THESE de DOCTORAT de l'UNIVERSITE PARIS 6

Spécialité :
Informatique

Présentée par
Christophe Meyer

Pour obtenir le grade de
DOCTEUR de L'UNIVERSITE PARIS 6

Sujet de la thèse :

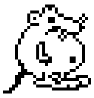
S.A.G.A.C.E.

Solution **A**lgorithmique **G**énétique pour l'**A**nticipation de **C**omportements **E**volutifs

Application aux jeux à information complète et imparfaite

Soutenue le 29 juin 1999
Devant le jury composé de

Jean-Gabriel GANASCIA	Directeur de thèse
Richard K. BELEW	Rapporteur
Jean-Paul DELAHAYE	Rapporteur
Michèle SEBAG	Rapporteur
Jean-François PUGET	Examinateur
Jean-Daniel ZUCKER	Examinateur



Remerciements

Comme tous mes contemporains, j'ai vainement cherché si par hasard un illustre inspiré n'avait pas enfin produit la métathèse dont tout postulant au titre de docteur a un jour rêvé : celle qui traite de la façon de rédiger les remerciements d'une thèse : comment n'oublier personne, comment ménager les susceptibilités, comment donner envie de les lire jusqu'au bout parce qu'on y a éparpillé quelques gags hilarants qu'on souhaiterait ne pas avoir passé deux heures à peaufiner pour rien, etc.

Cette métathèse étant inachevée depuis dix ans (l'auteur s'acharne sans doute sur les remerciements parce qu'ils permettront, sans appel, de se faire, en deux minutes, une idée précise du travail accompli), j'ai fait à mon idée, vous épargnant une poésie de mon cru, ma recette du bonheur et la liste de tous les gens que j'ai pu rencontrer durant les trois années et demie sur lesquelles s'est étalé ce travail.

En un mot comme en 100[*] : **Merci**…

---

[*] En parlant de gag hilarant, j'utilise le logiciel Microsoft Word pour rédiger ce document (cela suffira sans doute, en soi, à faire rire certains) et le mode de complétion automatique insère, à mon insu, la formule « Centre d'encaissement des amendes » à chaque fois que je souhaite simplement taper en toutes lettres le nombre 'Centre d'encaissement des amendes' !.. ce qui justifie la liberté que je prends de l'écrire en chiffres. Un logiciel intelligent devrait anticiper mon désarroi et ma colère et s'adapter afin d'éviter qu'ils n'augmentent si sournoisement et régulièrement. Doter les systèmes artificiels de telles facultés est justement un des propos de cette thèse.

…Bon, bon, je pensais que, dans une thèse consacrée aux jeux, on pouvait s'amuser un peu, surtout que les occasions vont se faire rares dans la suite de ce texte… Toutefois, cela m'ennuierait beaucoup qu'il y ait le moindre doute sur l'étendue de ma reconnaissance et donc,

Je remercie d'abord mon directeur de thèse, Jean-Gabriel Ganascia pour la confiance dont il m'a honoré tout au long de mon travail et pour tout ce qu'il m'a appris.
Pour avoir lu avec passion de nombreux articles de Jean-Paul Delahaye, je savais qu'il ferait un parfait rapporteur pour ma thèse. Je le remercie sincèrement de s'être laissé convaincre alors que, avec un emploi du temps moitié moins rempli que le sien, toute personne raisonnable m'aurait ri au nez. De la même façon, je suis très reconnaissant à Michèle Sebag d'avoir accepté le rôle de rapporteur malgré un emploi du temps également terriblement chargé. Enfin, il se trouve que pendant ma première année de thèse, j'ai passé six mois à l'U.C.S.D. dans l'équipe de Rik Belew auprès duquel j'ai beaucoup appris. Il a poussé l'hospitalité jusqu'à m'héberger pendant tout mon séjour dans ~~sa propre maison à 200 mètres de la plage~~ son propre bureau, me permettant ainsi de travailler tous les soirs jusqu'à des heures indécentes à l'université. Merci à lui et à toute sa merveilleuse équipe (famille et étudiants confondus).
Ma profonde reconnaissance va à Jean-François Puget et à Jean-Daniel Zucker pour avoir accepté de participer à mon jury.
De nombreuses autres personnes m'ont aidé dans ce travail, particulièrement l'équipe de recherche ACASA du LIP6 au sein de laquelle je me suis épanoui pendant toute sa durée.

Je remercie sincèrement Bruno Heintz, président de la société Mathématiques Appliquées S.A. pour le réel soutien qu'il m'a apporté depuis mon intégration au sein de ses équipes. J'associe à ces remerciements l'ensemble du personnel de la société.

Pendant les quatre années de ce travail de recherche, j'ai été formidablement soutenu par tous mes proches. Ma femme, Stéphanie, a contribué massivement à ce travail. M'encourageant par tous les moyens, elle a notamment donné naissance à Agathe qui, dès ses premiers jours, a participé au travail familial en m'évitant pendant des semaines de m'endormir avant l'aube…
Mon père, Jean-Arcady, auquel la clarté et le contenu de cette thèse doivent beaucoup, est sans doute un des plus soulagés de l'achèvement de ce travail dont il est un artisan majeur depuis maintenant trente ans ! Dans cette tâche, auprès de laquelle les travaux d'Hercule font figure de promenade digestive, il a été néanmoins grandement aidé par le reste de la tribu Meyer (Georgina, Véronique, Alexandre, Karine, Sophie, que je vous veux du bien !..). Mes deux grandes familles (Meyer-Touzin et Georget-Colin) m'ont apporté un soutien constant dont je suis parfaitement conscient (particulièrement en ce qui concerne Ben et Agnès).
J'ai également des amis (si, si…) : une troisième famille. Personne parmi eux n'est étranger au résultat final : Jean-Pierre et Candice, Alban, Michèle, Jean-Daniel (encore lui…) et Isabelle, Mourad, Dorothée, Pierre-Yves et Anne-Laure, David et Victoire, Bernard, Christophe, Olivier…

La liste complète est bien trop longue pour tenir ici, mais je n'oublie personne.

RESUME

Dans un monde où l'Homme est amené à interagir de plus en plus avec des machines physiques et/ou logicielles dotées de facultés artificiellement intelligentes, il convient de donner les moyens à ces machines de s'adapter à lui. La machine doit être au service de l'Homme et non l'inverse. A ce titre elle doit être autonome et capable d'anticiper ses besoins, ses désirs et ses comportements pour une simple question d'efficacité. La tâche est compliquée par le fait que l'Homme n'est pas une machine lui-même, il est adaptatif, il change d'avis, il est souvent irrationnel…

Le travail qui est décrit dans cette thèse s'inscrit dans ce cadre. Il concerne le développement d'une méthode modulaire d'anticipation de comportements évolutifs : S.A.G.A.C.E.

Les jeux permettent de confronter les machines à des humains dans un contexte où ils dévoilent la plupart de leurs facultés d'analyse, de réflexion et d'adaptation. La réalisation d'un système capable d'anticiper les stratégies d'un joueur humain devrait permettre, à terme, d'appréhender l'anticipation du comportement humain dans de nombreuses autres activités et répondre ainsi, au moins partiellement, au but précédemment énoncé.

ABSTRACT

In a world where men increasingly interact with hardware and/or software, it is advisable to give machines the means to adapt to their users. The machine should serve Man and not the reverse. In order to maximize efficiency, it has to be autonomous and capable of anticipating the user's needs, desires and behaviors. The task is complicated by the fact that Man is not a machine himself : he is adaptive, changes his mind, often acts irrationally…

The work that is described in this thesis lies within this framework; It concerns the development of a modular method for anticipating changing (adaptive) behaviors : S.A.G.A.C.E.

Games allow machines to confront humans in a context where they must make the most of their abilities of analysis, reflection and adaptation. The implementation of a system capable of anticipating human strategies must allow, in the long run, to manage the anticipation of human behavior in many other activities and thus to fulfil, at least partially, the above mentioned objective.

TABLE DES MATIERES

Une histoire…

Pour cette rencontre mensuelle du club 'Nihil a me alienum puto', le cogniticien avait été chargé de trouver un sujet de réflexion apte à satisfaire la curiosité intellectuelle de l'ensemble des membres. Il arriva presque à l'heure tandis que le mathématicien et le tacticien achevaient une partie d'Alésia et, sans mot dire, inséra une disquette dans l'ordinateur du club, ordinateur que l'informaticien et lui avaient eu tant de mal à faire pénétrer dans ce sanctuaire de la pensée humaine. Automatiquement, le programme s'exécuta et les membres du club purent remarquer que le cogniticien avait visiblement réalisé, sans doute avec l'aide de l'informaticien, un programme capable de jouer à Alésia, ce jeu dont les règles et les subtilités avaient animé tant de repas du club.

Tous s'étaient toujours accordés à dire que le jeu avait un intérêt certain mais aucun n'était d'accord sur la stratégie à adopter. Chacun avait réussi à vaincre chacun des autres au moins une fois et tous recherchaient La meilleure stratégie.

Le mathématicien demanda à terminer sa partie avec le tacticien puis d'avoir l'honneur de jouer la première contre la machine. Comme il était sans doute le meilleur joueur, on accéda à sa requête. Il regarda amicalement le tacticien et déclara : « Cher ami, j'ai gagné ». Le tacticien répondit respectueusement « la partie n'est pas terminée, il vous reste certes dix sept pierres mais j'en ai encore quatorze et il vous reste deux pas à faire ». Le mathématicien d'ironiser « si vous aviez suivi mon exposé la semaine dernière sur les arbres de Kuhn, vous sauriez que, quoi que vous fassiez maintenant, je vais gagner, il me suffit de jouer deux ou trois pierres ». Le tacticien, sans rien comprendre aux explications sibyllines de son ami, le connaissait trop bien pour douter de la véracité de ses propos et se tût.

Quelques applaudissements (très) retenus ponctuèrent la fin de la partie.

Le mathématicien s'installa devant l'ordinateur, rapidement entouré par tous les membres du club. Il regarda le cogniticien et se déclara prêt à relever le défi. Le premier coup du programme fut si stupide (a-t-on idée de jouer vingt pierres dès le début de la partie ?) que le mathématicien parvint difficilement à conserver tout son sérieux. Il pensa néanmoins rassurer le cogniticien après avoir gagné la première partie en disant « Cher ami, ne soyez pas déçu, comme je l'ai déjà dit cent fois dans cette illustre assemblée, un ordinateur n'est qu'une machine à calculer qui ne saurait triompher de l'ingéniosité humaine, même programmée par de grands esprits comme celui de notre ami l'informaticien ». Et d'entamer une seconde partie. « Amusant, il n'a pas fait la même ouverture et a remporté le coup... Monsieur le cogniticien, vous avez assurément mis une touche de fantaisie aléatoire dans votre programme ». Le mathématicien remporta la seconde partie, mais, intrigué, il décida de poursuivre... « Tiens, il vient de jouer onze alors que je joue toujours dix au début... Il s'agit sans doute d'une blague que vous me faites tous, vous devez renseigner le programme sur ce que je vais jouer à l'aide d'une télécommande ou autre chose... ». Le cogniticien avec un léger sourire affirma que ce n'était pas le cas et tous confirmèrent.

Les parties s'enchaînaient, les membres du club y allaient de leurs commentaires les plus avisés.

L'économiste : « économisez vos ressources, conservez des pierres »,

Le tacticien : « tâchez de le surprendre, changez de tactique », Le militaire : « frappez un grand coup, jouez tout ! ».

L'informaticien : « Relancez le programme… ».

Le sociologue : « mettez-vous à sa place, essayez de le comprendre ».

Le poète : « lancez un dé et jouez le montant obtenu ».

Le statisticien : « comptez donc le nombre de fois où il a joué ce nombre ! ».

Le politicien : « trompez-le, jouez beaucoup, puis très peu, puis l'inverse ».

Tous se passionnaient pour cette série de parties. En effet, chaque situation du jeu rappelait à chacun une situation professionnelle ou une forme particulière d'un dilemme se rapportant à son propre domaine d'activité.

Les trois parties suivantes se passèrent rapidement et de commentaires. Le mathématicien entamait la quinzième partie. Il avait remporté exactement la moitié des quatorze précédentes. « Je suis persuadé que votre programme va jouer douze au début, je vais donc jouer treize… » […] « ça alors, il a joué un… J'ai l'affreuse sensation qu'il lit dans mes pensées ».

Après avoir perdu la quinzième partie, le mathématicien décida de s'en remettre au hasard pour jouer… et il perdit. Il gagna la dix-septième en jouant très différemment de ses habitudes et, fièrement, dit « Vous voyez, il suffit d'être imprédictible pour gagner, votre programme doit faire une simple analyse statistique des coups… mais cela ne lui sera plus d'aucun secours maintenant que je le sais ».

Le mathématicien perdit trois parties de suite… ce n'était visiblement pas de la simple analyse statistique.

Lors de la trentième partie, le tacticien se permit une réflexion terrible : « Monsieur le mathématicien, vous êtes exactement dans ma *situation tout à l'heure quand nous jouions tous les deux... ». Le programme joua trois pierres et, en trois coups, gagna la partie... « dites-moi, mon ami, ne seriez-vous pas un peu mathématicien par hasard ? » demanda le mathématicien au cogniticien qui lui répondit sans ironie « Comme vous nous l'avez si brillamment expliqué la semaine dernière a propos des arbres de Kuhn et de la Théorie des jeux, si ce programme était seulement mathématicien, comme le jeu est équitable, il gagnerait en moyenne une partie sur deux. Or, si j'ai bien compté, il a remporté vingt parties et vous dix... ».*

Ce programme allait sans aucun doute faire l'objet de quelques débats et chaque membre du club espérait pouvoir trouver dans les techniques mises en œuvre un outil d'étude de sa propre science ou activité.

Certains éléments de cette histoire sont certainement obscurs, notamment en ce qui concerne le jeu d'Alésia. La rédaction de cette thèse a notamment pour vocation de clarifier ces éléments, de soulever quelques questions et même de leur apporter quelques éléments de réponses.

1 Introduction

1.1 Problématique

Cette thèse repose d’éternelles questions et sur le constat qu’elles n’ont, à ce jour, pas reçues de réponses pleinement satisfaisantes.

Si les machines n’étaient jusqu’aujourd’hui uniquement dévolues aux travaux physiquement pénibles pour des humains, le besoin se fait pressant de mettre à disposition de l’Homme des machines capables de l’assister dans des tâches qui requièrent de l’intelligence (robots autonomes, assistants électroniques, objets technologiques personnalisés, agents d’interface, etc.). L’intelligence d’un individu est communément évaluée en fonction de certaines capacités particulières : l’Apprentissage, l’Adaptation et l’Anticipation. Un système artificiel censé remplir les fonctions définies précédemment devrait disposer des mêmes facultés. Cela semble, en effet, indispensable pour que la coopération homme-machine soit efficace.

