vont choisir les stratégies associées au vecteur $\mathbf{a}^*_{-i}$, il ne peut que maximiser son gain en choisissant la stratégie a^*_i. Celle-ci est en fait une meilleure réponse de i à $\mathbf{a}^*_{-i}$ (notée br_i, pour *best response*) et, s'il n'y en a qu'une, elle correspond à :

$$br_i : \mathbf{a}^*_{-i} \mapsto \underset{a_i \in A_i}{\operatorname{argmax}} R_i(a_i, \mathbf{a}^*_{-i}).$$

Dès lors, l'équilibre de Nash peut aussi s'écrire :

$$\forall i, a^*_i \in br_i(\mathbf{a}^*_{-i}).$$

Comme on peut le voir, l'équilibre de Nash constitue une combinaison de stratégies où chaque joueur maximise ses gains compte tenu des actions supposées des autres. Il a donc une propriété de « stabilité » qui est satisfaite pour chacun des joueurs, c'est pourquoi on parle d'« équilibre ».

Dans l'exemple de la figure 4.6 deux entreprises ENTREPRISE1 et ENTREPRISE2 ont la possibilité de se lancer dans la production d'un nouveau bien pour lequel les débouchés sont limités, sans qu'il y ait de compromis possible entre elles si toutes deux décident de produire. Ce jeu comporte 2 équilibres de Nash, (*ne produit pas*, *produit*) dont les gains sont (0,8) et (*produit*, *ne produit pas*) dont les gains sont (10,0) chacun correspondant à une situation où l'une des entreprises produit, l'autre s'abstenant de le faire. Cet exemple montre que, des deux équilibres, il n'y en a pas un qui apparaisse plus raisonnable que l'autre.

		ENTREPRISE2	
		Produit	*Ne produit pas*
ENTREPRISE1	*Produit*	$-3, -2$	$10, 0$
	Ne produit pas	$0, 8$	$0, 0$

Figure 4.6. *Multiplicité de l'équilibre de Nash. Ce jeu comporte deux équilibres de Nash en stratégies pures : (*ne produit pas*,* produit*) dont les gains sont (0,8) et (*produit*,* ne produit pas*) dont les gains sont (10,0).*

A cette difficulté de multiplicité d'équilibres s'ajoute le fait qu'il peut ne pas y avoir du tout d'équilibre de Nash, en stratégies pures, pour un jeu particulier. Un exemple bien connu est celui du jeu pile ou face (*matching pennies*) présenté sous la forme stratégique en figure 4.7.

Dans ce cas, il faut se tourner vers les stratégies mixtes. La notion d'équilibre de Nash (comme celle de meilleure réponse) s'étend en effet naturellement – comme la notion de minimax auparavant – au cas des stratégies mixtes.

		JOUEUR2 *Pile*	JOUEUR2 *Face*
JOUEUR1	*Pile*	$1, -1$	$-1, 1$
	Face	$-1, 1$	$1, -1$

Figure 4.7. *Le jeu pile ou face : un exemple de jeu n'ayant pas d'équilibre de Nash en stratégies pures*

DÉFINITION 4.3.– *On dit qu'une combinaison de stratégies mixtes π^* est un* équilibre de Nash *(en stratégies mixtes) si on a l'inégalité suivante pour chaque joueur i :*

$$u_i(\pi_i^*, \pi_{-i}^*) \geq u_i(\pi_i, \pi_{-i}^*) \qquad \forall\, \pi_i \in \Delta A_i.$$

Dans le jeu pile ou face, présenté en figure 4.7, le JOUEUR1 a une probabilité p de choisir *Pile* et une probabilité de $1-p$ de choisir *Face*. Pour le JOUEUR2 les deux probabilités sont respectivement de q et $1-q$. Dès lors, le gain espéré du JOUEUR1 est reflété par la fonction u_1 linéaire en p suivante :

$$\begin{aligned} u_1(p,q) &= pqR_1(Pile, Pile) + p(1-q)R_1(Pile, Face) \\ &\quad +(1-p)qR_1(Face, Pile) + (1-p)(1-q)R_1(Face, Face). \end{aligned}$$

Dans ce cas particulier d'un jeu à deux joueurs, l'équilibre de Nash est un point selle de la surfac $u_i(p,q)$. Pour maximiser son gain espéré, le JOUEUR1 doit trouver une probabilité p telle que $\frac{d(u_1)}{dq}(p) = 0$, soit :

$$\begin{aligned} & pR_1(Pile, Pile) - pR_1(Pile, Face) \\ +(1-p)R_1(Face, Pile) - (1-p)R_1(Face, Face) &= 0. \end{aligned}$$

Si on remplace les R_1 par les valeurs indiquées dans la matrice présentée en figure 4.7, on obtient alors : $p + p - (1-p) - (1-p) = 0$, d'où $p = 1/2$.

Le même raisonnement pour le JOUEUR2 (où cette fois-ci on chercherait le point où $\frac{d(u_1)}{dp}(q) = 0$) donne $q = 1/2$.

Le résultat (1/2,1/2) est appelé *équilibre en stratégies mixtes* et il correspond au fait que l'un ou l'autre des joueurs choisisse une fois sur deux pile et une fois sur deux face.

Mais peut-on trouver un équilibre de ce type dans n'importe quel jeu en forme stratégique ? Si le jeu est fini, la réponse est oui, grâce au théorème suivant :

THÉORÈME 4.1 [NAS 51].– *Tout jeu en forme stratégique fini admet un équilibre de Nash en stratégies mixtes.*

On pourrait se demander si la solution donnée par l'équilibre de Nash correspond à un mécanisme de coordination efficace. Deux concepts peuvent aider à répondre à cette question : l'efficacité au sens de Pareto et l'optimum de Pareto.

Pour le premier concept, on dit qu'une combinaison de stratégies $\hat{\mathbf{a}}$ *domine au sens de Pareto* une autre combinaison $\mathbf{a}$ si :

$$\begin{aligned} \forall i \qquad & R_i(\hat{a}_i, \hat{\mathbf{a}}_{-i}) \geq R_i(a_i, \mathbf{a}_{-i}) \quad \text{et} \\ \exists j \text{ tel que} \qquad & R_j(\hat{a}_j, \hat{\mathbf{a}}_{-j}) > R_j(a_j, \mathbf{a}_{-j}). \end{aligned}$$

Une combinaison de stratégies $\hat{\mathbf{a}}^*$ est un *optimum de Pareto* s'il n'existe pas une autre combinaison qui la domine au sens de Pareto. Par exemple, dans le dilemme du prisonnier présenté en figure 4.5, la combinaison (*D*,*D*) est un équilibre de Nash mais la combinaison (*N*,*N*) la domine au sens de Pareto. Il faudra donc retenir qu'un équilibre de Nash n'est pas nécessairement un optimum de Pareto.

Toutefois, quand il y a multiplicité des équilibres de Nash, un équilibre peut en dominer un autre au sens de Pareto. Le problème est de savoir comment « attirer » les joueurs vers cet équilibre dominant au lieu d'un autre équilibre qui sera dominé.

4.2.3. *Jeux dynamiques à information complète*

Contrairement aux jeux en forme statique (tels que les jeux en forme stratégique) qui sont joués une fois, les jeux dynamiques décrivent des processus étendus dans le temps. Ces processus comportent plusieurs intervenants (agents/joueurs) qui peuvent conditionner leur comportement au moment présent sur les décisions observables des autres joueurs dans le passé. Dans cette section, on suppose que le jeu se déroule en plusieurs étapes et que toutes les actions passées sont observables et connues par tous les participants. Dans ce contexte, une étape pourrait représenter, mais pas toujours, une période temporelle. Une stratégie dans un tel jeu spécifie l'action que choisit chaque joueur à chaque étape où il intervient, en fonction de l'état du jeu qui prévaut en ce moment. Dès lors, l'historique du jeu a son importance et chaque joueur choisit l'action qu'il convient d'effectuer en tenant compte de l'histoire passée du jeu.

Généralement, on distingue deux types de jeux dynamiques en information complète. Un premier type, les *jeux en forme extensive* où les joueurs jouent chacun à tour de rôle. On va se limiter au cas où le jeu est non seulement à information complète, mais aussi à *information parfaite*, c'est-à-dire que, quand c'est à son tour de jouer, chaque joueur connaît la situation du jeu, en particulier les coups effectués par les autres joueurs jusqu'à présent.

Dans le second type, appelé *jeu répété*, plusieurs agents choisissent leurs actions simultanément à une étape donnée du jeu. Pour chaque joueur, les actions des autres joueurs ne sont toutefois pas connues, mais l'historique lui l'est et il influence le choix de chacun.

Un cadre conceptuel général de ces jeux dynamiques (à information parfaite) peut être défini comme suit. On désigne par $\mathbf{a}^\theta$ le vecteur des actions conjointes choisies à l'étape θ du jeu par des participants qui interviennent à cette étape. Soit t une étape quelconque du jeu. On définit alors l'historique du jeu à l'étape t, $h^t = \{\mathbf{a}^0, \mathbf{a}^1, \ldots, \mathbf{a}^{t-1}\}$, par la séquence de toutes les décisions prises par les joueurs lors des étapes antérieures $\theta = 0, 1, \ldots, t-1$, avec h^0 comme historique initial. Par ailleurs, toutes les actions passées sont observables et connues par tous les joueurs. Il convient de noter que, pour $\theta > t$, le reste du jeu peut être vu comme un autre jeu induit par l'historique h^t ; ce nouveau jeu est appelé sous-jeu $G(h^t)$.

Un tel cadre formel va maintenant nous permettre de définir ce que l'on entend par stratégie pure au niveau d'un jeu dynamique admettant T étapes. Pour cela, notons d'abord : $A(h_i)$ l'ensemble des actions disponibles pour le joueur i dans la situation décrite par l'historique h_i, et A_i l'ensemble des actions dont peut disposer le joueur i. Une stratégie pure δ_i pour le joueur i est une application $\delta_i : H \mapsto A_i$ où $\delta_i(h_i) \in A(h_i)$. Pour un joueur donné i, une stratégie pure est donc un ensemble de règles de sélection d'une action particulière par un tel joueur à chaque étape du jeu, compte tenu de l'historique qui s'est déroulé jusqu'alors. Contrairement à ce qu'on a vu précédemment dans le cadre d'une stratégie pure dans un jeu statique, ici la définition prend en compte les décisions choisies précédemment. Elle permet donc une analyse dynamique des choix.

Considérons d'abord les jeux en forme extensive à information parfaite.

4.2.3.1. *Jeux en forme extensive à information parfaite*

Prenons l'exemple de l'arbre représenté en figure 4.8 où le jeu se déroule en plusieurs étapes. Ainsi pour le joueur 1 (dénoté par un nœud 1), une stratégie consiste à choisir entre les actions a et b. Pour le joueur 2 (nœud 2) qui intervient juste après, sa stratégie a_2 est une fonction définie sur l'ensemble des stratégies de 1. Le principe

ici consiste par exemple à utiliser la *rétroduction*[3] où le raisonnement se fait en sens inverse du déroulement normal du jeu. Dès lors, le joueur 1 va se dire que le joueur 2 va jouer :

– b s'il joue a, aboutissant ainsi au gain $(2, 1)$ ou,

– a s'il joue b, aboutissant ainsi au gain $(1, 1)$.

Dès lors, le joueur 1 va jouer a, amenant ainsi le joueur 2 à jouer b et aboutissant ainsi à l'équilibre de Nash $(2, 1)$ pour lequel aucune déviation unilatérale n'est payante.

En fait, ce raisonnement est sous-tendu par le fait que le joueur 2 est censé choisir la meilleure action pour lui. Ainsi, si 1 joue a alors 2 joue b, mais si en revanche 1 joue b alors 2 joue a. Dès lors, le joueur 1 intègre une telle connaissance et agit en conséquence pour fournir sa meilleure réponse.