Nécessairement, ces machines doivent être capables de s’adapter à l’Homme, notamment en anticipant ses comportements pour ne pas les contrarier à mauvais escient. Par ailleurs, certaines machines seront utiles pour assister un utilisateur humain dans les négociations ou les affrontements avec d’autres humains dans des domaines divers (économiques, militaires, politiques, diplomatiques, etc.).

Pour tout cela, nous avons choisi de développer une méthode générique pour l’anticipation de comportements humains (et plus généralement de comportements évolutifs[1]). Nous avons choisi de limiter, dans le cadre de cette thèse, l’étude de telles anticipations à un domaine vaste et facilement transposable : celui des jeux de réflexion.

La Théorie des Jeux permet d’appréhender efficacement certains types de jeux et offrirait une solution acceptable au problème posé si elle n’avait pas de sérieuses limitations que nous décrirons. De fait, les théories apportent souvent des solutions parfaites dans un contexte singulier et stable, mais s’avèrent inadaptées aux problèmes dynamiques dont les données changent continuellement. Or, le facteur humain rend tout problème dynamique parce que l’Homme s’adapte, se montre parfois irrationnel et, par-là même, non déterministe.

---

[1] Il faudra bien, par exemple, anticiper également certains comportements d’assistants évoluant en fonction de leurs utilisateurs.

1.2 Plan de l'exposé

Les jeux

Ce chapitre expose les raisons pour lesquelles cette thèse traite essentiellement de l'anticipation de comportements évolutifs dans un contexte de jeu.

Il indique les différents types de jeux et les résultats de la Théorie des Jeux pour chacune de ces catégories ainsi que les approches pratiques classiques. Il dévoile également dans quel cadre s'inscrit le travail présenté.

Anticipation

Après quelques définitions, ce chapitre présente une théorie générale de l'anticipation ainsi que certains domaines de recherche concernés par ce sujet. Un certain nombre d'applications publiées sont succinctement décrites ainsi que les raisons pour lesquelles ces approches ne sauraient répondre de façon satisfaisante aux buts fixés par ce travail.

Théorie des Jeux

Ce chapitre nous a semblé indispensable dans la mesure où il expose de façon relativement détaillée la Théorie des Jeux et les raisons qui la rendent relativement inadaptée à la problématique de cette thèse. Il contribue à inscrire notre travail dans un cadre théorique général.

Apprentissage et Anticipation dans les jeux

Sont énumérées dans ce chapitre les différentes méthodes classiques d'apprentissage et particulièrement celles qui ont été utilisées (avec ou sans succès) pour la réalisation de programmes de jeux.

Nous décrivons également les raisons rendant indispensables des capacités d'anticipation pour le type de jeux concernés par ce travail. Quelques systèmes utilisant un modèle des adversaires pour l'anticipation sont décrits.

Les stratégies humaines

Ce chapitre éclaire le lecteur sur les stratégies développées par les joueurs humains qui ont été intégrées à la méthode S.A.G.A.C.E. Il traite de la mémoire, de la gestion du hasard, de l'adaptation, de l'apprentissage, de la rationalité (ou plutôt son absence), de la modélisation.

La méthode S.A.G.A.C.E.

Après avoir proposé une présentation des systèmes de classeurs, puis une description des différents jeux utilisés dans le cadre de cette thèse, ce chapitre décrit la méthode S.A.G.A.C.E. en détails.

S.A.G.A.C.E. est illustrée à l'aide de nombreux exemples de jeux ayant fait l'objet d'implémentations qui sont décrites et commentées dans le chapitre suivant.

Expérimentations

S.A.G.A.C.E. a été implémentée pour cinq jeux différents et ce chapitre décrit les expériences et les résultats correspondants.

Conclusions et perspectives

Le titre de ce chapitre se suffit à lui-même.

Annexe A ; Captures d'écran

A titre informatif, certaines copies d'écran des différentes implémentations de S.A.G.A.C.E. sont décrites dans cette partie.

Annexe B ; Suntzu – exemples

Suntzu est un jeu complexe et cette partie lui est dévolue. Elle présente le formalisme des règles implémentées et un exemple de généralisation.

Annexe C ; Alésia – aspects théoriques

Pour ce jeu, nous avons démontré qu'aucune stratégie pure n'était envisageable. Cette démonstration est présentée à titre informatif. Il nous a paru important, pour ce jeu du moins, de montrer pourquoi une approche telle que S.A.G.A.C.E. est indispensable.

2 Les jeux

> L'homme ne joue que là où, dans la pleine acception du mot, il est homme, et il n'est tout à fait homme que là où il joue.
> Friedrich von Schiller, 1759-1805

2.1 Pourquoi étudier les jeux

La conceptualisation scientifique dans la notion de jeu permet l'étude rigoureuse des situations dans lesquelles plusieurs acteurs sont amenés à effectuer des choix dont l'ensemble déterminera, en partie ou totalement, la situation future. L'estimation ou le calcul de l'intérêt d'un choix particulier pour la recherche d'un comportement optimal, les raisons et les méthodes de la coopération ou de la trahison calculées entre acteurs, l'apprentissage de nouveaux comportements, l'adaptation à celui des autres acteurs, l'étude de la rationalité de l'ensemble des acteurs, l'anticipation des comportements, sont autant de sujets traités par l'étude des jeux.

Pour l'étude de toutes les activités en rapport avec ces sujets, la science des jeux apporte des intuitions, parfois des réponses, voire des théorèmes. L'étude des jeux permet ainsi de résoudre indirectement nombre de problèmes dans des domaines aussi variés que l'informatique, la robotique, l'économie, la recherche opérationnelle, la stratégie militaire ou politique, l'écologie, etc. Enfin, l'étude des jeux concerne directement la notion fondamentale de *rationalité.* Elle s'exprime, dans les jeux, par les stratégies mises en œuvre par les différents joueurs pour optimiser leur gain.

Les jeux étant reconnus d'utilité scientifique, leur étude est ancienne et a beaucoup évolué. Les théories mathématiques sur les jeux de hasard remontent à l'origine de la théorie des probabilités. A cette époque, l'étude des jeux se limitait au cas où chaque événement (ou situation de jeu) avait une probabilité déterminée d'apparaître (c'est le cas des jeux de hasard comme les dés ou la roulette). Cependant, la théorie mathématique des jeux s'est surtout développée au XXème siècle avec les travaux d'Emile Borel, puis avec ceux de Von Neumann et de Morgenstein dans les années quarante, et de Nash, dans les années cinquante. C'est en effet à partir du XXème siècle que l'on a cherché à établir une théorie générale des jeux stratégiques, c'est-à-dire des jeux dans lesquels l'intelligence des différents acteurs (les joueurs) intervient.

2.2 Différents types de jeux

L'étude des jeux requiert différentes méthodes ou outils en fonction de leur nature. Il faut, en effet, distinguer différents types de jeux suivant l'information disponible aux joueurs lorsqu'ils doivent prendre leurs décisions, suivant le caractère isolé ou répété du jeu, suivant la possibilité ou non de coopération entre les joueurs, etc.

2.2.1 Information disponible

2.2.1.1 Jeux à information complète et parfaite

Il s'agit des jeux dans lesquels chaque joueur connaît en permanence l'état global du jeu, c'est-à-dire l'ensemble des choix que peut effectuer chacun des joueurs, les conséquences de ceux-ci (nouvel état du jeu, gains des différents joueurs) et les motivations de chaque joueur (en règle général, la seule motivation est de maximiser le gain). Ainsi, chaque joueur doit pouvoir calculer de manière déterministe l'état futur du jeu en fonction de tous les choix possibles des différents joueurs, et ce de manière récursive, jusqu'à la fin du jeu. En clair, chaque joueur doit avoir les moyens de développer l'arbre de toutes les situations de jeu possibles à partir de la situation courante.

Typiquement, il s'agit de jeux comme les échecs, les dames ou le Go.

2.2.1.1.1 *Les Echecs*

Il est incontestablement le jeu à information complète et parfaite le plus étudié depuis l'avènement de la Théorie des jeux. Bon nombre de réalisations algorithmiques performantes ont été réalisées pour l'étude de ce jeu.

Tous ceux qui s'intéressent de près ou de loin aux jeux ont suivi avec intérêt les deux mémorables parties qui ont opposé le champion du monde d'échecs Kasparov « le tigre de Bakou » à l'ordinateur Deep Blue d'IBM. La première partie s'était achevée par la victoire de l'humain, la seconde par celle de la machine. Cette victoire a suscité un nombre incroyable de polémiques sur l'avenir de l'homme et de la machine et particulièrement sur ce que doit être la définition de l'Intelligence. Si la machine d'IBM n'est qu'un monstre de calcul, il n'en reste pas moins qu'elle a réussi à triompher dans une activité cérébrale qui a souvent été désignée comme une mesure de l'intelligence. Kasparov lui-même déclarait qu'aucune machine, aussi rapide et puissante qu'elle soit, ne parviendrait à le battre avant l'an 2000 dans cette activité réclamant tant d'intelligence.

On évoquera aux sujets des échecs les écrits fameux de Clausewitz, de Engel ou de Shannon. Turing a également proposé un programme d'échec simulé à la main.

2.2.1.1.2 Les Dames

C'est un jeu qui a fait également l'objet de nombreux travaux scientifiques, à l'instar de ceux de Schaeffer [Schaeffer, 1996] qui a réalisé le programme « Chinook ». Ce programme est devenu champion du monde de checkers (dames anglaises).

Un tel succès de l'application conjointe de la Théorie des jeux et de l'apprentissage par renforcement[1] s'explique par le fait que les dames ont une combinatoire (ensemble des coups et des situations possibles d'un jeu) beaucoup plus faible que les échecs. Les meilleurs programmes à ce jeu sont ainsi capables de « voir » jusqu'à un horizon bien plus élevé qu'aux échecs.

2.2.1.1.3 Le Go

C'est le jeu commun dont la programmation reste la plus délicate dans la mesure où sa combinatoire est immense. Les meilleurs programmes actuels sont à peine capables de rivaliser avec un joueur humain débutant.

Parmi les écrits immuables concernant le jeu de Go se trouvent sans aucun doute ceux de Sun Tzu. *L'art de la guerre* écrit par Sun Tzu au IVème siècle avant J.-C. est le plus ancien manuscrit en rapport avec le jeu de Go. Même s'il n'y est jamais fait explicitement mention du jeu, cet écrit le concerne directement (de l'avis des experts compétents).

La notion d'information complète est parfois purement théorique. Par exemple, aucun humain ou ordinateur n'est en mesure de connaître l'état global au jeu de Go en se projetant dans l'avenir à un horizon de plus de quelques coups. Un calcul simple permet d'affirmer qu'il existe plus de positions possibles au Go que d'atomes dans l'univers…

Pendant longtemps, ce type de jeu a été le seul à faire l'objet d'études scientifiques. L'hypothèse d'information complète étant considérée comme fondamentale pour les théoriciens des jeux. Cette hypothèse a depuis les années cinquante été relâchée notamment grâce aux propositions d'Harsanyi [Harsanyi, 1967].

---

[1] Nous reviendrons sur cette technique et sur "Chinook" dans le chapitre consacré à l'apprentissage dans les jeux.

2.2.1.2 Jeux à information incomplète

Il s'agit des jeux dans lesquels certaines informations peuvent manquer aux joueurs au moment de leur prise de décision. L'information manquante peut être de différentes natures : les gains associés aux choix des joueurs, l'introduction de facteurs aléatoires dans les règles du jeu, les choix possibles des autres joueurs, etc.

C'est le cas des jeux de dés (on ne connaît pas les tirages à l'avance), des jeux de cartes (on ne connaît pas nécessairement les mains des autres joueurs), etc.

Typiquement, le bridge, le poker, le black-jack, le backgammon entrent dans cette catégorie.

2.2.1.2.1 Le Bridge

Il est couramment désigné comme le plus noble et le plus intellectuel des jeux de cartes. Les règles du jeu ont évolué pour limiter le rôle du hasard lié à la distribution des cartes (introduction des contrats et du « duplicate »).

2.2.1.2.2 Le Backgammon

Dans la catégorie des jeux stratégiques avec dés, le backgammon est le plus célèbre. Il a fait l'objet de plusieurs programme dont BKG 9.8. Ce programme, réalisé à l'université de Carnegie Mellon par Berliner, a battu à plate couture le champion du monde [Berliner, 1980].

2.2.1.2.3 Le Poker

Il s'agit, à notre avis, d'un des jeux les plus subtils qui soient. Il a été beaucoup étudié parce qu'il permet de juger les humains dans une activité intellectuelle demandant de nombreuses facultés (prise de risques, gestion d'un capital, bluff, modélisation des autres acteurs, etc.). Une partie de ce manuscrit est consacrée à différents systèmes informatiques capables de jouer (plus ou moins bien) au Poker.

2.2.1.3 Jeux à information complète et imparfaite

Entre ces deux extrêmes, se trouve un type de jeu remarquable : les jeux à information complète mais imparfaite. Il s'agit principalement des jeux dans lesquels les prises de décision des joueurs sont simultanées. Si les joueurs connaissent, comme c'est le cas pour les jeux à information complète et parfaite, l'ensemble des choix possibles et leurs conséquences, en revanche, ils ne connaissent pas le comportement immédiat des autres joueurs.

Typiquement, les jeux « Pierre/Ciseaux/Papier », « Pair/Impair » et le « dilemme des prisonniers » font partie de cette catégorie de jeux.

2.2.1.3.1 Pierre / Ciseaux / Papier

Ce jeu à information complète et imparfaite est sans doute le plus célèbre. A ce titre, il a fait l'objet d'une implémentation de la méthode S.A.G.A.C.E. et sera donc présenté ultérieurement.

2.2.1.3.2 Pair / impair

Le jeu le plus simple de cette catégorie. C'est un jeu itéré à deux joueurs (notés J_1 et J_2) qui choisissent pour chaque coup entre « pair » et « impair ». Si les deux joueurs ont choisi la même action, le joueur J_1 marque un point, si les deux joueurs ont choisi deux actions différentes, c'est le joueur J_2 qui marque.

Ce jeu peut sembler de peu d'intérêt. Il a pourtant suscité celui d'un des plus éminents fondateurs de l'Intelligence Artificielle : Claude Shannon. Ce dernier a réalisé une machine particulièrement efficace à ce jeu contre les humains [Shannon, 1953]. Ce jeu a fait l'objet d'une implémentation de la méthode S.A.G.A.C.E. et sera donc présenté précisément ultérieurement.

2.2.1.3.3 Dilemme des prisonniers

Ce jeu a été l'objet de la plupart des études de jeux à information complète et imparfaite. Il s'agit d'un dilemme qui se présente à deux complices d'un larcin arrêtés en même temps sur les lieux. Ces deux complices sont isolés l'un de l'autre et se voient offrir l'alternative suivante : il peuvent avouer ou ne pas avouer le crime. Si les deux avouent, ils effectuent chacun deux années de prison. Si l'un avoue mais pas l'autre, celui qui avoue effectue cinq années de prison alors que l'autre est libéré immédiatement. Enfin, si aucun n'avoue, ils effectuent tous les deux quatre années de prison.

La plupart des études de ce jeu, concernent une version itérée de ce dilemme. On citera notamment les travaux de Rappoport [Rappoport, 1965], de Axelrod [Axelrod, 1984] ou de Delahaye [Delahaye, 1995a], [Delahaye, 1995b], [Delahaye, 1996].

2.2.2 Equité

Certains jeux sont dits équitables. Cela signifie que, potentiellement, tous les joueurs ont des rôles interchangeables ou symétriques. En fait, cela signifie qu'aucun joueur n'est favorisé par rapport aux autres.

Les jeux à information complète et parfaite ne sont, a priori, pas équitables dans la mesure où le fait que les coups soient consécutifs introduit une dissymétrie dans le rôle des joueurs (aux échecs, le joueur qui joue avec les pièces blanches joue en premier). Dans de tels jeux, le premier joueur n'est pas systématiquement avantagé (dans le jeu « puissance 4 », le second joueur est avantagé). Aux échecs, on ne sait pas dire aujourd'hui si un joueur est avantagé ou pas, ni éventuellement lequel. Pour trancher sur cette question, il faudrait être capable de développer l'arbre des situations possibles dans son intégralité.

2.2.3 Somme des jeux

2.2.3.1 Jeux à somme nulle

Se dit des jeux dans lesquels la somme des gains de tous les joueurs pour chaque situation est nulle. En d'autres termes, lorsque tout gain positif pour certains joueurs est compensé par autant de perte pour d'autres joueurs.

2.2.3.2 Jeux à somme non nulle

Il s'agit des jeux dans lesquels la répartition des gains et des pertes peut ne pas être nulle. Par exemple, on a longtemps considéré que les rapports liant deux partenaires économiques étaient « formalisables » à l'aide de jeux à somme nulle. Cela sous-entendait qu'il devait y avoir un « gagnant » et un « perdant ». Depuis, on a introduit la notion de gains économiques partagés pour lesquels les deux acteurs peuvent être gagnants sous certaines conditions.

2.2.4 Itération

2.2.4.1 Jeux non itérés

Ce sont des jeux isolés. Une unique partie est disputée entre les joueurs. Il n'y a pas d'autres conséquences que les gains et les pertes éventuelles des différents joueurs.

2.2.4.2 Jeux itérés

Il s'agit de jeux constitués d'un ensemble (fini ou infini) de parties d'un même jeu. Ainsi, le résultat de chaque partie peut avoir des conséquences sur le comportement des différents joueurs lors des parties suivantes.

Ces jeux constituent la majorité des cas d'étude, dont le plus célèbre est le *dilemme itéré des prisonniers* dont il sera question ultérieurement.

2.3 Résultats de la Théorie des jeux

La Théorie des jeux sera décrite dans un chapitre consacré, les indications suivantes présentent simplement certains de ses résultats et objectifs.

2.3.1 Information complète et parfaite

La Théorie des jeux permet théoriquement[1] de résoudre tous les jeux à information complète et parfaite. Elle garantit à chaque joueur son espérance de gain (nous reviendrons en détail sur ces notions dans le chapitre consacré à la Théorie des jeux) sous forme de l'ensemble déterministe des coups à effectuer en fonction de tous ceux possibles d'un quelconque adversaire.

2.3.2 Information complète et imparfaite

Dans ce cadre, la Théorie des jeux garantit théoriquement la détermination de la stratégie optimale quand elle existe. La forme des stratégies optimales pour ces jeux diffère de celles pour les jeux à information complète et parfaite. En effet, en information complète et imparfaite, une solution optimale se présente sous la forme d'une répartition probabiliste des coups à jouer en fonction des situations de jeux. Une telle stratégie garantit l'espérance de gain.

Par exemple, au jeu de *Pierre/Ciseaux/Papier*, la stratégie optimale consiste à jouer aléatoirement et de façon uniforme *Pierre*, *Ciseaux* ou *Papier*. Adopter cette stratégie garantit l'espérance de gain : la nullité en moyenne (autant de parties gagnées que perdues).

L'étude de ces jeux implique de raisonner en termes de *rationalité,* de la modélisation et d'anticipation des comportements. En effet, pour vaincre un adversaire à un tel jeu, il est nécessaire de se donner les moyens d'anticiper ses choix. Pour cela, il faut être capable de construire un modèle de son comportement en fonction de la rationalité qu'on lui prête.

---

[1] Dans la pratique (programmation), comme pour le Go ou les Echecs, les calculs sont souvent irréalisables tant ils sont nombreux. Plus le nombre théorique de calculs à effectuer est important pour un jeu, moins celui-ci est efficacement programmable. Généralement, la combinatoire des jeux à information complète et parfaite détermine l'horizon des programmes (le nombre de coups envisageables par le programme dans un temps imparti).

Contrairement aux jeux à information complète et parfaite, les stratégies les meilleures contre un adversaire particulier ne sont pas nécessairement celles qui supposent que cet adversaire est parfaitement rationnel ni parfaitement adaptées à d'autres adversaires. Nous reviendrons ultérieurement en détail sur ces affirmations mais elles justifient l'étude de ces jeux et les recherches de méthodes de résolutions correspondantes.

2.3.3 Information incomplète

Harsanyi a introduit le concept d'incertitude « exogène » dans l'étude de ces jeux. Ce concept permet de conserver en grande partie le cadre théorique des jeux à information complète et imparfaite. L'idée sous-jacente à cette proposition consiste à supposer que certains paramètres de ces jeux dans le modèle original (issues, gains, comportements des joueurs) peuvent prendre aléatoirement diverses valeurs possibles dont l'ensemble reste connu des différents joueurs. Un joueur fictif, appelé *Nature*, prend part au jeu. Ses choix concernent la détermination des paramètres précédents. La présence de ce joueur transforme la nature du jeu et complète partiellement l'information qui reste cependant imparfaite. La résolution de tels jeux est compliquée par le fait qu'il n'est pas aisé de prêter des attentions à la *Nature*, de la modéliser.

C'est l'étude de ce type de jeux qui permet celle des dilemmes économiques ou politiques pour lesquels nombre de facteurs sont inconnus des différents acteurs.

2.4 Le travail présenté

Les jeux qui illustreront la méthode S.A.G.A.C.E. sont des jeux à somme nulle, à deux joueurs, à information complète et imparfaite.

S.A.G.A.C.E. est une méthode algorithmique qui apporte des solutions théoriques et pratiques pour l'anticipation de comportements évolutifs. A ce titre, elle s'adresse à des activités intellectuelles dans lesquelles il est important, sinon indispensable, de se donner des moyens de prédire le comportement d'agents ou d'acteurs avec lesquels le système qui utilise cette méthode interagit.

Si la prédiction des coups des adversaires dans les jeux à information complète et parfaite peut avoir une importance déterminante (pour l'accélération de la recherche de la solution optimale déterministe), elle n'est pas au cœur de leur étude.

L'étude des jeux à information incomplète est usuellement rapportée sous diverses hypothèses d'incertitude « exogènes » à celle des jeux à information complète et imparfaite (voire parfaite). Ainsi, quelques aménagements seront nécessaires pour que la méthode présentée puisse s'appliquer à ces jeux.

2.5 Jeux et intelligence artificielle

Les jeux ont souvent permis d'illustrer les avancées dans le domaine de l'Intelligence Artificielle (IA). La plupart des jeux concernent les préoccupations de l'IA : résolution de problème, apprentissage, modélisation, adaptation, anticipation, évolution…

Les progrès de l'informatique depuis les années 70 ont permis la réalisation de programmes joueurs aux performances surprenantes. Ces programmes s'avèrent désormais capables de résoudre la plupart des problématiques complexes des jeux de réflexion par leur puissance de calcul.

L'Intelligence Artificielle est concernée par toutes les activités réclamant une forme quelconque d'intelligence. La réussite des programmes de jeu, même quand elle n'est due qu'à la puissance calcul, présente un réel intérêt scientifique.

Shannon, dès 1949, a réalisé les premiers travaux sur la programmation des échecs. Il a proposé une modélisation judicieuse d'une fonction d'évaluation de l'intérêt des états possibles du jeu. Cette fonction incluait des considérations sur la valeur estimée des pièces, sur la position des pièces et sur leur mobilité potentielle. Les meilleurs programmes actuels utilisent toujours une telle fonction d'évaluation même si sa forme a évolué dans les détails.

Aujourd'hui, la machine triomphe de l'homme aux échecs, aux dames, à Othello, à l'Awélé, au Backgamon… Seul le jeu de Go offre une grande résistance à l'Intelligence artificielle. La raison principale de cette situation semble être que les techniques utilisées pour les autres jeux ne sont pas adaptées à celui-ci (combinatoire explosive, fonction d'évaluation particulièrement difficiles à définir). De nombreuses équipes de recherche se consacrent à des approches originales de la programmation d'un joueur de Go.

Le débat dans les communautés scientifiques ou non, est actuellement toujours passionné concernant la défaite du champion du monde des échecs devant la machine. Que signifie réellement cette défaite ? Que la machine est plus intelligente que l'Homme ? Que l'Homme est simplement moins doué pour le calcul ? Certains inquiets sur le devenir de l'Homme se rassurent sur son éventuelle suprématie intellectuelle en retirant tout simplement le statut de jeu « Intelligent » aux échecs. Il convient de rester lucide et réaliste : sans doute dans un avenir proche, les ordinateurs vaincront les humains dans tous les jeux de réflexion, jeu de Go y compris. l'Intelligence humaine ne se réduit heureusement pas aux facultés calculatoires. Le champion humain aux échecs est capable en une seconde d'évaluer une position quasiment aussi précisément que l'ordinateur. Le joueur humain n'envisage et ne développe peut-être qu'un coup mais, comme l'avait dit ironiquement un grand champion, il s'agit du meilleur coup. L'Intelligence se situe certainement davantage dans ces capacités d'analyse que dans les aptitudes au calcul.

L'intelligence artificielle s'est notamment fixé pour objectif l'étude des processus cognitifs humains ou animaux afin de s'en inspirer pour créer des programmes capables d'exhiber également des comportements intelligents. Dans cette optique, la réalisation de programme dits de « force brutale » (uniquement basés sur les capacités de calcul des super-ordinateurs) ne semble pas être un moyen adéquat pour comprendre le fonctionnement du cerveau humain. Ainsi, la gageure en Intelligence artificielle est bien plus de créer des programmes capables de reproduire les stratégies humaines que de les vaincre. C'est dans cette optique que nous avons développé la méthode S.A.G.A.C.E.

Elle a pour objectif de proposer des méthodes d'anticipation des comportements humains. En particulier, elle devrait permettre de mieux appréhender les processus de décision correspondants.

3 Anticipation

> Prévoir est difficile surtout quand il s'agit du futur.
>
> Niels Bohr, 1883-1962
> (prix Nobel de physiques, 1922)

3.1 Définition

Une anticipation est une prévision suivie d'une action. Par exemple, « Il devrait pleuvoir demain... je prendrai donc un parapluie ». On ne peut étudier l'anticipation sans considérer avec autant d'intérêt ces deux aspects (prévision et action).

3.1.1 L'anticipation stratégique

Pour un agent, anticiper les actions d'autres agents consiste à intégrer dans son processus de décision non seulement les données liées à ces autres agents mais également à intégrer un modèle de leurs propres processus de décision. Ce n'est donc pas tant de prédire une décision que d'en déterminer les raisons. La validité d'une telle anticipation part du postulat que ses objets sont rationnels.

3.1.2 L'anticipation aveugle

Anticiper de façon aveugle pour un agent consiste à n'intégrer dans son processus de décision qu'une observation et une analyse des données liées aux autres agents (leurs actions précédentes, leurs contextes, etc.). L'anticipation se réduit alors à une simple prédiction.

Dans un contexte de jeu, la prédiction consiste à effectuer une étude (statistique par exemple) sur les coups précédents des adversaires afin de déterminer leur(s) prochain(s) coup(s). L'anticipation implique également une telle étude mais consiste surtout à tenter d'élaborer un modèle comportemental et cognitif des adversaires afin d'identifier les processus de décision mis en jeu pour produire leurs coups. Autrement dit, il s'agit d'identifier les stratégies sous-jacentes et la façon dont elles évoluent.

3.2 Domaines de recherche

Il est intéressant de remarquer que des recherches variées sont conduites autour de la prévision ou de la prédiction.

- Mathématiques

- Etudes de séries chronologiques
- Phénomènes linéaires stochastiques
- Modèles économétriques
- Etudes asymptotiques

- Economie

- Prédiction des marchés financiers
- Cotations en bourse
- Calculs d'intérêts

- Théorie des Jeux

- Actions des adversaires, des partenaires
- Evolution des situations (théâtre du jeu, conflits, etc.)
- Besoins logistiques

- Physique / Chimie / Mécanique

- Prédiction des réactions chimiques
- Etudes des mouvements des corps, des particules
- Etudes des amortissements, des phénomènes ondulatoires
- Mécaniques des fluides
- Etudes des systèmes à masses variables

- Cybernétique / Robotique

- Etude des comportements, buts et téléologie
- Contrôle de robots
- Stratégies de routage, gestion de trafics

- Science et Nature

- Découvertes scientifiques
- Prévisions météorologiques
- Prédiction de risques de catastrophes naturelles

- Etude des Systèmes / Logiciels

- Prévision des performances
- Robustesse des systèmes
- Diagnostics de pannes
- Anticipation des besoins futurs

- Philosophie / Epistémologie

- Réalité physique et rôle des théories[1]
- Ethique

- I.A. / Approche ANIMATS

- Modélisation
- Classification (Conceptual clustering)
- Reconnaissance (des formes, des concepts...)
- Agents (autonomes et adaptatifs, d'interface, système multi-agents [hybrides] coopératifs / compétitifs)
- Prévision de l'environnement (Animats)

- Divers

- Musique
- Conduite / Navigation

3.3 Les méthodes de l'Anticipation

L'anticipation et la prédiction requièrent des méthodes très diverses : les Mathématiques, la Simulation, l'Apprentissage, la Logique floue ou des méthodes dites « cognitives ».

Les Mathématiques (ou les Statistiques) permettent exclusivement de calculer comment le phénomène analysé peut se comporter dans un avenir plus ou moins proche en se basant uniquement sur la façon dont il s'est réalisé dans le passé. Pour cela, les méthodes ne manquent pas :

---

[1] Carnap dans « Philosophical Foundations of Physics » parle de prédiction et d'explication comme double rôle de la théorie. 'La valeur suprême d'une nouvelle théorie réside en son pouvoir de prédire de nouvelles lois empiriques'. 'La portée cognitive d'une loi réside en ses potentialités de prédiction'.
Feigl explique, quant à lui, que le but de la recherche scientifique est de découvrir des rapports réguliers et ainsi de rendre les phénomènes observés le plus 'prédictibles' possible.
Enfin, Larre écrit 'Quant au pouvoir de prédiction, également revendiqué par tous (Mach, de Broglie, les logico-empiriques, Einstein (dans *l'évolution des idées en physique*), il semble concerner davantage la confirmation des théories que leur rôle explicatif'.