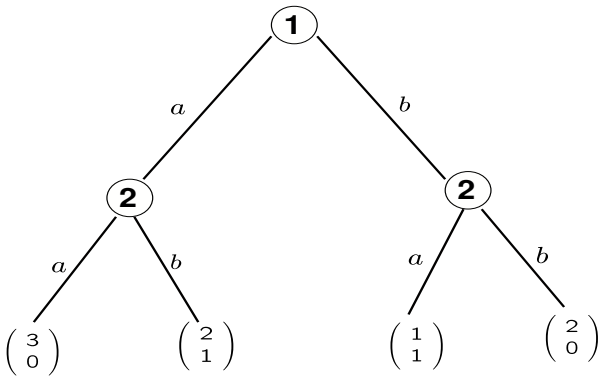

Figure 4.8. *Un exemple de jeu en forme extensive (jeu dynamique joué étape par étape)*

Dans le cas général, les $|Ag|$ joueurs peuvent apparaître dans l'arbre sans ordre régulier. On notera que la connaissance du nœud courant n_t dans l'arbre est une information équivalente à l'historique courante h_t. Dans un jeu à information imparfaite, le joueur « actif » n'a qu'une observation incomplète de l'historique courante – les actions de certains agents à certains instants sont cachées – ce qui revient à seulement savoir que le nœud courant fait partie d'un ensemble $\{n_1, \ldots, n_l\}$.

Un autre point important est que les jeux en forme extensive peuvent tous s'écrire de manière équivalente sous la forme de jeux en forme normale. Il suffit de noter que chaque joueur a un nombre fini de stratégies pures δ_i disponibles (parce que l'arbre à

3. *Backward induction*, qu'on traduit aussi par « induction rétroactive ».

une taille finie), ce qui permet de réécrire par exemple le jeu en forme extensive de la figure 4.8 sous forme matricielle comme sur la figure 4.9. On notera que, dans le cas de l'information parfaite, on se retrouve avec un jeu pour lequel il existe un équilibre en stratégies dominantes.

		JOUEUR 2			
		$a \to a \,/\, b \to a$	$a \to a \,/\, b \to b$	$a \to b \,/\, b \to a$	$a \to b \,/\, b \to b$
JOUEUR 1	a	$3, 0$	$3, 0$	$1, 1$	$1, 1$
	b	$1, 1$	$2, 0$	$1, 1$	$2, 0$

Figure 4.9. *Jeu de la figure 4.8 réécrit sous une forme matricielle. Pour le joueur 2, on indique quelle est sa réponse au coup du joueur 1*

Du fait de cette équivalence, nous n'approfondirons pas ici les jeux en forme extensive. Pour en savoir plus sur ces jeux, le lecteur peut se référer à l'un des ouvrages [YIL 03, FUD 91].

Les jeux répétés, un autre type des jeux dynamiques à information complète, sont de plus grand intérêt pour nous, car ils sont à la base du concept des jeux stochastiques – un des plus puissants modèles d'interaction entre agents rationnels.

4.2.3.2. *Les jeux répétés*

Avec les jeux répétés (ou jeux itérés), on modélise des situations où des joueurs interagissent de manière répétitive les uns avec les autres, en jouant le même jeu. Contrairement aux jeux à forme extensive et information parfaite, les joueurs d'un jeu répété choisissent leurs actions de manière simultanée (sans connaître le choix des autres joueurs). Une fois les actions choisies, celles-ci sont alors connues par les autres agents et dès lors font partie de l'historique du jeu.

Ainsi, le jeu du dilemme du prisonnier peut par exemple être répété plusieurs fois où l'on aurait par exemple l'historique $((N, N), (D, N), (D, D)$, etc$)$.

Dans de telles interactions, un joueur peut, à chaque étape, conditionner son comportement sur les comportements passés des autres intervenants. Les jeux répétés sont un cas particulier des jeux dynamiques introduits précédemment. La particularité réside dans le fait que la répétition se fait sur la base d'un même jeu. Quand ils sont engagés dans une situation répétitive, les joueurs doivent non seulement considérer leur gain à court terme (immédiat) mais également à long terme, sachant que leurs actes ont une influence sur les actes futurs des autres joueurs. Il y a ainsi introduction de phénomènes de réputation, de confiance, etc. Par exemple, si un dilemme du prisonnier est joué une fois, les joueurs auront tendance à jouer l'équilibre de Nash, soit (D, D). En revanche, si le même jeu est répété par les mêmes deux joueurs, certains

comportements pourraient faire émerger une coopération (entente implicite) qui aboutirait à (N, N). On distingue en général deux classes de jeux répétés : a) jeux répétés à horizon temporel fini et b) jeux répétés à horizon temporel infini.

A titre d'exemple et en reprenant la définition d'une stratégie pure d'un jeu dynamique donnée plus haut, on pourrait spécifier dans le cas du dilemme du prisonnier la stratégie suivante :

$$\begin{aligned} a_i(h^0) &= N, \\ a_i(h^t) &= \begin{cases} N & \text{si } a_j^\tau = N, j \neq i, \text{ pour } \tau = 0, 1, \ldots, t-1, \\ D & \text{autrement.} \end{cases} \end{aligned}$$

Une telle stratégie traduit « commencer par nier dans la première période et continuer ainsi tant que l'autre le fait également dans les périodes précédentes ; si ce n'est pas le cas, dénoncer ». Cette stratégie est appelée la stratégie *grim trigger*. D'autres stratégies peuvent être mises en évidence, en utilisant la même formulation, en particulier : (i) toujours dénoncer (*always Defect—ALL-D*) ; (ii) toujours nier (*always cooperate—ALL-C*) ; (iii) donnant-donnant (*Tit for Tat—TFT*) etc.

Comme dans les jeux statiques, la notion de stratégie mixte peut être définie dans le cadre des jeux dynamiques, et en particulier dans le cadre des jeux répétés. Dans un jeu répété, une *stratégie mixte* π_i pour le joueur i est une fonction $\pi_i : H \mapsto \mathcal{A}_i$, où $\mathcal{A}_i = \Delta A_i$ est l'espace des distributions de probabilité sur A_i, qui lie donc les historiques t-périodes possibles à des actions mixtes $\alpha_i \in \mathcal{A}_i$. On notera que, parce que seuls les coups effectivements joués par chaque joueur sont observables, la stratégie d'un joueur i dépend non des probabilités d'action passées de tous les joueurs (y compris lui-même) mais des actions passées observées (c'est-à-dire des historiques).

Voyons maintenant comment un jeu répété peut être analysé en termes de gains escomptés (gains futurs espérés) et contentons-nous pour cela du cas des stratégies pures pour l'exemple du dilemme du prisonnier (voir figure 4.5). Dans ce cas, les gains des joueurs pour toute la séquence des répétitions du jeu peuvent être représentées par *la moyenne des récompenses* qu'ils obtiennent à chaque période. Dans le cas du SUSPECT1 par exemple, sa fonction de gain sur l'ensemble du jeu répété pour toujours est :

$$\overline{u}_1 = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} R_1(\mathbf{a}^t)$$

où R_1 est la fonction de récompense d'un tel suspect. On peut de la même façon imaginer qu'un tel suspect utilise *la valeur actualisée de ses récompenses* sur la totalité

de la séquence de jeux :

$$u_{1,\gamma} = \sum_{t=1}^{\infty} \gamma^{t-1} R_1(\mathbf{a}^t) \qquad \gamma \in [0,1[$$

où γ est le facteur d'actualisation du SUSPECT1. En combinant les deux fonctions précédentes, on obtient la *moyenne actualisée des récompenses* :

$$\overline{u}_{1,\gamma} = (1-\gamma) \sum_{t=1}^{\infty} \gamma^{t-1} R_1(\mathbf{a}^t)$$

où l'on espère que l'égalité suivante est vérifiée :

$$\lim_{\gamma \to 1} \overline{u}_{1,\gamma} = \overline{u}_1.$$

Ainsi donc, il existe plusieurs possibilités pour représenter les gains à long terme des intervenants dans un jeu répété[4]. Voici un exemple tiré de [YIL 03] sur la manière de les utiliser. Reprenons le dilemme du prisonnier illustré en figure 4.5 répété infiniment et considérons la stratégie *grim trigger* discutée plus haut. Admettons que le joueur SUSPECT1 a adopté cette stratégie et que le SUSPECT2 est au courant de cela. On pourrait se demander quelle est alors la stratégie optimale de SUSPECT2, face à cette stratégie de SUSPECT1. S'il joue tout le temps *N*, il va obtenir un flux continu de récompenses égales à -1 jusqu'à la fin des temps. La valeur actualisée de ces récompenses dans ce cas est :

$$\sum_{t=1}^{\infty} -\gamma^{t-1} = \frac{-1}{1-\gamma}.$$

En revanche, s'il commence par *D* dès le premier tour, alors il va obtenir initialement 0 puis -8 pour le reste des périodes, ce qui lui donne une valeur actualisée de :

$$0 + (-8)\gamma + (-8)\gamma^2 + \ldots = -8(\gamma + \gamma^2 + \ldots) = \frac{-8\gamma}{1-\gamma}.$$

Le joueur SUSPECT2 aura donc intérêt à coopérer dès le début si :

$$\frac{-1}{1-\gamma} > \frac{-8\gamma}{1-\gamma}, \quad \text{c'est-à-dire} \quad \gamma > \frac{1}{8}.$$

La stratégie précédente du SUSPECT1 et sa contrepartie qu'on vient de détailler pour le SUSPECT2 forment un équilibre de Nash du dilemme du prisonnier répété

4. On retrouve les critères moyen et γ-pondéré vus dans les MDP en section 1.2.3.

infiniment en valeurs actualisées pour les valeurs de $\gamma > \frac{1}{8}$. Si on choisissait une autre forme pour la fonction d'utilité des agents (par exemple, la moyenne des récompenses ou la moyenne actualisée des récompenses), on pourrait obtenir une autre solution au même problème. Pour plus de précisions sur ces aspects, le lecteur est encouragé à consulter les ouvrages de référence [FUD 99, GEN 00, OSB 04, YIL 03].

4.3. Jeux stochastiques

Les jeux stochastiques (SG, pour *stochastic games*) – aussi appelés jeux de Markov – étendent les MDP au cas où il y a plusieurs joueurs dans un environnement commun. Ces agents exécutent une action conjointe qui définit la récompense obtenue par les agents et le nouvel état de l'environnement. De l'autre côté, un jeu stochastique peut être vu comme un jeu répété à plusieurs états. Ça veut dire qu'après avoir joué un jeu matriciel (correspondant à un état du jeu stochastique), les joueurs sont transférés dans un autre état du jeu stochastique pour jouer un autre jeu matriciel. C'est donc un modèle hybride réunissant jeux dynamiques (répétés) et MDP.

4.3.1. *Définition et équilibre d'un jeu stochastique*

Formellement, un jeu stochastique est défini par un quintuplet $\langle Ag, S, \mathbf{A} = \{A_i : i = 1, \ldots, |Ag|\}, \{R_i : i = 1, \ldots, |Ag|\}, T\rangle$ où Ag est l'ensemble des agents et $|Ag|$ est leur nombre, S l'ensemble fini d'états du jeu, $A_i = \left\{a_i^1, \ldots, a_i^{|A_i|}\right\}$ l'ensemble d'actions de l'agent i, $R_i : S \times A_1 \times \ldots \times A_{|Ag|} \mapsto \mathbb{R}$ est la fonction de récompense de l'agent $i \in Ag$ et $T : S \times A_1 \times \ldots \times A_{|Ag|} \times S \mapsto \mathbb{R}$ est le modèle de transition entre états, dépendant de l'action conjointe des agents (notée $\mathbf{a}$). Notons que les ensembles S et $\mathbf{A}$ sont généralement de taille exponentielle en fonction du nombre d'agents.

A chaque tour du jeu, étant donné l'état courant $s \in S$, les agents choisissent les actions $a_1, \ldots, a_{|Ag|}$. Chaque agent i obtient alors la récompense $R_i(s, \mathbf{a})$ et le système passe dans l'état s' en suivant le modèle de transition T. Une politique $\pi_i : S \mapsto \Delta A_i$ pour l'agent i définit une stratégie locale en chaque état au sens de la théorie des jeux. Autrement dit, $\pi_i(s)$ est un vecteur dont les éléments définissent une distribution de probabilité sur les actions du joueur i spécifiques au jeu en forme normale défini par l'état s.