- Méthodes empiriques d'analyse des séries chronologiques
- Méthodes de prévision par lissage
- Modélisation à composantes déterministes et aléatoires
- Modèles linéaires stochastiques
- Modèle linéaire général
- Modèles économétriques et modèles stochastiques multivariés

Les méthodes les plus connues sont ARIMA; ARMA; SARIMA; ARCH ; Box & Jenkins.

La Simulation permet, connaissant (toutes) les données d'un système, de recréer artificiellement son fonctionnement ou son évolution sur ordinateur. Ainsi, il est possible d'étudier quel sera son comportement réel dans chaque situation que l'on aura simulée. Les performances de la simulation dépendent du choix et de l'implémentation des paramètres du système et de l'environnement de celui-ci. Ainsi, il n'est pas possible de prévoir ou d'anticiper le comportement d'un système dans une situation non prévue en simulation.

L'Apprentissage est une méthode souvent employée pour la prédiction. Il revêt principalement trois formes qui ont chacune leur spécificité[1]. En Apprentissage par cœur, il s'agit de conserver une trace dans une base de connaissances de tous les événements et contextes que l'on souhaite pouvoir prédire. Dans une situation donnée (un contexte), on interroge la base de connaissances afin de repérer le cas appris correspondant et on connaît alors les événements liés. Cela suppose que l'on considère uniquement des systèmes déterministes (c'est-à-dire qu'à un contexte particulier ne peut correspondre qu'une série d'événements) et implique qu'on ne peut 'prédire' que ce que l'on a déjà appris... En Apprentissage par analogie, le principe reste le même, mais il est possible de considérer des cas non-appris en recherchant dans la base de connaissances des cas plus ou moins similaires... Le résultat dépend alors de la qualité de l'analogie. En Apprentissage par renforcement, des techniques de récompenses/punitions sont mises en œuvre pour accroître ou diminuer la 'crédibilité' des règles d'apprentissage. Plus une règle sera utilisée avec succès, plus sa 'valeur' sera augmentée. Ainsi, si une règle de prédiction a prédit dix fois le bon résultat, alors, il est raisonnable de lui faire confiance la onzième fois ce qui se traduit par le fait que sa 'valeur' est élevée.

Les réseaux de neurones, qui sont un formalisme très répandu pour l'implémentation de techniques d'apprentissage représentent la méthode la plus employée pour la prévision dans les recherches actuelles en Intelligence artificielle. (Voir les Applications).

La Logique floue permet l'étude des événements possibles et probables. Elle est utilisée dans plusieurs applications de prédiction (Voir les Applications).

Il existe plusieurs méthodes « cognitives » (nous appelons ainsi des méthodes a priori plus humaines qu'informatiques). L'imitation et le mimétisme peuvent permettre de prédire les événements en tentant de se « mettre à la place » du système. La généralisation de comportements consiste à créer des modèles du comportement du système. La méthode « d'attribution d'un but » revient à déduire ou prévoir le comportement d'un système en ne s'appuyant que sur les buts qu'on lui attribue. Par exemple, si on suppose qu'un humain veut se rendre le plus vite possible à un certain endroit, il est raisonnable de prévoir que son déplacement sera celui dont le parcours est censé demander le moins de temps.

---

[1] Il existe bien d'autres formes d'apprentissage : Apprentissage par induction, par déduction, par l'exemple, par découverte etc. mais elles sont marginales en ce qui concerne l'utilisation de l'apprentissage pour la prédiction.

3.4 Théorie de l'anticipation

Les anticipations dites *anticipations extrapolatives* sont des projections dans le futur de valeurs observées. Typiquement, la valeur d'une variable X à l'instant t peut être projetée en fonction des valeurs de X aux instants (t-1) et (t-2). Par convention, on écrit X* une anticipation de la valeur de X. On écrit alors :

$$X^*(t) = X(t-1) + \alpha(X(t-1) - X(t-2))$$

Le coefficient α est ce que l'on appelle une mesure du contenu extrapolatif des prévisions. Le cas particulier α=0 définit l'anticipation la plus basique : l'anticipation statistique qui considère que la variation est nulle.

Dans le cas général, la formulation de X* est la suivante :

$$X^*(t) = \sum_{i=1}^{n} \alpha_i X(t-i)$$

Avec, $0<\alpha_i<1$ et $\sum_i \alpha_i = 1$.

Le nombre n est une mesure de la mémoire du système anticipant. Les coefficients α_i expriment la distribution des retards échelonnés. Pour les économistes par exemple, il est communément admis que la mémoire d'un système anticipant est altérée par l'éloignement dans le temps et les α_i décroissent à mesure que i augmente.

Les *anticipations adaptatives* sont différentes. Elles supposent que le système anticipant tient compte des erreurs de ses anticipations passées par apprentissage. Les anticipations sont ainsi révisées en fonction de l'écart entre les données prévues et les données réelles. On écrit alors :

$$X^*(t) = X^*(t-1) + \beta(X(t-1) - X^*(t-1)) \text{ avec } 0 \leq \beta \leq 1.$$

Le coefficient β est une mesure de l'intensité de l'apprentissage par les erreurs. Lorsque β=0, il n'est tenu aucun compte des erreurs de prédiction. Cela signifie que la fonction d'anticipation est invariante. Le cas β=1 correspond aux anticipations statiques.

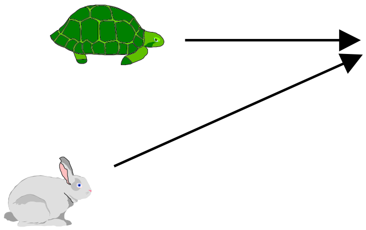
Enfin, les *anticipations régressives* sont à l'opposé des *anticipations extrapolatives*. Au lieu de projeter les données en les prolongeant, elles les projettent en les corrigeant par rapport à un niveau supposé normal.

Elles sont stabilisantes dans la mesure où leur principe est que les données sont plus ou moins stables et qu'une augmentation doit obligatoirement être suivie d'une diminution pour permettre un retour au niveau considéré comme normal.

Remarque : On retrouve évidemment dans ces formules d'anticipation des expressions rencontrées dans l'apprentissage par renforcement.

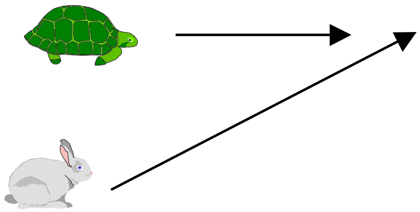
On peut illustrer ces trois formes d'anticipation à l'aide d'un jeu extrêmement simple : la poursuite d'un lapin tentant d'attraper une tortue aveugle (sa trajectoire n'est pas influencée par sa celle du lapin).

Anticipation extrapolative :



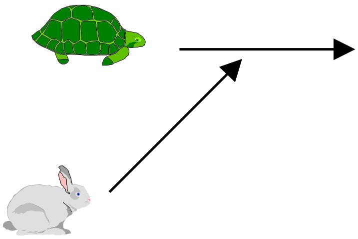
Le lapin se dirige vers une position estimée de la tortue au moment de la rencontre. Il extrapole cette position en fonction de la position et de la vitesse actuelles de la tortue et des siennes.

Anticipation adaptative :



Supposons que le lapin essaie pour la troisième fois d'attraper la tortue par *anticipation extrapolative* et qu'il arrive toujours trop tard (à gauche par rapport à la position réelle de celle-ci). L'anticipation adaptative va consister à ajuster sa course en fonction de ses précédentes erreurs d'estimation et il va, d'office, se diriger plus à droite que la position mal anticipée de la tortue.

Anticipation régressive :



Supposons que la tortue démarre très rapidement mais que le lapin ait une idée précise des facultés de cette dernière. Il peut, à juste titre, estimer qu'une vitesse rapide telle que celle qu'il observe actuellement n'est pas normale et que la tortue va s'essouffler. Il anticipe donc que la tortue va ralentir pour retrouver une vitesse plus lente et se dirige à gauche de la position qui serait estimée en fonction de la position et de la grande vitesse actuelles.

3.5 Applications publiées

Le sujet des recherches diffère souvent de la méthode utilisée pour les conduire. Pour distinguer le sujet de la recherche de la méthode, la référence est suivie d'une indication sur la méthode utilisée.

- _NN : Utilisation de réseaux de neurones,
- _MA : Utilisation des mathématiques ou des statistiques,
- _BC : Méthodes utilisant des bases de connaissances d'expertise,
- _GE : Indication du fait que le travail est général et conceptuel,
- _FL : Utilisation des techniques de la logique floue,
- _SI : Utilisation de la simulation,
- _HR : Utilisation de composants 'hardware'.
- _EX : Une série d'expérience est à la base des travaux considérés.
- _PE : Programmation par exemples.

Les listes suivantes ne sont pas exhaustives mais la bibliographie les complète.

3.5.1 En Mathématiques :

Des études sont dirigées par Syrmos, Misra et Aripirala concernant les phénomènes d'anticipation mathématiques en utilisant les solutions de l'équation discrète généralisée de Lyapunov [Syrmos, 1995]_MA.

Jin-Jen et Yogle présentent une comparaison entre OSP (One-sided Linear prediction) et TSP (two-sided linear prediciton) du point de vue de l'erreur de prédiction. [Jin-jen 1995]_MA.

Eisentein et Kanter étudient mathématiquement un genre de 'perceptron générateur de bits' à l'aide des séries temporelles et des réseaux de neurones. [Eisentein, 1995]_MA_NN.

Enfin, Prochazka et Sys utilisent un réseau de neurones, entraîné génétiquement, pour la prédiction de séries temporelles [Prochazka, 1994]_NN.

3.5.2 En Economie :

Chen utilise les réseaux de neurones pour la prédiction des marchés financiers [Chen 1994]_NN.

Utilisant des données concernant des entreprises qui 'chutent' ou qui progressent, Fanning et Cogger comparent une méthode basée sur les réseaux de neurones avec une méthode, plus ancienne, basée sur les modèles, pour la prédiction financière [Fanning, 1994]_NN.

Castillo et Melin, quant à eux, se basent sur des bases de données d'expertises pour analyser et découvrir des modèles mathématiques pour la prédiction de séries temporelles financières [Castillo, 1994]_MA_BC.

3.5.3 En systèmes / logiciels (performances et optimisations) :

Kahn et Abrams expliquent que la perfection est improbable dans la plupart des systèmes et qu'il convient donc, dès la conception, d'anticiper sur les mauvais fonctionnements et de préparer les mesures de réparation [Kahn 94]_GE.

C'est un contrôleur basé sur la logique floue qui permet à Galichet, Foulloy et Chebre d'anticiper les différentes perturbations d'un système de distillation [Galichet, 1994]_FL.

Mills étudie COMNET III, un système qui peut prédire les performances des communications des réseaux grâce à l'analyse de simulations [Mills, 1994]_SI.

Les mathématiques sont l'outil de Triantafyllos et Vassiliadis pour prédire les erreurs d'implémentation des systèmes informatiques [Triantafyllos, 1995]_MA.

Mahlke et Hank proposent de caractériser l'impact d'une exécution prédite pour la prédiction de 'branches' afin d'améliorer les performances des systèmes parallèles [Mahlke, 1994]_GE.

Deux méthodes sont comparées par Noviello, Serio et Plaitano pour la prédiction des performances des batteries. L'une est basée sur l'étude des indicateurs internes de ces batteries, l'autre sur l'utilisation de bases de connaissances provenant d'expertises. [Noviello, 1993]_MA_BC.

Krishnan et Vitter ont imaginé un système d'algorithme aléatoire qui converge vers un taux de fautes optimal dans le pire des cas [Krishnan, 1994]_MA_GE.

3.5.4 En Robotique / Cybernetique :

Tadokaro, Ishikawa et Takebe ont développé un système de prédictions stochastiques de mouvements humains pour le contrôle de robots travaillant avec des humains. Ce système permet de prévoir les futures positions des humains afin de prévenir tout mouvement dangereux des robots [Tadokaro, 1994]_MA.

Jung et Park, eux, utilisent des réseaux de neurones pour la prédiction des postures humaines pour une conception ergonomique de modèles d'humains [Jung, 1994]_NN.

Russ et Farber étudient des systèmes 'hardware' d'imagerie performants pour la prédiction en temps réel de la structure d'environnements complexes pour des robots mobiles [Russ, 1993]_RH.

Une étude de Kandel, Boe et Orlaguet montre comment on peut, en observant les 'coarticulations anticipatives' des mouvements de la main, prédire ce qui va être écrit [Kandel, 1993]_EX_GE.

Bakker et Kuniyoshi utilisent l'imitation pour l'apprentissage de robots au lieu des méthodes plus conventionnelles telles que la programmation figée ou l'apprentissage par renforcement [Bakker, 1996].

Demiris et Hayes travaillent également sur l'apprentissage des robots par observation et imitation [Demiris, 1994] et [Demiris, 1996].

3.5.5 En Sciences Naturelles :

Des expériences menées par Drevets, Burton et Videen tentent de comprendre les changements de flux sanguin dans le cortex humain durant des stimulations anticipées [Drevets, 1995]_EX.

Satoh et Funnatzu étudient SOPHIA, un système à base de règles dont le rôle est de prédire les réactions chimiques [Satoh, 1995]_BC.

Hoffman travaille sur les prédictions météorologiques et explique en quoi les systèmes de calcul doivent être plus ou moins performants suivant le type de prévisions souhaité [Hoffman, 1994]_MA_HR.

3.5.6 Dans le domaine des agents d'interface

Cypher a développé un système sous HyperCard qui utilise une nouvelle interface, appelée 'anticipation' pour montrer quelles sont les généralisations qu'il a effectuées en observant son utilisateur. Le système apprend donc à connaître son utilisateur et, en surlignant les actions futures probables de celui-ci, il lui montre ses progrès jusqu'au moment où, la confiance s'étant installée, l'utilisateur lui permettra d'effectuer directement certaines tâches. [Cypher, 1991]_PE.

Dent et Boticarjo étudient CAP, un système de prédiction de valeurs par défaut, orienté 'ligne de commande' [Dent, 1992]_BC.

Maes et Kozierok, quant à eux, considèrent plusieurs méthodes pour programmer de bons agents d'interface : l'observation puis l'imitation de l'utilisateur, l'utilisation d'un feedback de celui-ci et la possibilité offerte à l'utilisateur d'entraîner ses agents. [Maes, 1993]_GE.

Schlimmer et Hermens ont conçu un système de prise de notes pour les ordinateurs sans clavier. Ce système, modal et 'corrigeable', apprend en temps réel. [Schlimmer, 1993].

3.5.7 Les Animats :

Meyer explique les bases de l'approche Animat, notamment les problèmes et solutions liés à des environnements imprévisibles, non-donnés [MeyerJA, 1995]_GE.

L'apprentissage de modèles du monde est une technique étudiée par Sutton & Pinette [Sutton, 1985]_NN, Holland [Holland, 1986] et Riolo [Riolo, 1991] qui intègre aux modèles d'action une prédiction des conséquences de l'action.

Rosenschein [Rosenschein, 1985] et Chapman [Chapman, 1991] ont étudié les systèmes dynamiques couplés. Ceux-ci s'adaptent aux changements de l'environnement en modifiant leurs représentations internes et, donc, leur architecture de contrôle.

Fondberg étudie le modèle de Klopf, qui met en œuvre une méthode logicielle chargée d'anticiper ou de prédire le renforcement (apprentissage par renforcement), et censés représenter le fonctionnement du système limbique et de l'hypothalamus [Fondberg, 1986]_GE.

3.5.8 Divers :

Mozer étudie la composition musicale par prédiction grâce aux réseaux de neurones. Dans son système, chaque note est prédite de façon probabiliste en fonction d'une table de transitions et du contexte précédent [Mozer, 1994]_NN_MA.

3.6 Analyse et critique de ces travaux :

Ces travaux permettent de se faire une idée précise des méthodes utilisées pour prévoir des phénomènes variés. Mais, l'Anticipation n'est réellement abordée que dans certains domaines (Robotique, Agents d'interface, Animats), les autres sujets étant orientés uniquement vers l'aspect prédiction. L'anticipation ne se résume pas à la prédiction, la partie action est tout aussi importante.

D'autre part, nous sommes persuadé que pour être efficace durablement, une méthode d'anticipation doit être adaptative. Elle doit être en mesure d'évaluer ses propres performances et doit avoir les moyens de se corriger elle-même et ce, en temps réel.

En effet, comme cela sera explicité par la suite, l'Anticipation n'est pas une simple observation ou un simple calcul, dans la mesure où, souvent, elle influence directement, par sa composante 'action', l'objet de son étude. Par exemple, un chat poursuivant une souris modifie sans arrêt sa trajectoire parce qu'il anticipe les mouvements de la souris, c'est-à-dire une position future présumée de celle-ci. Sa partie action (son déplacement) modifie en permanence l'objet de son anticipation (les mouvements de la souris qui tente naturellement de le fuir).

Cette particularité de l'Anticipation n'est abordée dans aucun de ces travaux alors qu'elle nous paraît essentielle parce qu'elle justifie le fait que l'anticipation doive être auto-adaptative en temps réel.

C'est lorsqu'elle tient compte de cette particularité que l'Anticipation devient 'réfléchie'.

Il est tout de même possible de concevoir des systèmes anticipants performants qui ne prennent pas consciemment en compte l'influence de l'anticipation. Un tel système peut fonctionner en faisant de l'anticipation 'irréfléchie', non 'consciente'. En fait, grâce aux méthodes informatiques d'apprentissage, il est possible d'apprendre inconsciemment les conséquences de l'anticipation. Par exemple, dans l'exemple du chat poursuivant la souris, il suffit d'apprendre les mouvements de la souris, d'une part, et comment l'attraper, d'autre part, sans chercher à lier ces deux apprentissages. Ainsi, si le chat a une technique stable (c'est-à-dire sans trop de variations) de poursuite, les conséquences de ses poursuites seront toujours sensiblement les mêmes sur la souris et il n'est pas nécessaire de savoir que la souris est influencée par les mouvements du chat. En fait, c'est ce qui se passe dans les applications précédemment citées.

Pourtant, il est évident que de telles méthodes ne sont pas robustes. Il suffirait qu'une perturbation de l'environnement modifie le comportement de la souris (par exemple, un petit obstacle) pour que celui du chat soit modifié (car il poursuit toujours la souris) et de ce fait, sa propre influence sur la souris serait modifiée. Cependant, rien dans le système n'est capable de prendre un tel effet rapidement en compte. Il est possible qu'après un certain temps, le système s'adapte ; mais le temps d'adaptation est souvent très coûteux d'un point de vue des performances. Il l'est d'autant plus si les perturbations de l'environnement deviennent fréquentes et chaotiques...

Par ailleurs, il peut être particulièrement intéressant de prendre en compte l'influence de l'anticipation pour l'exploiter, ce que ne permettent pas les systèmes anticipatifs irréfléchis. Ainsi, considérant à nouveau l'exemple du chat et de la souris, le chat peut très bien exploiter l'influence de son anticipation. Cela peut lui permettre d'influencer, dans une direction particulière (rendant la capture plus aisée), le mouvement de la souris. En fait, ce genre d'utilisation est semblable aux travaux sur les modèles d'action, c'est-à-dire aux prédictions des conséquences de l'action (voir les travaux sur les Animats).

L'anticipation n'a pas toujours d'influence sur l'objet de son étude. Par exemple, si un chat tente d'attraper une souris mécanique - dont les mouvements sont déterministes - il est évident que son anticipation ne modifie pas le mouvement de la souris. En fait, cette forme d'anticipation est réduite au seul phénomène de prédiction du comportement d'un système indépendant et présente moins d'intérêt qu'une anticipation active et réfléchie (Voir Problématique de l'Anticipation - modèle passif).

La méthode S.A.G.A.C.E. propose une alternative aux méthodes d'anticipation non conscientes. Elle constitue une méthode d'anticipation réfléchie.

3.7 Problématique de l'Anticipation

L'étude de l'anticipation peut être conduite suivant trois directions : coopérative, passive ou compétitive. Dans le modèle coopératif, il s'agit d'anticiper le comportement de ses partenaires pour rendre l'association plus performante. Dans le modèle passif, on suppose que l'anticipé n'aide ni ne tente de gêner l'anticipant : c'est le cas de la plupart des réalisations. Enfin, le modèle compétitif suppose que le modélisé fait tout son possible pour demeurer le moins prévisible possible. Ce dernier modèle concerne particulièrement la théorie des jeux, ou bien la vision pessimiste selon laquelle la nature (l'environnement) est toujours plus ou moins hostile.

3.8 Influence de l'anticipant sur l'anticipé

La plus grande difficulté de l'anticipation est la gestion de sa propre influence sur autrui. Une méthode d'anticipation performante et robuste doit tenir compte de ce fait dans sa structure même et doit être adaptative.

Les exemples sont nombreux où l'on peut remarquer cette influence.

Les phénomènes de poursuite sont de bons exemples. Un poursuivant a toujours, a priori, une influence sur le poursuivi. Quand le poursuivant anticipe sur le déplacement de l'adversaire, il modifie son propre déplacement qui, à son tour, modifie celui du poursuivi...

On peut se demander si les anticipations que l'on formule vis-à-vis des progrès techniques n'ont pas une influence sur les directions de recherche. Ainsi, le fait que l'homme ait toujours pensé aller sur la lune n'a t-il pas contribué à ce que cela fut, un jour, rendu possible ? Le fait que Moore ait développé une théorie sur l'évolution du nombre de transistors intégrés dans les puces n'a t-il pas eu quelque influence sur cette évolution ?

L'anticipation concernant les besoins futurs en hommes ou en matériel dans l'industrie a toujours une influence sur les besoins eux-mêmes. Il n'est pas rare de constater qu'une anticipation sur les besoins en matériel suggère une augmentation des matériels et que cette augmentation induise une augmentation des besoins, etc.

Dans un article de l'association nationale américaine de la prévention des crimes, on peut lire que le fait d'avoir anticipé une augmentation de la violence dans certains Etats, a conduit à augmenter le nombre d'agents de sécurité mais que cette présence policière a eu pour effet pervers de rendre les citoyens plus craintifs et d'augmenter l'impression d'insécurité.

Les agents d'interface qui anticipent les besoins des utilisateurs peuvent modifier les choix par défaut dans les boites de dialogues, lancer d'avance certains programmes etc. Certains agents d'information sont développés pour parcourir les immenses réseaux câblés d'information pour y puiser celles qui sont susceptibles d'intéresser l'utilisateur. Il semble que le comportement 'anticipatif' de ces agents puisse influencer celui de l'utilisateur (en leur suggérant des actions qui peuvent être acceptées sans réfléchir - il n'est pas évident que cela soit toujours judicieux).

L'entreprise Intel a lancé dernièrement la commercialisation du processeur Pentium III. Cette puce a été développée, comme les précédents Pentium, de telle façon qu'elle est censée prédire l'avenir, autrement dit deviner quelles seront les instructions en attente de traitement. En fait, les ingénieurs d'Intel ont analysé des millions de lignes de programmes pour déterminer les suites de micro-opérations les plus fréquentes. A partir de cela, ils ont conçu l'architecture de leurs puces. Ces anticipations ne sont pas adaptatives et cela pose un réel problème d'influence de l'anticipant. En effet, les programmeurs, d'une façon ou d'une autre, vont être amenés à produire des programmes 'compatibles' avec cette fonctionnalité des puces, c'est-à-dire dont les instructions devront rester classiques pour demeurer prédictibles.

L'anticipation ayant une influence directe et immédiate sur les 'objets' de son étude (dans un modèle coopératif ou compétitif), il est évident qu'elle doit être adaptative en temps réel, c'est-à-dire qu'elle doit pouvoir, en permanence, se modifier en fonction de ses propres performances et des modifications qu'elle entraîne sur l'environnement.

Ces réflexions conduisent naturellement à la question de la profondeur de l'anticipation. En fait, il n'est jamais évident de savoir à quel degré arrêter la réflexion anticipative. Doit-on anticiper sur le fait que l'anticipation va produire un effet sur son objet ? Si oui, cette seconde anticipation aura manifestement un effet sur la première, effet qu'il convient de gérer, etc.

La méthode S.A.G.A.C.E. concerne l'anticipation des comportements évolutifs, particulièrement des comportements humains. A ce titre, elle intègre, comme nous le verrons dans la suite de ce manuscrit, des moyens de prédire des comportements qu'elle influence elle-même. C'est pour étudier cette influence et ses conséquences que S.A.G.A.C.E. a été appliquée aux jeux à information complète et imparfaite dans lesquels de telles capacités sont indispensables.

L'étude systématique des jeux a longtemps été conduite dans une seule direction : l'étude théorique. Cependant, si la Théorie des Jeux permet d'appréhender parfaitement certains jeux, elle est insatisfaisantes pour d'autres, particulièrement lorsqu'il s'agit d'affronter des humains.

4 Théorie des jeux

Le jeu n'a pas d'autre sens que lui-même.
Roger Caillois, 1913-1978

4.1 Présentation

Malgré son nom, la théorie des jeux, branche de la recherche opérationnelle qui étudie la prise de décision en situation de concurrence, n'est pas uniquement consacrée aux activités ludiques comme le jeu d'échec ou les tournois de bridge. Toutes les activités dites stratégiques, parce qu'elles portent sur la coordination et l'agencement des forces face à l'adversité, sont concernées, qu'il s'agisse de diplomatie, de politique, de choix industriels, ou de la conduite des affaires en matière économique et militaire.

Dans le passé, les théories mathématiques sur les jeux remontent à l'origine de la théorie des probabilités, pour des jeux de pur hasard, c'est-à-dire sans autre concurrent que le sort. La théorie mathématique des jeux, quant à elle, s'est développée au XXème siècle avec les travaux d'Emile Borel, puis avec ceux de Von Neumann et de Morgenstein [Von Neumann, 1944] dans les années quarante, et de Nash [Nash, 1950, 1951], dans les années cinquante. Ces derniers sont à l'origine des notions de mini/max et de recherche de point d'équilibre qui sont au cœur de ce que l'on a coutume d'appeler aujourd'hui la Théorie des Jeux.

Dès l'apparition des premiers ordinateurs, la théorie des jeux est apparue comme un formalisme approprié pour la conception de machines à jouer dans le cas de jeux à information complète et parfaite. Dans ce cadre, l'ordinateur a pour tâche de simuler les avenirs prévisibles, en supposant que les adversaires jouent eux-mêmes de façon totalement rationnelle, et tentent d'optimiser leurs avantages. Une telle simulation suppose le déploiement de l'ensemble des possibles, ou presque, sur un arbre de recherche. Lorsque cette simulation est rendue impossible du fait de la taille de cet arbre, des heuristiques sont introduites pour biaiser la combinatoire. La machine doit alors supposer que les concurrents adoptent des heuristiques similaires aux siennes. Ainsi, la plupart des machines qui jouent aux échecs reposent sur l'emploi de l'algorithme du mini/max et d'une fonction d'évaluation commune à tous et qui tient compte de l'état de l'échiquier après quelques demi-coups d'une exploration quasi exhaustive. [Ganascia,1998] & [Meyer, 1997].

Suivant le type des jeux, la Théorie apporte des solutions, des théorèmes ou bien des modèles de la solution 'la moins mauvaise'.

4.2 Les règles du jeu

Un jeu est une situation dans laquelle sont confrontés des individus (appelés joueur) qui doivent effectuer des choix dans un cadre strict appelé *règles du jeu*. Ces dernières spécifient l'ensemble des choix possibles des joueurs en fonction du contexte général (appelé l'état du jeu) et les conséquences de ces choix en fonction les uns des autres. Les conséquences définissent les états futurs du jeu et notamment l'état final, appelé issue du jeu. L'issue du jeu indique les gains ou pertes des différents joueurs en fonction du déroulement de l'ensemble du jeu.

4.3 Jeux sous forme extensive

Il s'agit d'une représentation des règles d'un jeu sous forme d'un arbre. Si on suppose que le jeu concerne n joueurs alors, la mise sous forme extensive d'un jeu inclut :

1. Un arbre topologique Γ et sa racine A (parfois appelée sommet).
2. Une fonction de gain associant un vecteur de dimension n à chaque nœud terminal de Γ.
3. Une partition des nœuds non terminaux de Γ en n+1 ensembles $S_0, S_1, \ldots, S_n$.
4. Un ensemble de répartitions probabilistes associées à chaque nœud de S_0 dont chaque élément est associé à un nœud suivant de S_0.
5. Pour chaque i=1 à n, une sous-partition de S_i en sous-ensembles $S_i^{\,j}$, appelés ensemble d'information, tels que deux nœuds dans le même ensemble d'information aient exactement le même nombre de nœuds suivants immédiats et qu'aucun nœud ne puisse en suivre un autre au sein de chaque $S_i^{\,j}$ (les nœuds d'un sous-ensembles sont au 'même niveau').
6. Pour chaque $S_i^{\,j}$,un ensemble d'index $I_i^{\,j}$ liant les nœuds de $S_i^{\,j}$ à leurs nœuds suivants.

La racine de l'arbre correspond au coup initial, soit le premier coup du premier joueur. Les nœuds terminaux, également appelés feuilles de l'arbre, correspondent chacun à une issue possible du jeu en fonction des actions des différents joueurs. Le vecteur associé à chaque feuille par la fonction de gain définit par sa $i^{ème}$ coordonnée le gain du $i^{ème}$ joueur en fonction de l'issue du jeu.