Le terme d'utilité espérée pour un joueur en théorie des jeux désigne l'espérance de récompense (immédiate) sur les *stratégies des joueurs adverses* : $u_i^{(\pi_1, \ldots, \pi_{|Ag|})} = E[R_i(\mathbf{a})]$, alors que la fonction de valeur des MDP est l'espérance *temporelle* de la récompense. Nous emploierons donc le concept d'utilité $U_i(s)$ en jeu stochastique comme l'espérance temporelle des utilités espérées $u_i(s)$ de l'agent i définies pour chaque état s comme pour les utilités espérées des jeux en forme normale

($u_i^{(\pi_1,\ldots,\pi_{|Ag|})}(s) = E_{\mathbf{a}\in\mathbf{A}} R_i(s,\mathbf{a})$). Par conséquent, les utilités $U_i(s)$ des états pour chaque joueur i, associées à la politique conjointe $\pi \equiv \times_{i=1}^{|Ag|}\pi_i$, sont définies comme l'utilité espérée par l'agent i à partir de l'état s si tous les agents suivent cette politique conjointe :

$$\begin{aligned} U_i^{\pi}(s) &= E\left[\sum_{t=0}^{\infty}\gamma^t u_i^{\pi}(s)^t | s^0 = s\right] \\ &= u_i^{\pi}(s) + \gamma \sum_{\mathbf{a}\in\mathbf{A}} \sum_{s'\in S} T(s,\mathbf{a},s')\pi^{\mathbf{a}}(s) U_i^{\pi}(s') \end{aligned}$$

où $\pi^{\mathbf{a}}(s)$ dénote la probabilité de l'action conjointe $\mathbf{a}$ dans l'état s selon la politique conjointe π. Comme pour les MDP, ici aussi s^0 est l'état initial et $\gamma \in [0,1[$ est le facteur d'actualisation.

Comme indiqué plus haut, chaque état d'un jeu stochastique peut être vu – localement – comme un jeu en forme normale. Le processus de transition entre les deux états est illustré par la figure 4.10.

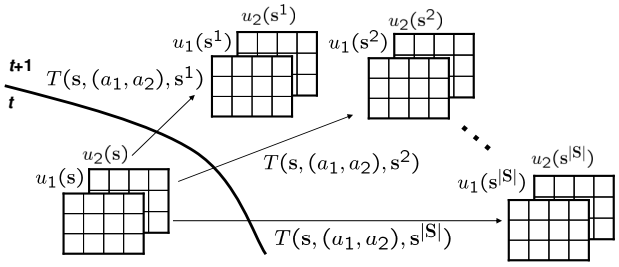

Figure 4.10. *Jeu stochastique à deux joueurs : transitions possibles entre le tour t et le tour $t+1$ lorsque les joueurs jouent l'action conjointe $\mathbf{a} = (a_1, a_2)$; chaque état peut être vu comme un jeu en forme normale.*

Les jeux stochastiques sont un cadre formel permettant de modéliser un environnement multi-agent. Les agents sont *a priori* non coopératifs puisqu'ils poursuivent des objectifs individuels. Ils peuvent cependant être amenés à se coordonner, voire à coopérer, pour atteindre ces buts individuels.

Dans un jeu stochastique, un vecteur de stratégies $(\pi_1^*, \ldots, \pi_{|Ag|}^*)$ est un équilibre de Nash si, pour tout état $s \in S$ et $i = 1, \ldots, |Ag|$:

$$U_i^{(\pi_1^*,\ldots,\pi_{|Ag|}^*)}(s) \geq U_i^{(\pi_1^*,\ldots,\pi_{i-1}^*,\pi_i,\pi_{i+1}^*,\ldots,\pi_{|Ag|}^*)}(s) \qquad \forall \pi_i \in \Pi_i$$

où Π^i est l'ensemble des politiques offertes à l'agent i.

DÉFINITION 4.4.– *Une politique (ou stratégie) dans un jeu stochastique est dite stationnaire (ou markovienne) si et seulement si la règle de décision qui associe une action à un état dépend seulement de l'état courant.*

Le théorème ci-après montre l'existence d'un équilibre de Nash en stratégies stationnaires.

THÉORÈME 4.2 [FIN 64].– *Tout jeu stochastique escompté à n-joueurs ($n \geq 2$) possède au moins un équilibre de Nash en stratégies stationnaires.*

4.3.2. *Résolution des jeux stochastiques*

Nous présentons ici quelques algorithmes de résolution de jeux stochastiques. Comme on a pu le constater au paragraphe précédent, les jeux stochastiques sont des jeux multi-étapes qui peuvent être vus comme une extension des MDP au multi-agent. Etant des jeux, ils n'offrent pas de solution optimale, au sens des MDP, qui soit indépendante des autres joueurs. On va voir que les jeux stochastiques posent essentiellement deux questions différentes :

1) Quelle est, pour le joueur i, la meilleure réponse aux stratégies des autres joueurs $(-i)$?

2) Quelles sont les situations d'équilibre pour ce jeu ?

Si on s'intéresse aux équilibres, le concept le plus utilisé comme solution d'un jeu stochastique est un équilibre de Nash en stratégies stationnaires, dont l'existence a été prouvée. Cependant, contrairement aux MDP, la solution (ou l'équilibre) dans la théorie des jeux n'est pas toujours unique (comme, par exemple, dans le jeu de la figure 4.6 de la section 4.2). Le fait d'avoir plusieurs équilibres avec des valeurs différentes pose généralement des problèmes de coordination. En effet, dans ce cas, si les agents choisissent de jouer des équilibres différents, l'action conjointe jouée peut ne pas constituer un équilibre. Selon le jeu joué par les agents, cela peut constituer une catastrophe si les valeurs des utilités varient beaucoup d'une action conjointe à une autre. Pour simplifier la présentation de ce qui suit, les équilibres sont supposés uniques dans cette section. Le lecteur intéressé par le problème de coordination dans les jeux stochastiques peut se référer à [LIT 94].

Tous les algorithmes proposés dans le cadre des jeux stochastiques que l'on va voir ont la même propriété commune. Ils sont composés de deux parties principales. La première partie peut être vue comme la partie « différences temporelles » pour résoudre la composante « multi-état » du jeu stochastique ; la deuxième partie, quant à elle, est la partie « jeu » pour trouver une solution de la composante « multi-agent » du jeu stochastique.

Selon ces observations, on peut regrouper les algorithmes de jeux stochastiques en quatre catégories (dont le sens sera défini plus loin) en leur donnant les appellations suivantes :

– itération sur les valeurs + théorie des jeux classique ;

– apprentissage par renforcement + théorie des jeux classique ;

– apprentissage par renforcement + modélisation de l'adversaire ;

– apprentissage par renforcement + descente du gradient.

On notera enfin que, si la plupart des approches sont présentées dans le cadre plus simple des jeux répétés, elles sont toutefois utilisables dans le cadre de jeux stochastiques. De même, on peut souvent s'intéresser à un jeu à deux joueurs avant de généraliser à N (sauf dans le cas des jeux à somme nulle).

4.3.2.1. *Itération sur les valeurs + Théorie des jeux classique*

Nous faisons état dans cette sous-section de deux algorithmes élaborés par des chercheurs venant de la communauté de la théorie des jeux, algorithmes ne considérant ici que des jeux à deux joueurs.

Le premier algorithme trouve un équilibre de Nash d'un jeu stochastique à deux joueurs et à somme nulle [SHA 53]. Cet algorithme n'est qu'une extension de la technique d'itération sur les valeurs (vue au chapitre 1) au cas des jeux stochastiques. Pour faire cela, Shapley cherche un équilibre de Nash du jeu en forme normale actuellement associé à cet état. Les fonctions $Equilibre$ et $Valeur$ retournent respectivement un équilibre d'un jeu d'état (une politique conjointe) et son vecteur de valeurs. Pour trouver un équilibre de Nash, Shapley utilise l'algorithme minimax qui a un temps d'exécution polynomial en la taille de la matrice du jeu d'état. La définition complète de l'algorithme de Shapley est donnée par l'algorithme 4.1. Dans ledit algorithme, la valeur $U(s)$ désigne le vecteur d'utilités espérées de tous les agents dans l'état s.

L'algorithme de Shapley est certain de trouver un équilibre dans n'importe quel jeu stochastique à deux joueurs et à somme nulle, car cet algorithme exploite le théorème suivant :

THÉORÈME 4.3 [SHA 53].– *Tout jeu stochastique G à somme nulle escompté à horizon infini possède une valeur unique. Cette valeur est donnée par la séquence des*

Algorithme 4.1 : Algorithme de Shapley [SHA 53] pour les SG à somme nulle

initialiser $U^0(s)$ par des valeurs arbitraires
$n \leftarrow 0$
répéter
  pour $s \in S$ **faire**
    Construire la matrice
    $G(s) = \{g_{\mathbf{a}} : g_{\mathbf{a}} = R(s, \mathbf{a}) + \gamma \sum_{s' \in S} T(s, \mathbf{a}, s') U^n(s')\}$
    $U^{n+1}(s) = Valeur(G(s))$
  $n \leftarrow n + 1$
jusqu'à $\| U^{n+1}(s) - U^n(s) \| < \epsilon \; \forall s$
pour $s \in S$ **faire**
  Construire la matrice $G(s)$
  $\pi(s) = Equilibre(G(s))$
retourner $U^n(s)$, $\pi(s) \; \forall s$

valeurs uniques des jeux d'état $G(s) \forall s$. Chaque joueur du jeu G possède une stratégie optimale qui utilise les stratégies minimax mixtes dans chaque jeu d'état $G(s)$.

Kearns, Mansour et Singh [KEA 00] sont allés plus loin. Ils ont proposé un algorithme similaire à celui de Shapley (à savoir d'itération sur les valeurs) mais leur algorithme, appelé FiniteVI, a été conçu pour les jeux stochastiques à somme générale. Les auteurs ont montré que FiniteVI converge vers un équilibre de Nash à horizon fini. En ce qui concerne les jeux à horizon infini, il a été récemment montré [ZIN 06] qu'il existe une classe de jeux stochastiques pour lesquels aucun algorithme d'itération sur les valeurs basé sur l'application directe de l'équation de Bellman (ce qui est fait, par exemple, dans FiniteVI) peut ne pas converger vers une politique stationnaire.

La structure de l'algorithme FiniteVI est la même que celle de celui de Shapley. Il se différencie toutefois par le fait que 1) l'horizon T est utilisé comme critère d'arrêt ; et 2) les fonctions *Equilibre* et *Valeur* sont définies différemment.

Tandis que l'équilibre minimax utilisé par l'algorithme de Shapley est assuré être unique, ce n'est plus le cas dans le cadre des jeux à somme générale ; par contre, il peut y avoir plusieurs équilibres possédant des valeurs différentes. Dès lors, comme un seul équilibre doit être choisi à chaque itération, Kearns et ses collègues ont proposé d'utiliser une fonction $f(G(s))$ choisissant un seul équilibre parmi plusieurs. Cependant, cette fonction $f(G(s))$ n'a pas été définie, ce qui pose le problème d'application directe de FiniteVI. Il convient de noter également qu'à l'inverse du cas « à somme nulle », il n'existe pas d'algorithme polynomial trouvant un équilibre de Nash dans un jeu à somme générale.

Parmi d'autres algorithmes d'itération sur les valeurs pour les jeux stochastiques, le lecteur peut se référer à [POL 69, HOF 66, BOW 03a].

4.3.2.2. *Apprentissage par renforcement + théorie des jeux classique*

Les algorithmes de cette catégorie combinent les techniques de recherche d'équilibre dans les jeux en forme normale avec les techniques d'apprentissage par renforcement comme le Q-learning vu au paragraphe 2.4.3 du chapitre 2.