Les n+1 ensembles S_0, S_1, …, S_n sont appelés les ensembles des joueurs et correspondent aux différents choix possibles des joueurs.

L'ensemble S_0 correspond à un joueur fictif que les théoriciens aiment à appeler *Nature*. Ce joueur est censé jouer (choisir son action) de façon aléatoire suivant, toutefois, une répartition connue des autres joueurs. C'est la présence éventuelle de ce joueur qui permet de définir un cadre commun entre un jeu à information complète ou un jeu à information incomplète.

4.3.1 Exemple

Il existe un jeu dit de pure stratégie (GOPS) décrit par Guillermo Owen [Owen, 1985] qui se prête très bien à l'illustration d'un jeu sous forme extensive. Le jeu original se joue à deux joueurs. Chacun reçoit une série complète de treize cartes (de l'as au roi). Une troisième série est mélangée, cartes face cachée, et disposée entre les joueurs. Les cartes de cette série sont ensuite découvertes l'une après l'autre et les joueurs misent une de leurs cartes (le choix de leur mise leur incombe). Le joueur qui a misé la carte de valeur la plus haute remporte la carte découverte de la troisième série.

Le jeu se termine après 13 coups et le score est donné par la différence entre ce que les deux joueurs ont gagné (somme des valeurs des cartes remportées).

Pour simplifier l'arbre, nous considérons le jeu limité à des séries de trois cartes numérotées 1,2 et 3.

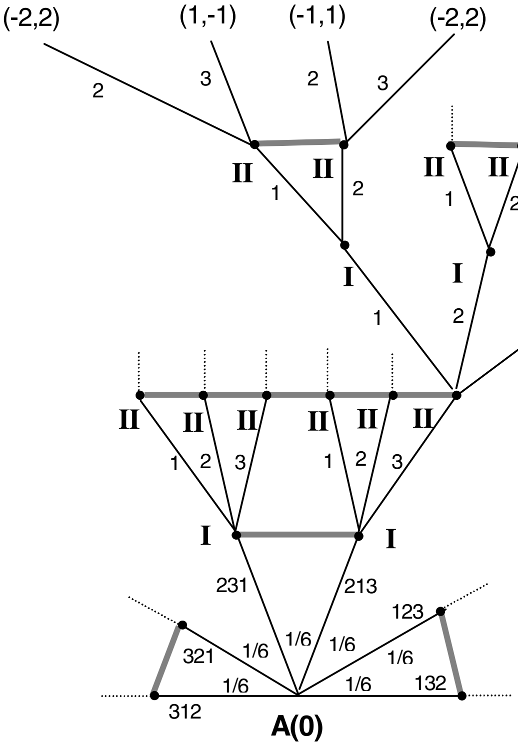


Figure 4.1. Jeu sous forme extensive.

Le sommet de l'arbre A définit le point de départ du jeu. Les différentes configurations possibles de la troisième série peuvent apparaître avec une probabilité de 1/6 chacune. Le joueur *Nature* effectue donc 'son choix de répartition de ses trois cartes' aléatoirement.

En supposant que la première carte retournée soit le 2, il existe encore deux possibilités pour les deux autres cartes (soit l'ordre est 2,3,1 soit 2,1,3). Quel que soit cet ordre, le fait de retourner la première carte offre exactement la même information. Cela correspond aux sous-ensembles S_i^j qui sont symbolisés par les traits gris sur la figure 4.1.

Cette information connue, le premier joueur (I) effectue son choix en misant une de ses trois cartes. Ce choix étant présenté au deuxième joueur, ce dernier possède une information supplémentaire pour effectuer son propre premier choix.

Aux feuilles de l'arbre (nœuds sans suivants) sont associés les vecteurs de gains. Par exemple, si la troisième série était ordonnée comme suit : 2,1 puis 3 ; si le premier joueur a misé 3 au premier tour alors que le deuxième joueur a misé 1 puis qu'il a misé 1 au deuxième tour tandis que le deuxième joueur a misé 3 alors, le vecteur correspondant est (1,-1). Cela signifie que le premier joueur a gagné 1 point et le second perdu 1 point.

4.3.2 Arbre simplifié

En fait, la représentation sous forme extensive est uniquement judicieuse lorsqu'on considère des jeux à information complète et parfaite. Dans ce contexte, si les joueurs interviennent les uns après les autres suivant un ordre précis, et s'ils ont un nombre fini de choix possibles à chaque étape, on peut construire un arbre beaucoup plus simple. Un tel arbre s'appelle un arbre de Kuhn simplifié.

4.3.3 Exemple

Imaginons la situation suivante : deux marchands de glaces européens décident de s'installer sur une plage et se rencontrent le soir précédant l'installation. Le premier marchand (noté A) doit écrire ses tarifs sur un tableau et donner ensuite la peinture au deuxième (noté B) pour qu'il écrive les siens. Pour simplifier les transactions, le prix d'une glace doit pouvoir être payé à l'aide d'une seule pièce et doit être supérieur au prix de revient (10 centièmes d'□). Les possibilités sont : 20 centièmes ou 50 centièmes d'□. Quand le premier joueur aura choisi son prix et l'aura écrit, le deuxième joueur pourra choisir et écrire le sien.

L'arbre de Kuhn correspondant pourrait être le suivant :

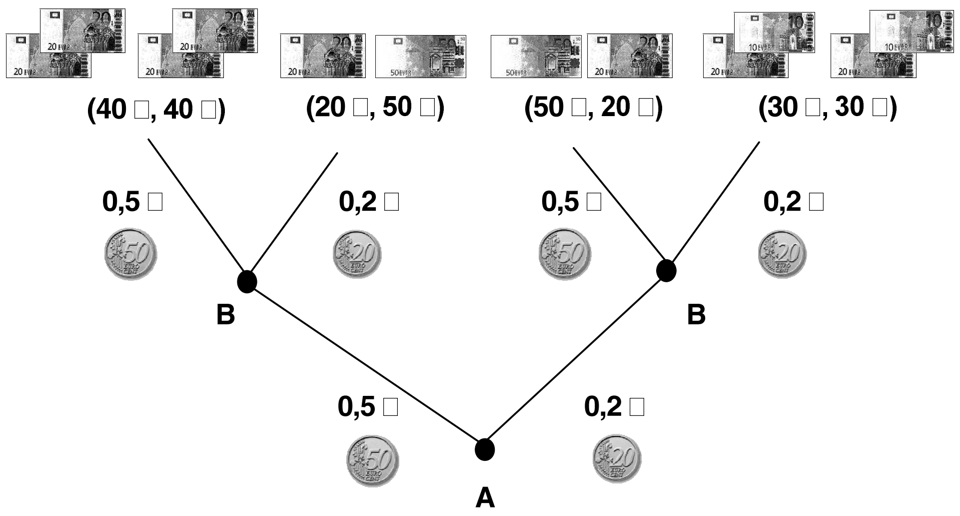


Figure 4.2. Jeu des glaciers.

Dans ce cas, il n'y a pas d'incertitudes pour B. Il a tous les éléments en main avant de faire son choix. L'information est complète et parfaite en ce sens que A connaît toutes les règles du jeu et toutes les conséquences de ses choix en fonction de celles possibles de B. Par exemple, si A fixe son prix à 0,5 €, B a intérêt à fixer le sien à 0,2 € parce que dans ce cas, A gagnera 20 € tandis que lui en gagnera 50.

L'issue du jeu va dépendre de la façon dont A et B vont se comporter. Pour effectuer un choix judicieux, ces derniers vont étudier tous les cas de figure possibles.

4.3.4 Résolution des jeux sous forme extensive simplifiée

4.3.4.1 Récurrence à rebours

Pour résoudre les jeux à information complète et parfaite sous cette forme, chaque joueur peut se mettre en pensées à la place de chacun des autres afin de calculer la stratégie optimale de chacun d'entre eux.

Dans l'exemple précédent, quand le joueur A doit prendre sa décision, il peut se 'mettre à la place du joueur B'. Se faisant, il lui est facile de remarquer que si lui-même établit son prix à 0,5 € en espérant partager équitablement les 80 €, il sera judicieux de la part du joueur B de fixer son prix à 0,2 € pour gagner 50 €, ne laissant au joueur A que 20 €. Ce dernier va donc rationnellement choisir de fixer son prix à 0,2 € ce qui lui garantit au minimum (quel que soit le choix du joueur B) 30 €.

Ce choix étant fait, le joueur B va fixer son propre prix à 0,2 □. Cela correspond à la solution du jeu. Un lecteur, même peu averti, constatera que cette solution est loin d'être optimale, les deux joueurs connaissant cette solution auraient tout intérêt à s'entendre et à fixer leur prix à 0,5 □ : C'est là toute l'ambiguïté de la notion de rationalité dans les jeux. Par ailleurs, se pose le problème de la confiance et de la duperie des joueurs. En effet, rien ne garantit que, s'étant initialement entendu avec le premier joueur pour aboutir à la solution (40 □,40 □), le deuxième joueur tienne sa promesse. La rationalité des joueurs concerne donc plus l'assurance d'un gain minimum que la recherche du gain maximal.

4.3.4.2 Jeux itérés

La plupart des jeux ne comportent pas qu'un coup. Dans ce cas, les coups s'enchaînent les uns après les autres. Pour effectuer le choix initial, le premier joueur peut dérouler mentalement l'ensemble des configurations possibles et peut raisonner à rebours de façon récursive.

4.3.4.3 Méthodes calculatoires

Les méthodes calculatoires de résolution correspondantes sont les algorithmes de recherche des stratégies Minimax ou Maximin, les algorithmes Alpha/Béta, etc.

Nous reviendrons dans ce chapitre sur le théorème du Minmax proprement dit. Ce théorème établit la preuve que tout jeu à information complète (qu'elle soit parfaite ou imparfaite) présente au moins une stratégie optimale pour chacun des joueurs. Une stratégie optimale est une stratégie qui garantit un gain minimum. Dans l'exemple précédent, on a pu déterminer cette stratégie optimale pour les deux joueurs parce qu'il était possible de raisonner à rebours depuis l'issue du jeu et parce qu'on pouvait déterminer les issues possibles du jeu par le calcul.

Dans certains cas, comme le jeu des échecs, la combinatoire des possibilités de jeu est trop importante pour permettre un parcours exhaustif de toutes les possibilités du jeu.

Dans ces cas où l'arbre de recherche est trop important pour un parcours exhaustif, on a recours à des méthodes heuristiques permettant de parcourir uniquement des sous-arbres de l'arbre global.

La première méthode s'appuie directement sur le théorème du Minmax.

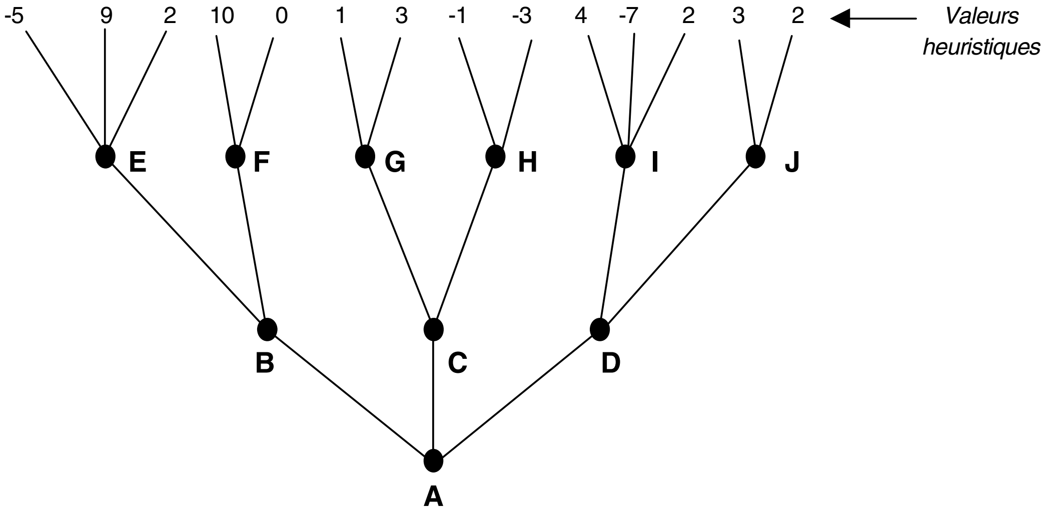
4.3.4.3.1 MinMax

Cette méthode s'applique quand il est possible d'explorer, à partir d'une situation du jeu, un nombre limité de coups. Elle nécessite une méthode (heuristique) permettant d'estimer la valeur d'une quelconque position du jeu (une anticipation calculée du gain éventuel résultant des issues possibles du jeu à partir de toute position). Cette évaluation d'une situation est appelée fonction d'évaluation du jeu.

A partir de la situation du jeu, on détermine l'arbre des coups possibles à la profondeur maximale que l'on peut explorer et on attribue aux feuilles de ce sous-arbre les valeurs heuristiques correspondantes.

Exemple

Soit le sous-arbre suivant construit à partir d'une position (A) d'un jeu global (non représenté).



Considérons le joueur dont c'est le tour en A. L'algorithme pratique consiste à raisonner à rebours sur les valeurs heuristiques des gains estimés pour ce joueur après son deuxième coup.

Les valeurs maximales des feuilles qu'il pourra atteindre sont :

- Pour E : 9
- Pour F : 10
- Pour G : 3
- Pour H : -1
- Pour I : 4
- Pour J : 3

MAX des feuilles pour le premier joueur

Si on considère alors à rebours les choix possibles pour le deuxième joueur. On calcule les minima des maxima établis à l'étape précédente.

- Pour B : 9
- Pour C : -1
- Pour D : 3

MIN des feuilles pour le premier joueur

Finalement, on considère les choix pour le premier joueur en A. On calcule pour cela le maximum des minima établis précédemment. Dans l'exemple décrit, il s'agit de la valeur 9 obtenue pour B. La stratégie MinMax établit donc que le premier joueur doit choisir l'action menant à B.

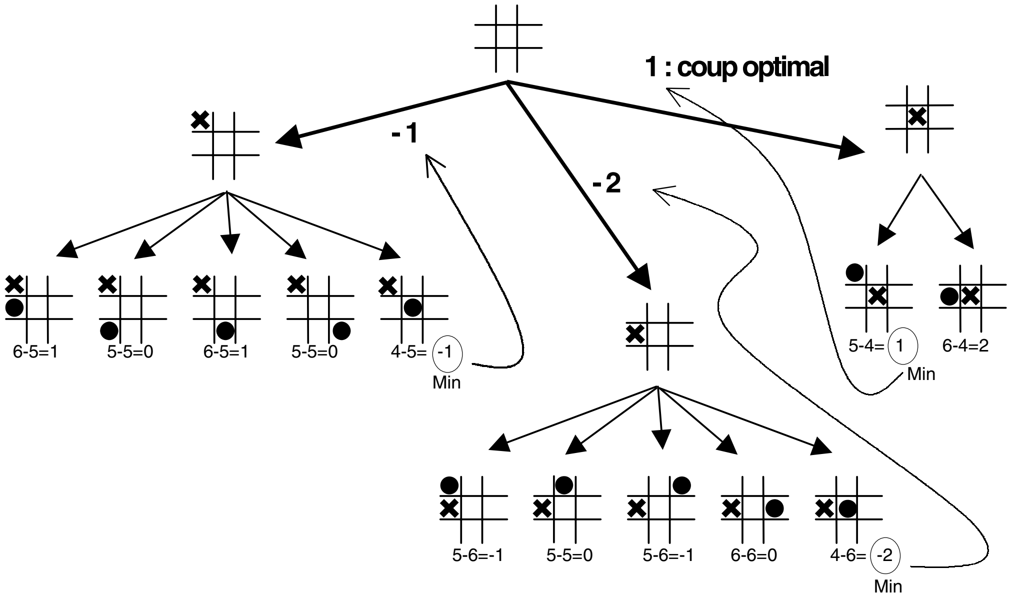
Les performances d'un programme utilisant la méthode *MinMax* dépendent de deux facteurs :

- La qualité de la fonction d'évaluation ;
- La profondeur de recherche accordée au programme.

La profondeur correspond au nombre de demi-coups (coups d'un joueur) envisagés par le programme à chaque évaluation.

Pour des jeux simples comme le tic-tac-toe, une bonne fonction d'évaluation peut être très simple à découvrir. Par exemple, une bonne fonction pour ce jeu est : $f(S)=(Nb_j(s) - Nb_a(S))$ où Nb_j est le nombre de colonnes et diagonales ouvertes pour le joueur et Nb_a, le nombre de colonnes ou diagonales ouvertes à l'adversaire. La méthode du MinMax s'applique très bien avec cette fonction d'évaluation et aboutit à la solution optimale même si l'arbre n'est parcouru qu'à une profondeur de 1.

La méthode du MinMax avec cette fonction pour le Tic-tac-toe peut se représenter ainsi pour le premier coup :



Shannon a proposé dès 1950 un programme pour le jeu d'échecs reposant sur la procédure du MinMax [Shannon, 1950]. Newell, Shaw et Simon ont utilisé les idées de Shannon pour construire un nouveau programme en 1958 [Newell, 1957, 1958]. Il faut croire que la fonction d'évaluation n'était pas très judicieuse, ou bien que la profondeur de recherche était beaucoup trop faible, parce qu'il fut battu par un enfant débutant de 10 ans ! L'originalité de l'approche était l'introduction d'une nouvelle idée, la méthode Alpha/béta.

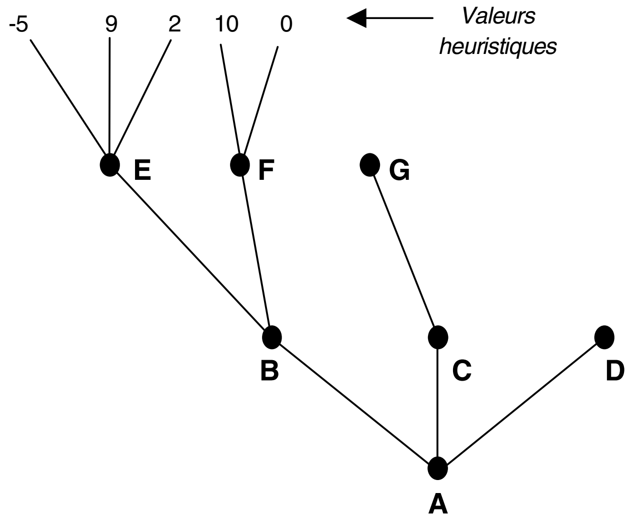
4.3.4.3.2 Alpha-béta

La méthode du MinMax, bien qu'elle puisse être très efficace, présente toutefois un inconvénient majeur : elle oblige à 'parcourir' tout le sous-arbre construit à partir de la position de jeu considérée.

Il est possible d’optimiser cette recherche par un algorithme astucieux.

Exemple

Imaginons, dans le cas de l’exemple précédent, que l’on n'ait développé qu'une partie du sous-arbre précédent :



En ayant développé le coup B, on a établi les valeurs de E,F et B :

Pour E :	9	MAX des valeurs de E.
Pour F :	10	MAX des valeurs de F.
Pour B :	9	MIN des valeurs de E et F.

Ainsi, si on peut établir que la valeur de C est inférieure à 9, il devient inutile de développer la branche correspondante. Pour cela, il suffit d’établir que la valeur de G est inférieure à 9 (car la valeur de C est le minimum de celle de G et H). Ainsi, le seul fait de déterminer les valeurs heuristiques associées à G suffit à rendre inutile le parcours exhaustif de la branche C.

On peut faire le même raisonnement avec la branche D. En effet, toutes les autres valeurs heuristiques du sous-arbre sont inférieures à 9.

Les alpha et béta qui ont donné le nom à la méthode sont les seuils au delà ou en deçà desquels il est inutile de développer les branches correspondantes.

La première description de l'algorithme alpha/béta est due à Newell, Shaw et Simon [Newell, 1958] qui l'avaient présentée en même temps que leur programme d'échecs. Une analyse plus précise de la méthode a été réalisée par Knuth et Moore en 1975 [Knuth, 1975]. Des résultats complémentaires ont été exposés par Newborn [Newborn, 1977] puis par Baudet [Baudet, 1978].

4.4 Stratégies : forme normale

Par stratégie on entend une fonction qui associe pour chaque ensemble d'information S_i^j d'un joueur i un arc conduisant à un nœud suivant du nœud associé à S_i^j.

L'ensemble des stratégies d'un joueur est noté Σ_i.

Pour déterminer son choix, un joueur tente de maximiser son gain (il tente de maximiser la i^{ème} composante de la fonction de gain). Dans le cas général (*avec un joueur Nature*), il n'est pas possible de déterminer à l'avance les conséquences d'un choix. Il est alors judicieux de prendre en compte l'espérance de gain.

Celle-ci se définit comme :

$$\Pi(\sigma_1, \sigma_2,\ldots, \sigma_n) = (\Pi_1(\sigma_1, \sigma_2,\ldots, \sigma_n), \Pi_2(\sigma_1, \sigma_2,\ldots, \sigma_n), \ldots, \Pi_n(\sigma_1, \sigma_2,\ldots, \sigma_n)) \qquad (\sigma_i \in \Sigma_i)$$

Il est ainsi possible de calculer l'espérance de gain $\Pi(\sigma_1, \sigma_2,\ldots, \sigma_n)$ pour toutes les valeurs possibles de $\sigma_1, \sigma_2,\ldots, \sigma_n$. Les valeurs correspondantes peuvent être présentées sous forme d'une matrice de dimension n (nombre de joueurs). Cette représentation matricielle des valeurs de gain ou d'espérance de gain est appelée forme normale du jeu.

4.4.1 Exemple

Considérons le célèbre jeu Pierre/Ciseaux/Papier[1].

La forme normale de ce jeu peut être présentée comme suit :

		Joueur A		
		Pierre	Ciseaux	Papier
Joueur B	Pierre	(0 / 0)	(-1 / 1)	(1 / -1)
	Ciseaux	(1 / -1)	(0 / 0)	(-1 / 1)
	Papier	(-1 / 1)	(1 / -1)	(0 / 0)

---

[1] Dans certaines provinces de chine, l'équivalent de ce jeu est « Homme / Poulet / Vers» (l'Homme mange le poulet qui mange les vers qui, fatalement, un jour…).

Chaque case de la matrice représente le vecteur de gains pour chacun des deux joueurs, le gain de A puis celui de B.

4.5 Equilibres

La notion d'équilibre est fondamentale dans la théorie des jeux, elle est la clé des calculs menant à la résolution des jeux.

4.5.1 Définition

Un ensemble particulier de stratégies des n joueurs d'un jeu est dit *en équilibre* si aucun d'entre eux n'a un quelconque intérêt à changer unilatéralement sa stratégie. Mathématiquement, cela se traduit de la façon suivante :

Etant donné un jeu Γ, on dit qu'un ensemble de stratégies $(\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)$ des n joueurs est en équilibre si et seulement si :

$$\forall i \in [1..n] \text{ et } \sigma_i \in \Sigma_i \text{ , } \Pi_i(\sigma_1^*,\ldots,\sigma_n^*) \geq \Pi_i(\sigma_1^*,\ldots, \sigma^*_{i-1}, \sigma_i \text{ , } \sigma^*_{i+1}, \ldots, \sigma_n^*)$$

4.5.2 Exemples

Soit le jeu suivant :

		Joueur A	
		A_1	A_2
	B_1	(0 / 0)	(4 / 3)
Joueur B	B_2	(1 / 2)	(-1 / 0)

Les deux ensembles de stratégies (A_1,B_2) et (A_2,B_1) sont des équilibres. On remarque en effet que, dans ces deux cas, aucun des deux joueurs n'a intérêt à modifier unilatéralement sa stratégie. Par exemple, pour (A_1,B_2), si A change de stratégie pour adopter A_2 alors que B ne modifie pas la sienne, son gain passera de 1 à –1. De même, si B change de stratégie pour adopter B_1 alors que A conserve la sienne, son gain passera de 2 à 0.

Considérons maintenant le jeu de pair/impair dont la forme normale est la suivante :

		Joueur A	
		Pair	Impair
	Pair	(1 / -1)	(-1 / 1)
Joueur B	Impair	(-1 / 1)	(1 / -1)

Remarque : le jeu consiste pour les deux joueurs à choisir secrètement entre pair et impair puis à annoncer simultanément leur choix. Si les deux joueurs ont choisi la même chose, le joueur A est déclaré vainqueur, sinon, c'est B.

On peut constater qu'il n'y a aucun équilibre. Pour chaque couple de stratégies de A et B, un des deux joueurs a systématiquement intérêt à modifier sa stratégie. Ce cas est typique des jeux dits *à somme nulle* (la somme des gains de l'ensemble des joueurs est nulle) sur lesquels nous reviendrons plus loin.

4.5.3 Théorème

Pour tout jeu fini à information complète et parfaite concernant n joueurs, il existe au moins un ensemble de stratégies formant un équilibre.

4.6 Des jeux particuliers

Les jeux les plus étudiés sont les jeux à somme nulle, à information complète et imparfaite, concernant deux joueurs. De fait, les autres jeux à information complète et imparfaite sont des extensions de ces jeux et les théories développées autour d'eux découlent, avec des ajustements parfois complexes, des jeux à deux joueurs et à somme nulle.

Les principaux ajustements concernent les jeux à n joueurs dans lesquels certains joueurs peuvent se coaliser. Dans ce cas, il faut considérer chaque groupe de coalition comme un joueur unique pour exploiter les propriétés des jeux purement compétitifs.

La notion de somme nulle interdit aux joueurs de trouver des intérêts communs et pallie le principal problème de la rationalité.

4.6.1 Définition

Un jeu Γ est à somme nulle si et seulement si, pour toute feuille de son arbre, la fonction de gain $(p_1,\ldots,p_n)$ vérife :

$$\sum_{i=1}^{n} p_i = 0$$

Dans le cadre plus précis des jeux à deux joueurs, cela signifie que le gain d'un des joueurs est toujours compensé par une perte équivalente pour l'autre joueur. Ces jeux sont dits purement compétitifs.

4.6.2 Théorème

Soit Γ un jeu à deux joueurs à somme nulle et soient (σ_1 , σ_2) et (τ_1 , τ_2) deux équilibres de Γ, alors :

(i) (σ_1 , τ_2) et (τ_1 , σ_2) sont aussi des équilibres et,

(ii) $\pi(\sigma_1 , \sigma_2) = \pi(\tau_1 , \tau_2) = \pi(\sigma_1 , \tau_2) = \pi(\sigma_1 , \tau_2)$.

4.6.3 Point selle

Remarque : pour les jeux à somme nulle, on indique uniquement les gains d'un joueur, les gains de l'autre joueur étant les opposés par définition. Ainsi, dans les matrice de la forme normale des jeux n'apparaissent que les gains du premier joueur (le joueur 'ligne').

4.6.3.1 Définition

Soit A la matrice d'un jeu à deux joueurs à somme nulle et soient a_{ij} les gains associés au joueur *ligne* (dans le cas où il a choisit sa $i^{ème}$ stratégie alors que son adversaire a opté pour sa $j^{ème}$ stratégie). La combinaison de stratégies (i,j) est un équilibre du jeu si et seulement si a_{ij} est à la fois le maximum de la $j^{ème}$ colonne et le minimum de la $i^{ème}$ ligne. Dans ce cas, on dit que la combinaison (i,j) forme un point selle du jeu.

4.6.3.2 Exemple

Soit le jeu suivant :

$$\begin{pmatrix} 9 & 1 & 5 \\ 5 & 3 & 7 \\ -5 & -1 & -1 \end{pmatrix}$$

Il possède un point selle à la seconde ligne pour la seconde colonne. Si le joueur *Ligne* choisit sa deuxième stratégie et le joueur *Colonne* également, aucun des deux n'a intérêt à changer unilatéralement de stratégie : il s'agit bien d'un équilibre.

4.6.4 Stratégies mixtes

4.6.4.1 Définition

Une stratégie mixte pour un joueur est une répartition probabiliste sur l'ensemble de ses stratégies possibles. Dans le cas où cet ensemble est fini (de cardinal k), une stratégie mixte est un vecteur à k dimensions x = (x_1, x_2, …, x_k) tel que :

$$x_i \geq 0 \text{ et } \sum_{i=1}^{k} x_i = 1$$

4.6.4.2 Valeurs d'un jeu en stratégies mixtes

Soit X l'ensemble des stratégies mixtes possibles pour le joueur L (*Ligne*) et soit Y l'ensemble des stratégies mixtes possibles pour le joueur C (*Colonnes*). Si L et C jouent à un jeu à deux joueurs à somme nulle dont la matrice est A alors, si L utilise une stratégie mixte x et C une stratégie mixte y, l'espérance de gain pour L est :

$$A(x, y) = \sum_{i=1}^{l} \sum_{j=1}^{c} x_i a_{ij} y_j$$

où l et c sont respectivement le nombre de lignes et de colonnes de A .

Cette formule s'écrit également :

$$A(x, y) = xAy^T$$

Ainsi, L doit raisonner en supposant que C va tenter de minimiser A(x,y). Ainsi, son gain minimal va être :

$$v(x) = \min_{y \in Y}(xAy^T)$$

Dans ces conditions, xAy^T est la moyenne du gain attendu pour L s'il utilise x contre les stratégies pures de C. Le minimum sera obtenu pour une stratégie pure j de C :

$$v(x) = \min_j xA._j \quad (A._j \text{ est la } j^{\text{ème}} \text{ colonne de A}).$$

Le joueur L doit donc maximiser v(x) pour obtenir :

$$v_L = \max_{x \in X} \min_j xA._j \, .$$

La stratégie x obtenue est appelée la stratégie maximin du joueur L.