L'idée sous-tendant les algorithmes de ce type est la suivante. Les agents réalisent un apprentissage par renforcement en faisant des actions simultanées et en recevant des récompenses. Après chaque tour du jeu (ou après chaque séquence « action conjointe–transition ») les agents mettent à jour leurs fonctions Q qui associent des valeurs réelles à des couples « action conjointe–état » :

$$Q_i(s, \mathbf{a}) \leftarrow Q_i(s, \mathbf{a}) + \alpha \left(R_i(s, \mathbf{a}) + \gamma Valeur_i(s') - Q_i(s, \mathbf{a}) \right) \qquad (4.2)$$

où la fonction $Valeur_i$ retourne la valeur d'un équilibre d'un jeu G composé des Q-valeurs des agents dans l'état s'. On voit bien ici que, pour pouvoir calculer cette fonction, l'agent i a besoin d'observer les actions et les récompenses de tous les autres agents dans l'environnement ou que les autres agents doivent lui communiquer cette information.

Dans l'algorithme Minimax-Q de Littman [LIT 94] (pour jeux à deux joueurs et à somme nulle), utilisant ce principe, la fonction $Valeur_i$ retourne au joueur i la valeur minimax d'un jeu composé des Q-valeurs des agents. La politique suivie par les agents est alors la politique de minimax en stratégies mixtes.

Quant à l'algorithme Nash-Q de Hu et Wellman [HU 03] (applicable pour des jeux à somme générale et à N joueurs), la fonction $Valeur_i$ proposée par les auteurs retourne la valeur d'un équilibre de Nash du jeu d'état. Le fait d'avoir choisi le même équilibre pour tous les agents est assuré par une politique de coordination qui prescrit aux agents de choisir toujours un équilibre prédéfini, par exemple le premier. La politique suivie par les agents dans ce cas est alors la politique correspondant à l'équilibre de Nash choisi.

Une définition plus formelle de l'algorithme de base pour les algorithmes Nash-Q et Minimax-Q est représentée par l'algorithme 4.2. Comme c'était déjà le cas des algorithmes de Shapley et de Kearns, la différence entre les deux algorithmes réside dans la définition des fonctions $Equilibre_i$ et $Valeur_i$. Dans le cas de Minimax-Q, ces deux fonctions retournent à l'agent i respectivement la composante π_i de la politique d'équilibre minimax dans l'état s et la valeur de cet équilibre ; tandis que, dans Nash-Q, elles retournent la politique d'un équilibre de Nash et sa valeur (son vecteur de valeur dans Nash-Q) pour l'agent i.

Algorithme 4.2 : Algorithme de la base pour les algorithmes Minimax-Q et Nash-Q pour un joueur i

pour tous les s, $\mathbf{a}$ *et* $j \in Ag$ **faire** Initialiser $Q_j(s, \mathbf{a})$ par des valeurs arbitraires.
Mettre dans s l'état courant.
Construire le jeu $G(s)$ à partir des valeurs Q_j dans s, $\forall j \in Ag$.
Choisir une politique $\pi_i(s) = Equilibre_i(G(s))$.
répéter
  Jouer la politique $\pi_i(s)$ avec une certaine exploration.
  Observer l'action conjointe et la mettre dans $\mathbf{a}$.
  Observer les récompenses de chacun et les mettre dans $\mathbf{r}$.
  Observer le nouvel état et le mettre dans s'.
  Construire le jeu $G(s')$.
  Choisir une politique $\pi_i(s') = Equilibre_i(G(s'))$.
  pour chaque *joueur* j **faire** Mettre à jour la valeur $Q_j(s, \mathbf{a})$ en utilisant l'équation (4.2).
  $s \leftarrow s'$, $t \leftarrow t + 1$.
jusqu'à $t > T$
pour $s \in S$ **faire**
  Construire la matrice $G(s)$ comme précédemment.
  $\pi_i(s) = Equilibre_i(G(s))$.
retourner $\pi_i(s)$ $\forall s$.

Dans l'algorithme 4.2, le terme « une certaine exploration » fait référence aux différentes techniques d'exploration discutées au chapitre 2.

Il convient de noter que les preuves de convergence des deux précédents algorithmes se trouvent dans [LIT 94] pour le premier et dans [HU 03] pour le second. Dans le cas de ce dernier, la preuve est faite sous certaines restrictions assez contraignantes.

4.3.2.3. *Apprentissage par renforcement + modélisation de l'adversaire*

Les algorithmes de cette catégorie combinent également deux parties provenant de deux champs de recherches distincts : celui de l'apprentissage mono-agent et celui de la théorie des jeux dynamiques dont la technique de modélisation de l'adversaire fait partie. Si tout est assez clair avec la première partie – un algorithme des « différences temporelles » est utilisé pour traiter le côté « multi-état » d'un jeu stochastique (voir la section 2.4 du chapitre 2) – il reste à préciser ce qu'on entend par « modélisation de l'adversaire ».

La modélisation de l'adversaire est une technique largement utilisée dans la théorie des jeux dynamiques pour rendre chaque joueur capable de s'adapter à ses adversaires et à leurs éventuels changements de politique [CLA 98, UTH 03].

Le premier effort fait en ce sens dans le cadre d'un jeu répété est dû à Brown [BRO 51]. La technique utilisée sous le nom de « jeu fictif » (*fictitious play*) permet à un joueur d'estimer la stratégie jouée par son adversaire afin de jouer la meilleure réponse à cette estimation.

Dans le jeu fictif, les joueurs jouant un jeu répété maintiennent des croyances empiriques individuelles sur les stratégies suivies par les autres joueurs. Le modèle du jeu fictif suppose que chaque joueur i choisit ses actions à chaque période pour maximiser son utilité espérée, $u_i^{\hat{\pi}_{-i}}$, étant donnée son estimation des politiques mixtes de chacun de ses adversaires, $\hat{\pi}_j$. Cette estimation prend la forme suivante. On suppose que le joueur i a, pour chaque autre joueur j, une fonction de poids initiale qui serait $c_{ij}^0 : A_j \mapsto \mathbb{R}_+$. Ce poids est mis à jour en ajoutant 1 au poids d'une stratégie adverse a_j lorsque celle-ci est jouée par l'adversaire j :

$$c_{ij}^t(a_j) = c_{ij}^{t-1}(a_j) + \begin{cases} 1 & \text{si } a_j^{t-1} = a_j, \\ 0 & \text{sinon.} \end{cases}$$

Dans ces conditions, la probabilité estimée par le joueur i que son adversaire j joue une certaine action a_j à la date t est donnée par :

$$(\hat{\pi}_j^{a_j})^t = \frac{c_{ij}^t(a_j)}{\sum_{a'_j} c_{ij}^t(a'_j)}.$$

Dès lors, la meilleure action à jouer pour l'agent i est celle qui maximise son utilité espérée étant donnée son estimation des politiques adverses (en d'autres mots, sa meilleure réponse à la politique jointe correspondante) :

$$a_i^t = \operatorname*{argmax}_{a_i} \sum_{\mathbf{a}_{-i}} R_i(a_i, \mathbf{a}_{-i})(\hat{\pi}_{-i}^{\mathbf{a}_{-i}})^t.$$

Le jeu fictif converge vers un équilibre de Nash dans les jeux appelés *iterated dominance solvable*, c'est-à-dire les jeux dans lesquels il est possible d'enlever les actions dominées de façon itérative pour obtenir à la fin une seule action ou un ensemble d'actions équivalentes [FUD 99].

La version du jeu fictif adaptée aux jeux stochastiques, étant donnée la fonction de transition $T(s, \mathbf{a}, s')$, est présentée sur l'algorithme 4.3 ; elle est due à Vrieze [VRI 87].

Un algorithme similaire proposé par Gies et Chaib-draa [GIE 06] est appelé « *Q-learning par jeu adaptatif* ». L'approche est basée sur la technique du jeu adaptatif pour les jeux répétés proposée par Young [YOU 93]. L'algorithme de Young est très similaire au jeu fictif. La différence réside dans la manière d'estimer la politique de l'adversaire. Tandis que, dans le jeu fictif, tout l'historique est pris en compte, dans le jeu adaptatif l'agent fait un échantillonnage à partir d'un historique récent de taille

Algorithme 4.3 : Algorithme du jeu fictif dans les SG pour un joueur i

pour tous les s, a_i **faire** Initialiser
$q_i(s, a_i) \leftarrow \frac{1}{|\mathbf{A}_{-i}|} \sum_{\mathbf{a}_{-i} \in \mathbf{A}_{-i}} R_i(s, (a_i, \mathbf{a}_{-i}))$.
$n \leftarrow 0$.
répéter
  pour tous les s **faire**
    Choisir une action $a_i^n = \text{argmax}_{a_i'} \frac{q_i(s,a_i')}{n}$.
    Jouer l'action a_i^n.
    Observer l'action conjointe et la mettre dans $\mathbf{a}$.
    pour tous les a_i **faire**
      Mettre à jour la valeur $q_i(s, a_i)$:
      $q_i(s, a_i) \leftarrow$
        $q_i(s, a_i) + R_i(s, (a_i, \mathbf{a}_{-i})) + \gamma \sum_{s' \in S} T(s, \mathbf{a}, s') Valeur(s')$
        où $Valeur(s') = \max_{a_i} \frac{q_i(s,a_i)}{n}$
  $n \leftarrow n + 1$.
jusqu'à $n > N$
pour tous les s **faire** $\pi_i^{a_i}(s) \leftarrow 1$ si $a_i = a_i^N$ et $\pi_i^{a_i}(s) \leftarrow 0$ sinon.
retourner $\pi_i(s)$ $\forall s$.

limitée. Une telle modification permet de prouver la convergence du jeu adaptatif pour une plus large classe des jeux de coordination appelés *weakly acyclic games* [YOU 98].

Une autre approche proposée par Claus et Boutilier [CLA 98], appelée « *Joint Action Learners* » ou JALs,[5] est basée sur le Q-learning et par conséquent ne dépend pas du modèle. De plus, comme c'est une technique essai-erreur, elle ne nécessite aucun mécanisme de coordination des agents. Notons que cette technique ressemble beaucoup à celle du jeu fictif à deux différences près. La première réside dans le fait que les Q-valeurs dans JALs sont associées aux couples « état–action conjointe » au lieu de « état–action simple » du jeu fictif. La deuxième différence réside dans l'équation utilisée pour mettre à jour les Q-valeurs. Claus et Boutilier utilisent la mise à jour usuelle du Q-learning (algorithme 4.4) au lieu d'une forme de l'équation de Bellman telle qu'utilisée dans le jeu fictif.

5. JALs était initialement prévue pour des agents coopératifs, mais a été employé dans d'autres situations [UTH 03].

Algorithme 4.4 : Algorithme JALs pour les SG pour un joueur i proposé par [CLA 98], adapté de [BOW 02a]

pour tous les s *et* $\mathbf{a}_{-i}$ **faire** Initialiser $Q_i(s, \mathbf{a}_{-i})$ arbitrairement,
$c_i(s, \mathbf{a}_{-i}) \leftarrow 0$ et $c(s) \leftarrow 0$.
Initialiser $n \leftarrow 0$, $s \leftarrow s^0$.
répéter
  Choisir une action $a_i^n = \text{argmax}_{a_i} \sum_{\mathbf{a}_{-i}} \frac{c_i(s,\mathbf{a}_{-i})}{c(s)} Q_i(s, (a_i, \mathbf{a}_{-i}))$.
  Jouer l'action a_i^n avec un certain bruit d'exploration.
  Observer l'action conjointe des autres agents, $\mathbf{a}_{-i}$.
  Observer la récompense, $R_i(s, (a_i, \mathbf{a}_{-i}))$.
  Observer le nouvel état, s'.
  $Q_i(s, (a_i, \mathbf{a}_{-i}))$
    $\leftarrow (1-\alpha) Q_i(s, (a_i, \mathbf{a}_{-i})) + \alpha \left(R(s, (a_i, \mathbf{a}_{-i})) + \gamma Valeur(\mathbf{s}')\right)$
    où $Valeur(s') = \max_{a_i} \sum_{\mathbf{a}_{-i}} \frac{c_i(s',\mathbf{a}_{-i})}{n(s')} Q_i(s', (a_i, \mathbf{a}_{-i}))$.
  $c(s) \leftarrow c(s) + 1$, $c_i(s, \mathbf{a}_{-i}) \leftarrow c_i(s, \mathbf{a}_{-i}) + 1$, $n \leftarrow n + 1$.
jusqu'à $n > N$
pour tous les s **faire**
  si $a_i = \text{argmax}_{a_i} \sum_{\mathbf{a}_{-i}} \frac{c_i(s,\mathbf{a}_{-i})}{c(s)} Q_i(s, (a_i, \mathbf{a}_{-i}))$ **alors**
    $\pi_i^{a_i}(s) \leftarrow 1$,
  sinon
    $\pi_i^{a_i}(s) \leftarrow 0$.
retourner $\pi_i(s)$ $\forall s$.

4.3.2.4. *Apprentissage par renforcement + descente de gradient*

La technique de la descente de gradient (voir le chapitre **??** du volume 2) a été tout d'abord étudiée dans l'apprentissage multi-agent par Singh *et al.* [SIN 94]. Leur algorithme, IGA (pour *Infinitesimal Gradient Ascent*), suppose que chaque joueur d'un jeu répété effectue une montée de gradient dans l'espace de ses politiques pour trouver la politique qui maximise son utilité escomptée (chaque agent ayant son propre critère). Cet algorithme a été étudié dans le cadre d'un jeu à deux joueurs et deux actions par joueur, représenté par deux matrices de récompenses pour les joueurs ligne et colonne, l et c, comme suit :

$$R_l = \begin{bmatrix} l_{11} & l_{12} \\ l_{21} & l_{22} \end{bmatrix}, \quad R_c = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}.$$

Les joueurs l et c choisissent simultanément une action de l'ensemble $A_{l,c} = \{1, 2\}$, le joueur ligne l joue une action i et le joueur colonne c choisit une action j. Les récompenses obtenues sont alors l_{ij} et c_{ij} respectivement.

Comme il s'agit d'un jeu à deux actions par joueur, une stratégie mixte d'un joueur peut être représentée avec une seule valeur. Soit $\alpha \in [0,1]$ la probabilité avec laquelle le joueur l choisit l'action 1 et $(1-\alpha)$ la probabilité de jouer l'action 2. De façon similaire, soit $\beta \in [0,1]$ et $(1-\beta)$ les probabilités pour le joueur c de jouer les actions 1 et 2 respectivement. L'utilité espérée d'une stratégie $\pi = (\alpha, \beta)$ peut être alors calculée comme suit :

$$\begin{aligned} u_l^{(\alpha,\beta)} &= l_{11}\alpha\beta + l_{22}(1-\alpha)(1-\beta) + l_{12}\alpha(1-\beta) + l_{21}(1-\alpha)\beta, \\ u_c^{(\alpha,\beta)} &= c_{11}\alpha\beta + c_{22}(1-\alpha)(1-\beta) + c_{12}\alpha(1-\beta) + c_{21}(1-\alpha)\beta. \end{aligned}$$

Pour estimer l'effet d'un changement de sa politique courante, un joueur peut calculer la dérivée partielle de son utilité espérée par rapport à sa stratégie mixte :

$$\begin{aligned} \frac{\partial u_l^{(\alpha,\beta)}}{\partial \alpha} &= \beta u - (l_{22} - l_{12}), \\ \frac{\partial u_c^{(\alpha,\beta)}}{\partial \beta} &= \alpha u' - (c_{22} - c_{21}), \end{aligned}$$

où $u = (l_{11} + l_{22}) - (l_{21} + l_{12})$ et $u' = (c_{11} + c_{22}) - (c_{21} + c_{12})$.

A chaque pas de temps, le joueur IGA ajuste sa stratégie courante dans la direction du gradient en vue de maximiser son utilité espérée :

$$\begin{aligned} \alpha_{t+1} &= \alpha_t + \eta \frac{\partial u_l^{(\alpha_t,\beta_t)}}{\partial \alpha}, \\ \beta_{t+1} &= \beta_t + \eta \frac{\partial u_c^{(\alpha_t,\beta_t)}}{\partial \beta}, \end{aligned}$$

où η est la taille d'un pas, typiquement $0 < \eta \ll 1$. Evidemment, la stratégie mixte de l'adversaire est supposée connue par les joueurs.

Singh et ses collègues ont prouvé la convergence de IGA vers un équilibre de Nash ou vers une valeur moyenne équivalente – les deux stratégies continuant à évoluer en décrivant une trajectoire elliptique si on trace la courbe $x = \alpha(t), y = \beta(t)$ – en *self-play* (c'est-à-dire, quand les deux joueurs utilisent le même algorithme) et ce dans le cas où le pas η tend vers 0 ($\lim_{\eta \to 0}$), d'où le qualificatif « infinitésimal » dans le nom de l'algorithme. L'algorithme IGA n'est pas voué à être appliqué à un grand nombre de problèmes du monde réel, et ce pour les deux raisons principales suivantes :

1) il suppose une connaissance omnisciente des stratégies courantes d'autrui et,

2) il a été conçu pour le cas « deux agents–deux actions » ; une extension directe au cas « plusieurs agents–plusieurs actions » n'est pas évidente.

Le premier algorithme « pratique » héritant (en pratique) des propriétés de convergence de IGA est l'algorithme du *Policy Hill Climbing* (PHC) proposé par Bowling et Veloso [BOW 02a] (dans le cadre des jeux stochastiques). Cet algorithme n'exige pas la connaissance de la politique courante de l'adversaire. Pour l'essentiel, PHC réalise une ascension locale (*hill-climbing*) dans l'espace des stratégies mixtes. De ce fait, c'est une simple modification du Q-learning mono-agent (pour rechercher une politique stochastique, c'est-à-dire une stratégie mixte). Il comporte deux parties : la première est basée sur le Q-learning en vue de maintenir les valeurs des actions simples (et non des actions conjointes) dans les états ; la deuxième est une partie de théorie des jeux qui maintient la stratégie mixte courante dans chacun des états du système.

Dans PHC, la politique courante est ajustée *via* une augmentation de la probabilité de choisir l'action ayant la valeur la plus élevée. Ceci est fait en utilisant un petit pas δ. Les Q-valeurs sont mises à jour en utilisant le facteur d'apprentissage α. Si δ égale à 1, l'algorithme devient équivalent au Q-learning mono-agent, car l'agent exécutera alors toujours l'action dont la valeur est la plus élevée. Cet algorithme fournit un joueur dit « rationnel » parce que, si les autres joueurs convergent vers des politiques stationnaires, alors sa stratégie converge vers la meilleure réponse à ces stratégies. Toutefois, si les autres joueurs sont en train d'apprendre leurs politiques, PHC peut ne pas converger vers une politique stationnaire bien que sa récompense moyenne au cours du temps converge vers la récompense d'un équilibre de Nash, comme cela a été montré par Bowling et Veloso [BOW 02a].

La définition formelle de l'algorithme PHC est présentée via l'algorithme 4.5. Dans cet algorithme (et ceux qui suivent plus bas), l'expression « une certaine exploration » fait référence aux différentes techniques d'exploration des espaces d'états ou d'actions, telles que ϵ-greedy et autres (voir le chapitre 2).

Bowling et Veloso [BOW 02a] ont développé des extensions importantes des algorithmes IGA et PHC, appelées respectivement WoLF-IGA et WoLF-PHC (le dernier est illustré par l'algorithme 4.6). WoLF (pour *Win or Learn Fast*) est une technique selon laquelle un agent i change le pas d'ajustement de la politique δ selon qu'il « gagne » ou qu'il « perd ». S'il gagne, son δ est petit ; s'il perd, il est grand. Bowling a montré de façon formelle qu'une telle alternance assure la convergence de WoLF-IGA en self-play vers un équilibre de Nash dans le cas où il y a deux joueurs qui ont deux actions chacun. En pratique, la convergence de WoLF-PHC a aussi été observée dans plusieurs jeux « problématiques ». Notons que le principe WoLF aide la convergence en donnant aux autres joueurs plus de temps pour s'adapter aux changements de stratégie du joueur considéré. Réciproquement, il permet à un joueur de s'adapter plus

Algorithme 4.5 : Algorithme PHC pour les SG pour un joueur i, adapté de [BOW 02a]

Initialiser $\alpha \in (0,1]$, $\delta \in (0,1]$, $\gamma \in (0,1)$.
pour tous les $s \in S$ *et* $a_i \in A_i$ **faire** Initialiser $q_i(s,a_i) \leftarrow 0$.
pour tous les $a_i \in A_i$ **faire** Initialiser la politique courante $\pi_i^{a_i}(s) \leftarrow \frac{1}{|A_i|}$.
L'état courant $s \leftarrow s^0$.
répéter
Choisir une action a_i selon la stratégie $\pi_i(s)$.
Jouer a_i avec une certaine exploration.
Observer le nouvel état s' et la récompense obtenue R_i.
Mettre à jour $q_i(s,a_i)$ en utilisant la règle suivante :

$$q_i(s,a_i) \leftarrow (1-\alpha) q_i(s,a_i) + \alpha \left(R_i + \gamma \max_{a_i'} q_i(s',a_i') \right).$$

Mettre à jour la stratégie courante, $\pi_i(s)$, en utilisant la règle suivante :

$$\begin{aligned} \pi_i^{a_i}(s) &\leftarrow \pi_i^{a_i}(s) + \Delta_{sa_i}, \text{ où} \\ \Delta_{sa_i} &= \begin{cases} -\delta_{sa_i} & \text{si } a_i \neq \operatorname{argmax}_{a_i'} q_i(s,a_i') \\ \sum_{a_i' \neq a_i} \delta_{sa_i'} & \text{sinon ;} \end{cases} \\ \text{avec} \quad \delta_{sa_i} &= \min \left(\pi_i^{a_i}(s), \frac{\delta}{|A_i|-1} \right), \end{aligned}$$

en se limitant à la distribution de probabilité légale.
Mettre à jour l'état courant $s \leftarrow s'$.
$n \leftarrow n+1$.
jusqu'à $n > N$
retourner $\pi_i(s)$ $\forall s$.

rapidement aux changements des stratégies des autres joueurs quand ces changements sont défavorables.

D'une façon plus formelle, l'algorithme WoLF-PHC exige d'utiliser deux valeurs du pas δ : δ_{perdre} et δ_{gagner}. Le fait de gagner ou perdre est déterminé en comparant la valeur espérée de la politique courante, π, avec la valeur espérée d'une « politique moyenne », $\bar{\pi}$ (voir l'algorithme 4.6). Cette politique moyenne est une moyenne « en ligne » des politiques du joueur i depuis le début de l'apprentissage. Si la valeur espérée de π est inférieure à celle de $\bar{\pi}$, alors le joueur est considéré comme perdant et la valeur de δ_{perdre} est utilisée, sinon δ_{gagner} est utilisée pour ajuster sa politique. Pour le reste, l'algorithme WoLF-PHC est identique à PHC (voir l'algorithme 4.5).

Algorithme 4.6 : Algorithme WoLF-PHC pours les SG pour un joueur i, adapté de [BOW 02a].

Initialiser $\alpha \in (0,1]$, $\delta \in (0,1]$, $\gamma \in (0,1)$.
pour tous les $s \in S$ *et* $a_i \in A_i$ **faire** Initialiser $q_i(s,a_i) \leftarrow 0$ et $c(s) \leftarrow 0$.
pour tous les $a_i \in A_i$ **faire** Initialiser la politique courante $\pi_i^{a_i}(s) \leftarrow \frac{1}{|A_i|}$.
Initialiser l'état courant $s \leftarrow s^0$.
répéter
Choisir une action a_i selon la stratégie $\pi_i(s)$.
Jouer a_i avec une certaine exploration.
Observer le nouvel état s' et la récompense obtenue R_i.
Mettre à jour $q_i(s,a_i)$ en utilisant la règle suivante :

$$q_i(s,a_i) \leftarrow (1-\alpha) q_i(s,a_i) + \alpha \left(R_i + \gamma \max_{a_i'} q_i(s',a_i') \right).$$

Mettre à jour l'estimation de la politique moyenne, $\bar{\pi}(s)$, comme suit :

$$\begin{aligned} c(s) &\leftarrow c(s)+1, \\ \bar{\pi}_i^{a_i'}(s) &\leftarrow \bar{\pi}_i^{a_i'}(s) + \frac{1}{c(s)}(\pi_i^{a_i'}(s) - \bar{\pi}_i^{a_i'}(s)),\ \forall a_i' \in \mathcal{A}_i. \end{aligned}$$

Mettre à jour la stratégie courante, $\pi_i(s)$, en utilisant la règle suivante :

$$\begin{aligned} \pi_i^{a_i}(s) &\leftarrow \pi_i^{a_i}(s) + \Delta_{sa_i}, \text{ où} \\ \Delta_{sa_i} &= \begin{cases} -\delta_{sa_i} & \text{si} \quad a_i \neq \operatorname{argmax}_{a_i'} q_i(s,a_i'), \\ \sum_{a_i' \neq a_i} \delta_{sa_i'} & \text{sinon,} \end{cases} \\ \text{avec} \quad \delta_{sa_i} &= \min\left(\pi_i^{a_i}(s), \frac{\delta}{|A_i|-1} \right), \text{et} \\ \delta &= \begin{cases} \delta_{gagner} & \text{si} \quad \sum_{a_i'} \pi_i^{a_i'}(s) q_i(s,a_i') > \sum_{a_i'} \bar{\pi}_i^{a_i'}(s) q_i(s,a_i'), \\ \delta_{perdre} & \text{sinon,} \end{cases} \end{aligned}$$

en se limitant à la distribution de probabilité légale.
Mettre à jour l'état courant $s \leftarrow s'$.
$n \leftarrow n+1$.
jusqu'à $n > N$
retourner $\pi_i(s)\ \forall s$.

4.3.3. *Complexité et extensibilité des algorithmes d'apprentissage multi-agent*

Cette section fait état de la complexité algorithmique de certains algorithmes d'apprentissage multi-agent et leur extensibilité (leur capacité de passage à l'échelle[6] à des problèmes de grande taille.

Comme on l'a précisé plus haut, l'algorithme de Shapley [SHA 53] pour les jeux stochastiques (à deux joueurs et à somme nulle), qui s'appuie sur la technique d'itération sur les valeurs, utilise l'algorithme minimax pour trouver l'équilibre dans chaque état et à chaque itération. Puisque le temps d'exécution de l'algorithme d'itération sur les valeurs est lui-même polynomial en la taille de l'espace d'états, alors le temps d'exécution de l'algorithme de Shapley est polynomial en le nombre de joueurs et le nombre d'états du jeu stochastique.

Le même raisonnement peut être appliqué pour conclure sur le temps d'exécution de l'algorithme de Kearns *et al.* [KEA 00]. Le temps d'exécution de l'algorithme de Kearns est quadratique en la taille de l'espace d'états en supposant que le temps d'exécution de la fonction f (censée trouver un équilibre de Nash) est unitaire. Cependant, comme on ne connaît pas d'algorithme polynomial trouvant un équilibre de Nash dans un jeu matriciel, le temps d'exécution de l'algorithme de Kearns, utilisant un des algorithmes connus permettant de calculer un équilibre de Nash, ne peut être polynomial non plus.

L'analyse du temps d'exécution des algorithmes basés sur le Q-learning est plus compliqué. On sait [KOE 96] que le temps d'exécution du Q-learning peut être exponentiel en la taille de l'espace d'états, mais ce temps peut être réduit à un polynôme si certaines restrictions sur le modèle de récompense sont appliquées [KOE 96]. De plus, les tailles des espaces d'état (et d'action conjointe) sont elles-mêmes généralement exponentielles en le nombre d'agents. Par conséquent, le temps d'exécution d'un tel algorithme comme Nash-Q n'est pas polynomial en la taille de la matrice du jeu d'état (en raison du fait que la procédure utilisée pour calculer un équilibre de Nash est elle-même non polynomiale). De plus, ce temps peut être exponentiel en le nombre d'états (en raison de la structure de la fonction de récompense utilisée dans la partie « Q-learning » de cet algorithme).

Comme on peut le constater, l'application directe aux problèmes de grande taille des algorithmes de planification ou d'apprentissage multi-agents basés sur le modèle des jeux stochastiques est problématique à cause de leur complexité élevée. Une exception est l'algorithme de Shapley qui a un temps d'exécution polynomial mais qui ne s'applique qu'à des jeux à deux joueurs et à somme nulle.

---

6. Pour l'anglais *scalability*.

Un bon candidat à une application dans les problèmes de grande taille est l'algorithme WoLF-PHC de Bowling [BOW 02a]. Vu sa simplicité structurelle, ses conditions de convergence tolérantes et ses exigences modestes relativement à l'information provenant de l'environnement (en fait, les agents WoLF-PHC doivent percevoir seulement l'état courant et leurs propres récompenses), les techniques connues d'approximation de fonction peuvent être directement appliquées à cet algorithme. Dans ce contexte, Bowling [BOW 02b] a proposé de combiner : (i) une technique appelée *Tile Coding* [SUT 98] pour généraliser la fonction de valeur,[7] (ii) une méthode de montée du gradient[8] de la politique [SUT 00] comme méthode d'apprentissage de base (au lieu du Q-learning) et (iii) le principe WoLF pour favoriser la convergence vers un équilibre de Nash. Cette méthode a permis d'obtenir un algorithme d'apprentissage multi-agent facilement extensible appelé GraWoLF [BOW 02b]. Bowling a montré *via* des expérimentations que GraWoLF peut être utilisé pour faire de l'apprentissage multi-agent dans les problèmes de très grande taille, comme le jeu de cartes Goofspiel [BOW 02b] et l'entraînement des robots compétitifs [BOW 03b] de type RoboCup [KIT 97].

4.3.4. *Au-delà de la recherche d'équilibre*

Dans les travaux cités précédemment, la préoccupation principale était la recherche de situations d'équilibre (principalement de Nash). Ces derniers temps, les chercheurs s'interrogent sur la nécessité de la convergence d'un algorithme d'apprentissage vers un équilibre de Nash. Il existe plusieurs raisons à cela. Tout d'abord, il peut y avoir plusieurs équilibres dans un jeu et il peut ne pas y avoir de méthode de coordination des choix des agents. Ensuite, la complexité de calcul d'équilibres de Nash est peu étudiée et on ne connaît pas d'algorithme polynomial pouvant calculer un tel équilibre.

La troisième raison est plus philosophique. Certains auteurs [SHO 04] attirent l'attention sur le fait que, dans plusieurs cas, comme par exemple dans le dilemme du prisonnier, jouer l'équilibre de Nash peut être catastrophique pour les agents, tandis que l'action coopérative, bien qu'elle ne soit pas un équilibre, serait un choix plus « judicieux ». Autrement dit, l'action coopérative est le seul choix rationnel, comme on l'a vu dans le cas du dilemme du prisonnier répété.

Enfin, un joueur préfèrera, s'il le peut, exploiter les faiblesses de ses adversaires.

4.3.4.1. *Jeu efficace*

Certains travaux récents ont mis l'accent sur le fait de jouer « plus efficacement » contre un certain autre type de joueur et non pas le fait de converger vers une valeur ou

7. Voir le chapitre **??** du volume 2.

8. Voir le chapitre **??** du volume 2.

une politique stable, comme on le fait usuellement. Parmi ces approches, il convient de citer les travaux suivants [CHA 02, POW 05b, POW 05a, TES 04].

Chang et Kaelbling [CHA 02] ont été les premiers à proposer une méthode permettant à un joueur d'exploiter l'algorithme d'apprentissage de son adversaire s'il a une certaine forme (dans des jeux répétés). En particulier, leur algorithme appelé « PHC-Exploiter » peut l'emporter plus de fois que son adversaire dans les jeux à somme nulle, tels que Pierre-Papier-Ciseaux et ce si l'adversaire utilise l'algorithme PHC [BOW 02a]. En fait, PHC-Exploiter est capable d'estimer la valeur du pas d'apprentissage δ de l'algorithme PHC de l'adversaire et changer sa politique au bon moment afin d'exploiter cette connaissance.

Tesauro [TES 04] est allé plus loin. Il a proposé une technique qui semble être plus générale que celle de Chang et Kaelbling. En effet, l'algorithme Hyper-Q de Tesauro peut jouer de façon plus efficace dans les jeux à somme nulle contre un joueur sans connaître l'algorithme employé par ce dernier. Par exemple, Tesauro a montré *via* les expérimentations que l'algorithme Hyper-Q est plus efficace que PHC et IGA dans le jeu Pierre-Papier-Ciseaux. L'idée sous-tendant l'approche de Tesauro est la suivante. L'agent Hyper-Q discrétise l'espace des stratégies de son adversaire de façon uniforme. Il associe ensuite une Q-valeur à chacune de ses actions simples et à chacune des valeurs discrètes de la stratégie adverse. Pour estimer la stratégie de son adversaire, il utilise une technique d'estimation de distribution de probabilité. Finalement, en interagissant avec l'adversaire, l'agent Hyper-Q apprend des Q-valeurs de chaque couple « stratégie de l'adversaire–action simple ». Il convient de noter qu'il n'a pas été proposé d'extension de cet algorithme aux jeux stochastiques. Une approche similaire à celle de Tesauro, appelée « ADL » (pour *Adaptive Dynamics Learning*) a été proposée par Burkov et Chaib-draa [BUR 07]. La différence de cette dernière par rapport à Hyper-Q réside dans la manière d'assigner des Q-valeurs ; dans ADL, ces dernières sont assignées à des historiques de taille limitée à la place des distributions de probabilités. Alors les Q-valeurs de ADL ont la forme « historique du jeu–action simple ». Pour apprendre ces valeurs, on utilise la règle standard de mise à jour du Q-learning.

4.3.4.2. *Minimisation du regret*

Une autre direction de recherche [HAR 00, AUE 95, ZIN 03] vise à apporter des réponses à la question suivante. Etant donné un historique du jeu, dans quelle mesure la politique exécutée par l'agent (par un algorithme d'apprentissage ou celui réalisant une politique fixe) pourrait-elle être améliorée ? Plus précisément, il s'agit d'utiliser la notion de « regret » mesurant jusqu'à quel degré la performance observée d'un algorithme est pire que la meilleure stratégie pure.

Décrivons cette notion de regret de façon plus formelle et limitons-nous au cas des jeux répétés. Soit $\mathbf{r}_i^t \in \mathbb{R}^{|A_i|}$ le vecteur des récompenses que le joueur i pourrait

obtenir au temps t sachant les choix faits au même temps par les autres joueurs. Soit π_i^t la stratégie adoptée par le joueur i au temps t. L'utilité espérée u_i^t de stratégie π_i^t est alors la suivante :

$$u_i^t = \sum_{a_i \in A_i} \pi_i^{a_i,t} r_i^{a_i,t},$$

où $r_i^{a_i,t}$ désigne la récompense de i pour l'action a_i selon $\mathbf{r}_i^t$. En utilisant le produit scalaire de deux vecteurs, on peut aussi écrire :

$$u_i^t = \pi_i^t \cdot \mathbf{r}_i^t.$$

Au pas de temps t, le regret $\mathcal{R}_i^t$ de l'agent i pour avoir joué une politique π_i^t au lieu d'une action simple a_i est la différence entre les récompenses de ces deux stratégies, étant donné le choix des stratégies des adversaires :

$$\mathcal{R}_i^t(\pi_i^t, a_i) = r_i^{a_i,t} - u_i^t.$$

En désignant par 1_{a_i} une politique de l'agent i dans laquelle la probabilité de 1 est assignée à une action a_i, le regret total $\mathcal{R}_i^T$ de l'agent i pour une séquence de tours du jeu de la taille T s'écrit :

$$\mathcal{R}_i^T = \max_{a_i \in A_i} \sum_{t=1}^{T} \left((\mathbf{r}_i^t \cdot 1_{a_i}) - (\mathbf{r}_i^t \cdot \pi_i^t) \right).$$

Le regret moyen $\bar{\mathcal{R}}_i$ du même algorithme s'écrit alors comme :

$$\bar{\mathcal{R}}_i = \lim_{T \to \infty} \frac{1}{T} \mathcal{R}_i^T.$$

Pour un algorithme donné, la propriété de ne pas avoir de regret (*no-regret*) dans le sens du regret moyen signifie que la récompense moyenne que l'agent obtient en utilisant cet algorithme est au moins aussi grande que ce que peut lui apporter une stratégie pure fixe (le regret $\bar{\mathcal{R}}_i$ est négatif ou nul quel que soit l'adversaire).

Zinkevich [ZIN 03] a proposé une extension de l'algorithme IGA [SIN 94] aux jeux admettant plus de deux actions et deux joueurs, à savoir aux jeux à somme générale sans aucune contrainte. Ledit auteur a prouvé que son algorithme, appelé GIGA (pour *Generalized Infinitesimal Gradient Ascent*), n'a pas de regret dans les jeux en forme normale. L'algorithme GIGA est similaire à IGA et PHC. La mise à jour de la politique de l'agent i se fait comme suit :

$$\pi_i^{t+1} = P(\pi_i^t + \eta \mathbf{r}_i^t).$$

La fonction $P(\tilde{\pi}_i)$ est une fonction qui réduit la politique $\tilde{\pi}_i$ résultante de la sommation $(\pi_i^t + \eta \mathbf{r}_i^t)$ à une distribution de probabilité légale :

$$P(\tilde{\pi}_i) = \underset{\pi_i \in \Delta A_i}{\operatorname{argmin}} ||\tilde{\pi}_i - \pi_i||$$

où l'opérateur $||\cdot||$ est la norme euclidienne usuelle. Zinkevich a montré que le regret total de GIGA est borné par :

$$\mathcal{R}_i^T \leq \frac{\left(\max_{\pi_i, \pi_i'} ||\pi_i - \pi_i'||\right)^2 \sqrt{T}}{2} + \left(\sqrt{T} - \frac{1}{2}\right) \left(\sup_{\pi_i, t=1\ldots T} ||\mathbf{r}_i^t||\right)^2.$$

En supposant que toutes les récompenses $r_i^{a_i}$ du joueur i sont bornées par r_i^{max} et en tenant compte que $\max_{\pi_i, \pi_i'} ||\pi_i - \pi_i'|| = \sqrt{2}$, on peut écrire :

$$\mathcal{R}_i^T \leq \sqrt{T} + \left(\sqrt{T} - \frac{1}{2}\right) |A_i| (r_i^{max})^2.$$

En utilisant la règle de l'Hôpital[9], on peut trouver que :

$$\lim_{T \to \infty} \frac{1}{T} \left(\sqrt{T} + \left(\sqrt{T} - \frac{1}{2}\right) |A_i| (r_i^{max})^2\right) = 0,$$

ce qui nous amène à conclure que $\bar{\mathcal{R}}_i \leq 0$ et, donc, GIGA n'a pas de regret au sens du regret moyen.

Quant à lui, Bowling [BOW 05] a montré que le principe WoLF appliqué à GIGA fait converger ce dernier vers un équilibre en *self-play*. Son WoLF-GIGA hérite alors de la propriété de GIGA d'être sans regret et converge vers un équilibre de Nash en stratégies mixtes dans les jeux à somme générale à deux joueurs–deux actions.

4.3.4.3. *Métastratégies*

Un autre point de vue sur l'apprentissage dans les jeux répétés a été proposée par Powers et Shoham [POW 05b, POW 05a]. Leur méthode garantit des propriétés différentes selon la classe de l'adversaire, ce qui se traduit par trois propriétés :

– *optimalité ciblée :* si l'adversaire appartient à une classe cible, le joueur fournit la meilleure réponse ;

– *compatibilité :* contre lui-même (en *self-play*), le joueur trouve un équilibre de Nash non Pareto-dominé par un autre ;

– *sûreté :* contre tout autre adversaire, le joueur assure son niveau de sécurité dans ce jeu.

---

9. Cette règle est définie comme suit : si f et g sont deux fonctions dérivables en a, s'annulant en a et telles que le quotient $\frac{f'(a)}{g'(a)}$ soit défini, alors $\lim_{x \to a} \frac{f(x)}{g(x)} = \frac{f'(a)}{g'(a)}$.

Les algorithmes proposés (Metastrategy [POW 05b] et Manipulator [POW 05a]) commencent tous deux par une phase d'identification de la classe de l'adversaire, avant de choisir le meilleur algorithme expert à employer contre lui.

Il est instructif d'observer que, dans les jeux répétés, on est passé successivement :

– de stratégies écrites à la main (Tit-for-Tat, etc.),

– à des algorithmes apprenant à s'adapter face à des stratégies « fixes » ;

– puis à des algorithmes capables de s'adapter à des adversaires apprenant ;

– et ici à des méta-algorithmes sélectionnant un algorithme expert selon le type d'adversaire identifié.

La question qui vient naturellement est : que se passe-t-il quand un méta-algorithme en rencontre un autre ? Va-t-il falloir mettre au point des méta-méta-algorithmes ?

4.3.5. *Discussion*

Il est intéressant de voir comment les différents algorithmes introduits précédemment peuvent être comparés entre eux. A cet effet, le tableau récapitulatif 4.1 présente les différentes caractéristiques des algorithmes d'apprentissage ou de planification multi-agents vus dans ce chapitre. Ce tableau est inspiré des travaux d'Aras [DUT 06].

	Connu ou observé				Type	Tous	Propriétés
Algorithme	S	R_i	$\mathbf{A}_{-i}$	$\mathbf{R}_{-i}$	équ.	jeux	théoriques
[SHA 53]	connu	connu	connu	connu	Nash	non	oui
[KEA 00]	connu	connu	connu	connu	Nash	oui	oui
Minimax-Q	connu	connu	observé	observé	Nash	non	oui
Nash-Q	connu	connu	observé	observé	Nash	non	oui
Jeu fictif	connu	connu	observé	non	Nash	non	oui
[GIE 06]	observé	observé	observé	non	Nash	non	non
JALs	observé	observé	observé	non	Nash	non	non
WoLF-PHC	observé	observé	observé	non	Nash	non	oui
GIGA-WoLF	observé	observé	observé	non	Nash	non	oui
Hyper-Q	observé	observé	observé	non	?	?	non
ADL	observé	observé	observé	non	?	?	non
Manipulator	observé	observé	non	non	?	?	oui
MetaStrategy	observé	observé	non	non	?	?	oui

Tableau 4.1. *Comparaison des algorithmes évoqués*

Dans ce tableau, tous les algorithmes introduits dans ce chapitre sont situés relativement à des exigences concernant l'observabilité de certaines propriétés du jeu. La colonne S indique si les états de l'environnement sont supposés connus par les agents ou si les agents n'en ont qu'une perception partielle ; la colonne R_i indique si

les agents connaissent leurs récompenses ou bien s'ils sont capables de les observer en interagissant avec les autres. Les colonnes $\mathbf{A}_{-i}$ et $\mathbf{R}_{-i}$ indiquent si les agents sont supposés capables de connaître par avance ou d'observer les actions pouvant être faites par les autres joueurs. « Non » dans les cellules signifie que les agents ne peuvent pas connaître ni observer certaines propriétés du jeu. Les colonnes « Type équi. » et « Tous jeux » indiquent si l'équilibre de Nash est atteint par chacun des algorithmes et si ce fait est valide pour n'importe quel jeu, ou bien s'il existe certaines restrictions (comme par exemple le fait que le jeu soit à somme nulle, ou que seulement deux actions–deux joueurs soient permises). La colonne « Propriétés théoriques » indique si oui ou non certaines propriétés théoriques des algorithmes sont prouvées.

Il convient de noter que les points d'interrogation dans les cellules du tableau 4.1 signifient que cet aspect n'est pas bien étudié et/ou compris dans la littérature. Comme certains des algorithmes évoqués sont assez récents, il reste beaucoup de travail à faire en vue d'expliquer leur comportement dans différentes situations et d'étudier leurs propriétés de convergence, leur complexité et ainsi de suite.

Une autre classification intéressante des algorithmes de jeu (répétés ou stochastiques) a été proposée par Chang et Kaelbling [CHA 02]. Ces auteurs ont proposé une classification des algorithmes *via* le produit cartésien des stratégies possibles et des croyances possibles sur les stratégies des adversaires. Dans ce contexte, les stratégies possibles d'un agent ont été classées selon la longueur de l'historique gardé en mémoire, de $\mathcal{H}_0$ jusqu'à $\mathcal{H}_\infty$. Evidemment, en ayant plus de mémoire, les agents peuvent élaborer des politiques plus complexes et plus « astucieuses ». Dans le même ordre d'idée, un agent peut classer ses adversaires selon le même principe. S'il croit que son adversaire n'a pas de mémoire, il le classe dans $\mathcal{B}_0$. S'il croit que la mémoire de son adversaire est illimitée alors il le classe dans $\mathcal{B}_\infty$. La classification proposée par Chang et Kaelbling, adaptée pour refléter certains algorithmes évoqués dans ce chapitre, est présentée par le tableau 4.2. Notons que les algorithmes dénotés comme « Bully », « Godfather » et « Poids multiplicatif » n'ont pas été discutés dans ce chapitre. Pour plus de détails, le lecteur peut se référer aux documents suivants [LIT 01, FRE 99].

	$\mathcal{B}_0$	$\mathcal{B}_1$	$\mathcal{B}_\infty$
$\mathcal{H}_0$	Minimax-Q, Nash-Q		Bully
$\mathcal{H}_1$			Godfather
$\mathcal{H}_\infty$	Q-learning, (WoLF)-PHC, Jeu fictif	ADL avec $\|h\| = 1$	Poids multiplicatif

Tableau 4.2. *Classification des algorithmes proposée par Chang et Kaelbling et adaptée de [CHA 02] pour les algorithmes présentés dans ce chapitre*

4.4. Conclusion et perspectives

Nous avons présenté dans ce chapitre un survol de la théorie des jeux et son application à la prise de décision dans les environnements multi-agents. A cet effet, les

jeux en forme normale et les jeux dynamiques ont été tout d'abord discutés, en tant que modèles d'interaction entre agents rationnels. Ces modèles ont ensuite été étendus au monde multi-état en les combinant avec les MDP. Ceci a alors fait émerger un modèle plus puissant, appelé « jeux stochastiques », qui permet de décrire des interactions complexes du monde réel étendu dans le temps.

Plusieurs algorithmes de planification et d'apprentissage utilisant le formalisme des jeux stochastiques ont été présentés et discutés. On a ainsi pu constater que certains d'entre eux possèdent une bonne base théorique tandis que d'autres ne sont encore étudiés qu'expérimentalement. Comme ce domaine de recherche est assez jeune et en période de croissance rapide, on peut s'attendre à ce que, dans un proche avenir, ces algorithmes soient convenablement analysés d'un point de vue théorique et que d'autres approches intéressantes soient proposées.

Hormis les assises théoriques manquantes, il existe d'autres problèmes qui doivent être résolus afin de permettre à ces algorithmes d'être appliqués à des problèmes du monde réel. Tout d'abord, la complexité des algorithmes reste assez élevée. En fait, ce problème est caché dans le modèle même des jeux stochastiques dont le nombre d'états (et d'actions conjointes) croît exponentiellement avec le nombre d'agents. De plus, certains algorithmes exigent de percevoir des actions conjointes de tous les autres joueurs, ce qui n'est pas toujours une hypothèse réaliste. A cela s'ajoute le problème induit par la multitude des équilibres dans un jeu et pour lequel la coordination sur le choix d'un seul équilibre n'est pas évidente pour le moment.

Il convient de noter que ce chapitre n'a pas abordé une autre branche de recherches dans le domaine de la décision dans l'incertain dans les environnements multi-agents que sont les jeux stochastiques partiellement observables (POSG, pour *Partially Observable Stochastic Games*). Dans ce modèle, les jeux en forme normale sont combinés avec le modèle des POMDP, ce qui permet de résoudre des problèmes où les agents ont une vue partielle de l'état de l'environnement. De bons exemples de cette problématique et des approches proposées pour la résolution de celle-ci se trouvent dans [HAN 04, EME 04].

De nombreux travaux se sont concentrés sur les jeux stochastiques « strictement coopératifs » au sens que tous les agents partagent une même fonction de récompense. C'est d'ailleurs le sujet du chapitre **??** du volume 2, lequel présente différentes approches : MMDP, MTDP ou DEC-POMDP par exemple. Ce dernier modèle, celui des POMDP décentralisés, peut être vu comme le modèle POSG avec $R_i = R_j$ pour tout i, j. Des algorithmes spécifiques sont développés qui exploitent cette coopération. En particulier, il est courant – mais pas systématique – que ces algorithmes soient centralisés [BER 02].

Enfin, il faut savoir qu'un outil, dénommé GAMUT, a été développé pour faciliter l'évaluation expérimentale d'algorithmes dans le domaine de la théorie des jeux. L'article [NUD 04] fait une présentation complète de cet outil.

Remerciements : les auteurs tiennent à remercier M. Bowling pour l'aide qu'il a prodiguée relativement à la partie technique du gradient de la politique.

4.5. Bibliographie

[AUE 95] AUER P., CESA-BIANCHI N., FREUND Y., SCHAPIRE R., « Gambling in a Rigged Casino : The Adversarial Multi-Armed Bandit Problem », *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, p. 322–331, 1995.

[AUM 02] AUMANN R. J., HART S., Eds., *Handbook of Game Theory with Economic Applications*, vol. 3, Elsevier Science, Hollande, 2002.

[BER 02] BERNSTEIN D., GIVAN R., IMMERMAN N., ZILBERSTEIN S., « The Complexity of Decentralized Control of Markov Decision Processes », *Mathematics of Operations Research*, vol. 27, n°4, p. 819–840, 2002.

[BOW 02a] BOWLING M., VELOSO M., « Multiagent Learning using a Variable Learning Rate », *Artificial Intelligence*, vol. 136, n°2, p. 215–250, 2002.

[BOW 02b] BOWLING M., VELOSO M., « Scalable Learning in Stochastic Games », *AAAI Workshop on Game Theoretic and Decision Theoretic Agents*, 2002.

[BOW 03a] BOWLING M., Multiagent Learning in the Presence of Agents with Limitations, PhD thesis, University of Toronto, 2003.

[BOW 03b] BOWLING M., VELOSO M., « Simultaneous Adversarial Multirobot Learning », *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, 2003.

[BOW 05] BOWLING M., « Convergence and No-Regret in Multiagent Learning », *Advances in Neural Information Processing Systems 17 (NIPS'04)*, p. 209–216, 2005.

[BRO 51] BROWN G., « Iterative solution of games by fictitious play », *Activity Analysis of Production and Allocation*, vol. 13, p. 374–376, 1951.

[BUR 07] BURKOV A., CHAIB-DRAA B., « Multiagent Learning in Adaptive Dynamic Systems », *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'07)*, 2007.

[CHA 02] CHANG Y., KAELBLING L., « Playing is Believing : The Role of Beliefs in Multi-Agent Learning », *Advances in Neural Information Processing Systems 14 (NIPS'01)*, 2002.

[CLA 98] CLAUS C., BOUTILIER C., « The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems », *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI'98)*, Menlo Park, CA, AAAI Press, p. 746–752, 1998.

[DUT 06] DUTECH A., ARAS R., CHARPILLET F., « Apprentissage par renforcement et théorie des jeux pour la coordination des systèmes multi-agents », *Colloque africain pour la recherche en informatique (CARI'06)*, 2006.

[EME 04] EMERY-MONTEMERLO R., GORDON G., SCHNEIDER J., THRUN S., « Approximate Solutions for Partially Observable Stochastic Games with Common Payoffs », *Proceedings of the 3rd Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'04)*, 2004.

[FIN 64] FINK A., « Equilibrium in a Stochastic n-Person Game », *Journal of Science in Hiroshima University Series*, vol. 28, p. 89–93, 1964.

[FRE 99] FREUND Y., SCHAPIRE R., « Adaptive Game Playing using Multiplicative Weights », *Games and Economic Behavior*, vol. 29, p. 79–103, 1999.

[FUD 91] FUDENBERG D., TIROLE J., *Game Theory*, Mit Press, 1991.

[FUD 99] FUDENBERG D., LEVINE D. K., *The Theory of Learning in Games*, MIT Press, Cambridge, MA, 1999.

[GEN 00] GENTIS H., Ed., *Game Theory Evolving : A Problem-Centered Introduction to Modeling Strategic Interaction*, Princeton University Press, Princeton, N.J., 2000.

[GIE 06] GIES O., CHAIB-DRAA B., « Apprentissage de la coordination multiagent : une méthode basée sur le Q-learning par jeu adaptatif », *Revue d'Intelligence Artificielle*, vol. 20, n°2-3, p. 385–412, 2006.

[GRÄ 02] GRÄDEL E., THOMAS W., WILKE T., Eds., *Automata, Logics and Infinite Games*, Springer-Verlag, vol. 2500 of LNCS, 2002.

[HAN 04] HANSEN E., BERNSTEIN D., ZILBERSTEIN S., « Dynamic Programming for Partially Observable Stochastic Games », *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI'04)*, p. 709–715, 2004.

[HAR 00] HART S., MAS-COLELL A., « A Simple Adaptive Procedure Leading to Correlated Equilibrium », *Econometrica*, vol. 68, n°5, p. 1127–1150, Blackwell Synergy, 2000.

[HAU 98] HAURIE A., KRAWCZYK J. B., *An Introduction to Dynamic Games*, Faculty of Economics and Social Sciences, University of Geneva, Genève, Suisse, 1998, Handouts.

[HOF 66] HOFFMAN A., KARP R., « On Nonterminating Stochastic Games », *Management Science*, vol. 12, n°5, p. 359–370, JSTOR, 1966.

[HU 03] HU J., WELLMAN M., « Nash Q-learning for General-Sum Stochastic Games », *Journal of Machine Learning Research*, vol. 4, p. 1039–1069, MIT Press, 2003.

[KEA 00] KEARNS M., MANSOUR Y., SINGH S., « Fast Planning in Stochastic Games », *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'00)*, 2000.

[KIT 97] KITANO H., ASADA M., KUNIYOSHI Y., NODA I., OSAWA E., « RoboCup: The Robot World Cup Initiative », *Proceedings of the 1st International Conference on Autonomous Agents (Agents'97)*, New York, NY, Etats-Unis, ACM Press, p. 340–347, 1997.

[KOE 96] KOENIG S., SIMMONS R. G., « The Effect of Representation and Knowledge on Goal-Directed Exploration with Reinforcement-Learning Algorithms », *Machine Learning*, vol. 22, p. 227–250, 1996.

[LIT 94] LITTMAN M., « Markov Games as a Framework for Multi-Agent Reinforcement Learning », *Proceedings of the 11th International Conference on Machine Learning (ICML'94)*, 1994.

[LIT 01] LITTMAN M., STONE P., « Leading Best-Response Strategies in Repeated Games », *IJCAI'01 Workshop on Economic Agents, Models, and Mechanisms*, 2001.

[MAR 75] MARTIN D. A., « Borel determinacy », *Annals of Mathematics*, vol. 102, p. 363–371, 1975.

[MYE 97] MYERSON R. B., Ed., *Game Theory : Analysis of Conflict*, Harvard University Press, Boston, MA, 1997.

[NAS 51] NASH J. F., « Non-cooperative games », *Annals of Mathematics*, vol. 54, p. 286–295, 1951.

[NEU 28] VON NEUMANN J., « Zur Theorie der Gesellschaftsspiele », *Mathematische Annalen*, vol. 100, n°1928, p. 295–320, 1928.

[NUD 04] NUDELMAN E., WORTMAN J., SHOHAM Y., LEYTON-BROWN K., « Run the GAMUT : A Comprehensive Approach to Evaluating Game-Theoretic Algorithms », *Proceedings of the 5th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, p. 880–887, 2004.

[OSB 04] OSBORNE M. J., Ed., *An Introduction to Game Theory*, Oxford University Press, Oxford, Royaume-Uni, 2004.

[PAP 95] PAPADIMITRIOU C. H., « Algorithms, Games, and the Internet », *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC'91)*, ACM Press, p. 749–753, 1995.

[PAR 02] PARSONS S., GMYTRASIEWICZ P., WOOLWRIDGE M., Eds., *Game Theory and Decision Theory in Agent-Based Systems*, Springer Verlag, 2002.

[POL 69] POLLATSCHEK M., AVI-ITZHAK B., « Algorithms for Stochastic Games with Geometrical Interpretation », *Management Science*, vol. 15, n°7, p. 399–415, JSTOR, 1969.

[POW 05a] POWERS R., SHOHAM Y., « Learning Against Opponents with Bounded Memory », *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.

[POW 05b] POWERS R., SHOHAM Y., « New Criteria and a New Algorithm for Learning in Multi-Agent Systems », *Advances in Neural Information Processing Systems 17 (NIPS'04)*, Cambridge, MA, MIT Press, 2005.

[RUS 95] RUSSELL S., NORVIG P., *Artificial Intelligence : A Modern Approach*, Prentice Hall Series in Artificial Intelligence, Englewood Cliffs, New Jersey, 1995.

[SHA 53] SHAPLEY L., « Stochastic Games », *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, vol. 39, p. 1095–1100, 1953.

[SHO 04] SHOHAM Y., POWERS R., GRENAGER T., « Multi-Agent Reinforcement Learning : a Critical Survey », *Proceedings of the AAAI Fall Symposium on Artificial Multi-Agent Learning*, 2004.

[SIN 94] SINGH S., KEARNS M., MANSOUR Y., « Nash Convergence of Gradient Dynamics in General-Sum Games », *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'94)*, San Francisco, CA, Morgan Kaufman, p. 541–548, 1994.

[SMI 02] SMITH J. M., Ed., *Evolution and the Theory of Games*, Cambridge University Press, Cambridge, Royaume-Uni, 2002.

[STO 00] STONE P., VELOSO M., « Multiagent Systems : A Survey from a Machine Learning Perspective », *Autonomous Robots*, vol. 8, n°3, p. 345–383, Springer, 2000.

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

[SUT 00] SUTTON R., MCALLESTER D., SINGH S., MANSOUR Y., « Policy Gradient Methods for Reinforcement Learning with Function Approximation », *Advances in Neural Information Processing Systems 12 (NIPS'99)*, Cambridge, MA, MIT Press, p. 1057–1063, 2000.

[TES 04] TESAURO G., « Extending Q-Learning to General Adaptive Multi-Agent Systems », *Advances in Neural Information Processing Systems 16 (NIPS'03)*, Cambridge, MA, MIT Press, 2004.

[THI 04] THISSE J.-F., *Théorie des jeux : une introduction*, Université catholique de Louvain, Département des sciences économiques, 2004.

[UTH 03] UTHER W., VELOSO M., Adversarial reinforcement learning, Rapport n°CMU-CS-03-107, School of Computer Science, Carnegie Mellon University, 2003.

[VRI 87] VRIEZE O., *Stochastic Games with Finite State and Action Spaces*, Centrum voor wiskunde en informatica, Amsterdam, Pays-Bas, 1987.

[YIL 03] YILDIZOGLU M., Ed., *Introduction à la théorie des jeux*, Dunod, Paris, 2003.

[YOU 93] YOUNG H., « The Evolution of Conventions », *Econometrica*, vol. 61, n°1, p. 57–84, 1993.

[YOU 98] YOUNG H., *Individual Strategy and Social Structure : An Evolutionary Theory of Institutions*, Princeton University Press, Princeton, N.J., 1998.

[ZIN 03] ZINKEVICH M., « Online Convex Programming and Generalized Infinitesimal Gradient Ascent », *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*, 2003.

[ZIN 06] ZINKEVICH M., GREENWALD A., LITTMAN M., « Cyclic Equilibria in Markov Games », *Advances in Neural Information Processing Systems 18 (NIPS'05)*, 2006.