De la même façon, le joueur C va choisir une stratégie parmi les stratégies y définies par :

$$v(y) = \max_i A_i y^T$$

afin d'obtenir :

$$v_C = \min_{y \in Y} \max_i A_i y^T$$

La stratégie y obtenue est appelée stratégie minimax pour le joueur C.

v_L et v_C sont appelées respectivement valeurs du jeu pour le joueur C et le joueur L.

4.6.5 Le théorème du MinMax

Le théorème du MinMax, établi par Von Neumann en 1928, indique que : $v_L = v_C$ [1]

Le mathématicien Nash a laissé son nom à la notion d'équilibre de Nash qui est une combinaison de stratégies des différents joueurs telle qu'aucun d'entre eux ne peut espérer augmenter son gain en changeant unilatéralement de stratégie.

En information complète et parfaite, certains jeux, mêmes très simples, peuvent ne pas avoir d'équilibre de Nash ou alors en avoir plusieurs, souvent non comparables selon le critère de Paréto. Pour cette raison, les théoriciens ont introduit la notion de *stratégie mixte*.

Nash a démontré que tout jeu à information complète mais imparfaite comporte au moins un équilibre en stratégies mixtes. L'équilibre correspondant est un équilibre de Nash : toute modification unilatérale par un joueur des probabilités qu'il a attribuées à ses stratégies pures ne peut entraîner une hausse de son espérance de gain.

---

[1] lorsque $v_L = v_C = 0$, le jeu est équitable (les deux joueurs ont une espérance de gain nulle).

Le théorème du MinMax constitue un cas particulier du théorème de Nash concernant les équilibres.

Ce théorème est fondamental car il peut donner un but à un joueur : trouver le point d'équilibre qui lui garantisse son espérance de gain. Pour autant, la théorie ne fournit pas la solution pour déterminer cet équilibre… D'autre part, les équilibres peuvent être multiples et certains peuvent être faibles, ce qui implique que la modification de stratégie ne peut augmenter l'espérance de gain, tout en ne la diminuant pas nécessairement. Dans ces conditions, un joueur n'est pas obligé de s'y tenir.

4.6.6 Le théorème des stratégies optimales

Ce théorème indique que pour un jeu de matrice A(l,c), soit le joueur C a une stratégie optimale y avec $y_c > 0$, soit le joueur L a une stratégie optimale x telle que :

$$\sum_{i=1}^{l} a_{ic} x_i > v$$

4.6.7 Détermination des stratégies optimales

Si le théorème du MinMax garantit l'existence de stratégies optimale pour les jeux à deux joueurs à somme nulle, il n'indique pas la façon de les déterminer.

Dans le cas des jeux les plus simples, la détermination de ces stratégies est aisée comme nous allons le voir.

4.6.7.1 Point selle

Dans le cas où un jeu présente un point selle, il constitue lui-même une stratégie optimale pour les joueur L et C.

Ces stratégies sont de pures stratégies et sont équivalentes à des stratégies mixtes avec tous les coefficients de probabilité nuls sauf pour les stratégies correspondant au point selle pour lequel les coefficients valent 1.

4.6.7.2 Stratégies dominées

Pour un jeu représenté par la matrice A, on dit que la $i^{ème}$ ligne domine la $j^{ème}$ ligne si :

$$\forall k, a_{ik} \geq a_{jk} \text{ et } \exists k / a_{ik} > a_{jk}$$

de la même façon, on dit que la $i^{ème}$ colonne domine la $j^{ème}$ colonne si :

$$\forall k, a_{ki} \leq a_{kj} \text{ et } \exists k / a_{ki} < a_{kj}$$

Si dans la matrice A, les lignes $i_1,i_2,\ldots,i_k$ sont dominées alors le joueur L a une stratégie optimale x telle que :

$$x_{i_1} = x_{i_2} = \ldots = x_{i_k} = 0$$

De plus, toute stratégie optimale pour un jeu obtenu en éliminant les lignes dominées est également une stratégie optimale pour le jeu initial.

De la même façon, on peut réduire les cas où il existe des colonnes dominées.

Exemple

Soit le jeu suivant :

$$\begin{pmatrix} 0 & 1 & 2 & 4 \\ 2 & 1 & 1 & 2 \\ 0 & 2 & 3 & 5 \end{pmatrix}$$

on remarque aisément que la deuxième colonne domine la quatrième que l'on peut donc éliminer :

$$\begin{pmatrix} 0 & 1 & 2 & \cancel{4} \\ 2 & 1 & 1 & \cancel{2} \\ 0 & 2 & 3 & \cancel{5} \end{pmatrix}$$

On remarque alors que la troisième ligne domine maintenant la première que l'on peut donc éliminer :

$$\begin{pmatrix} \cancel{0} & \cancel{1} & \cancel{2} & \cancel{4} \\ 2 & 1 & 1 & \cancel{2} \\ 0 & 2 & 3 & \cancel{5} \end{pmatrix}$$

Maintenant, la deuxième colonne domine la troisième que l'on élimine donc :

$$\begin{pmatrix} \cancel{0} & \cancel{1} & \cancel{2} & \cancel{4} \\ 2 & 1 & \cancel{1} & \cancel{2} \\ 0 & 2 & \cancel{3} & \cancel{5} \end{pmatrix}$$

pour obtenir un jeu 2x2 qu'il est très facile de résoudre comme nous allons le montrer.

4.6.7.3 Les jeux 2x2

Soit un jeu A :

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

Si le jeu possède un point selle, il est résolu immédiatement, sinon, on cherche les stratégies optimales x(x_1,x_2) et y(y_1,y_2) sachant que :

$$a_{11}x_1y_1 + a_{12}x_1y_2 + a_{21}x_2y_1 + a_{22}x_2y_2 = v$$

ce que l'on peut écrire également :

$$x_1(a_{11}y_1 + a_{12}y_2) + x_2(a_{21}y_2 + a_{22}y_2) = v$$

Comme x_1 + x_2 = 1, qu'ils sont positifs tous les deux et que y est une stratégie optimale, les termes entre parenthèse sont égaux à v.

De la même façon, $(a_{11}x_1 + a_{12}x_2) = v$ et $(a_{12}x_1 + a_{21}x_2) = v$

Ce que l'on peut écrire :

$$Ay^T = \begin{pmatrix} v \\ v \end{pmatrix} \text{ et } xA = (v, v)$$

on obtient alors :

$$x = vJA^{-1}$$

Où J est le vecteur (1,1). En éliminant v, on peut écrire :

$$x = \frac{JA^{-1}}{JA^{-1}J^T} \text{ et } y = \frac{A^{-1}J^T}{JA^{-1}J^T}$$

Ce que l'on peut encore écrire sous la forme d'un théorème usuel :

Si A est un jeu 2x2 et qu'il ne possède pas de point selle alors, sa solution optimale unique et sa valeur sont données par :

$$x = \frac{JA*}{JA*J^T}$$

$$y = \frac{A*J^T}{JA*J^T}$$

$$v = \frac{|A|}{JA * J^T}$$

où A* est l'adjointe de A, |A|, le déterminant de A et J le vecteur (1,1).

Exemple

Soit le jeu suivant :

$$A = \begin{pmatrix} 1 & 0 \\ -1 & 2 \end{pmatrix}$$

Ce jeu n'a pas de point selle et on peut donc appliquer le théorème précédent.

On a :

$$A * = \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix}$$

et |A|=2 ; JA*=(3,1) ;a*J^T=(2,2) et JA*J^T = 4 donc,

$$x = \left(\frac{3}{4}, \frac{1}{4} \right) \qquad y = \left(\frac{1}{2}, \frac{1}{2} \right) \qquad v = \frac{1}{2}$$

Une autre méthode de calcul est la suivante :

L'espérance de gain du joueur L est $1 * x_1 * y_1 + 0 * x_1 * y_2 - 1 * x_2 y_1 + 2 * x_2 y_2$

Soit $v = y_1(x_1 - x_2) + y_2(2x_2)$

Si $y_1 = y_2$,alors on obtient $y_1(x_1 + x_2)$. Ce qui donne une valeur indépendante de x_1 et x_2 puisque $x_1 + x_2 = 1$ par définition.

Cela donne une solution optimale y=(½ ,½) et $v = y_1 = ½$.

De la même façon, on peut chercher comment rendre l'espérance de gain indépendante de Y. Pour cela, on écrit : $v = x_1(y_1) +x_2(2y_2-y_1)$. Si $x_1=3x_2$, on obtient $v = 2\ x_1(y_1+y_2)$ qui est indépendant de y_1 et y_2. Ainsi, $x_1= 3x_2$ et $x_1+x_2=1$ donc x_1= ¾ ; x_2= ¼ et v = 2 * ¼ = ½.

Cette méthode est très pratique et s'applique à d'autres types de jeux.

Considérons, par exemple, le jeu du 'Baccarat du bagne' (autrement appelé Pierre/Ciseaux/Papier).

Sous sa forme stratégique, il se représente ainsi :

		Joueur A		
		Pierre (Ar)	Ciseaux (Ac)	Papier (Ap)
Joueur B	Pierre (Br)	0	-1	1
	Ciseaux (Bc)	1	0	-1
	Papier (Bp)	-1	1	0

Ar, Ac et Ap sont les probabilités de A de choisir respectivement Pierre, Ciseaux et Papier. Les gains donnés par le tableau sont ceux de A et sont symétriques pour B.

L'Equilibre de Nash consiste à trouver, pour A, une stratégie mixte telle que les choix de B n'aient aucune influence sur les résultats. On calcule ainsi :

Espérance_de_gain(A)= 0*Ar*Br + 1*Ar*Bc + (-1)*Ar*Bp

+ (-1)*Ac*Br + 0*Ac*Bc + 1*Ac*Bp

+ 1*Ap*Br + (-1)*Bc*Ap + 0*Ap*Bp

Soit, Br*(Ap-Ac) + Bc*(Ar-Ac) + Bp*(Ap-Ar). Cette espérance est indépendante du comportement de B si Ap=Ac=Ar. La somme des probabilités des trois stratégies devant être égale à 1, on en déduit que la stratégie optimale à ce jeu contre un adversaire quelconque est de joueur 1/3 Pierre, 1/3 Ciseaux et 1/3 Papier. C'est une répartition particulière parce qu'elle est uniforme. Ce résultat est tout à fait logique compte tenu de la symétrie de la matrice des gains.

4.6.8 Jeux 2 x n et jeux m x 2

Il s'agit de deux formes également simples à résoudre de jeux à deux joueurs à somme nulle.

Les principes des jeux 2x2 s'appliquent presque directement. Par exemple, dans le cas 2 x n, Le premier joueur doit maximiser

$$v(x) = \min_j (a_{1j}x_1 + a_{2j}x_2)$$

Comme $x_1 + x_2 = 1$, on a :

$$v(x) = \min_j ((a_{2j} - a_{1j})x_2 + a_{1j})$$

Ce qui peut se résoudre facilement par une méthode graphique.

Exemple

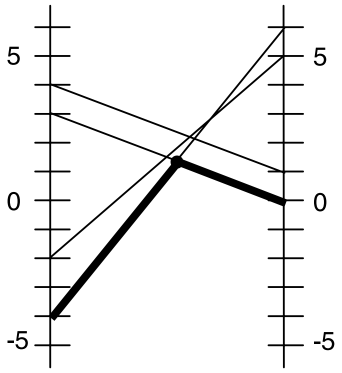
Soit le jeu défini par :

$$\begin{pmatrix} -4 & -2 & 3 & 4 \\ 6 & 5 & 0 & 1 \end{pmatrix}$$

Il ne présente pas de point selle et ne peut donc pas être résolu directement.

La 4ème stratégie du joueur C est dominée par sa 3ème ce qui évite d'avoir à la considérer.

On trace alors la figure suivante :



Le maximum de l'enveloppe minimale de la courbe donne la stratégie optimale. Ici, Il s'agit du point intersection de y_1 et y_3. Le jeu précédent se réduit alors à la matrice 2x2 suivante :

$$\begin{pmatrix} -4 & 3 \\ 6 & 0 \end{pmatrix}$$

qui ne pose aucun problème de résolution. Les stratégies optimales sont x=(6/13 ; 7/13) et y= (3/13 ; 0 ;10/13 ;0). La valeur du jeu est : v = 18/13.

On vérifie bien que –4*(6/13)+6*(7/13) = 3*(6/13) = 18/13 et que –4*(3/13)+3*(10/13) = 6*(3/13) = 18/13.

4.6.9 Cas général

Dans le cas général, la résolution d'un jeu implique celle d'un système d'inéquations dont les inconnues sont les coefficients probabilistes associés à chaque stratégie pure définissant la répartition probabiliste optimale au sens de Nash.

La méthode de calcul mathématique la plus souvent implémentée pour ce type d'optimisation linéaire est la méthode de programmation linéaire dite du *Simplex*. La plupart des jeux à information complète et imparfaite ayant fait l'objet d'implémentation informatique l'ont été de cette façon.

Le détail de l'algorithme correspondant dépasse le cadre de ce manuscrit[1] mais pourra être consulté dans [Press, 1986] ou [Owen, 1995].

4.7 Les limites de la théorie des jeux

4.7.1 Jeux à information complète et parfaite

Dans ce cadre très restreint des jeux, la théorie fournit des solutions mais ces dernières restent souvent purement théoriques à cause d'une mise en œuvre irréalisable.

La théorie suppose que les joueurs ont un comportement rationnel et cherchent donc à maximiser leurs gains tout en minimisant leurs pertes. Il est ainsi possible, l'information étant complète, de se 'mettre à la place de chacun des joueurs' et de calculer son comportement.

---

[1] Toutefois, nous avons réalisé un tel programme pour le jeu ALESIA à titre de comparaison avec notre propre approche. Nous décrirons donc une implémentation particulière de cet algorithme dans la partie consacrée à la méthode S.A.G.A.C.E.

Pour résoudre le jeu, il ‘suffit’ de développer l’ensemble de toutes les actions possibles de chaque joueur jusqu’à la fin de la partie (représentation sous forme d’arbre appelé arbre de Kuhn) puis de raisonner par récurrence à rebours. Les méthodes calculatoires correspondantes sont le Minimax, le Maximin, les algorithmes Alpha/Béta etc.

Il existe théoriquement une solution à tout jeu à information complète et parfaite. Pour autant, certains jeux ne sont pas résolvables pratiquement. A l’heure actuelle, par exemple, le jeu d’échecs n’est pas résolu. Bien que les techniques calculatoires qui s’appuient sur la théorie aient donné d’excellents résultats dernièrement (cf. Deep Blue dont nous avons déjà parlé), il n’est pas encore possible de déterminer La Stratégie gagnante aux échecs. Le fait est que la combinatoire (l’ensemble des coups à envisager) est beaucoup trop forte pour qu’un ordinateur puisse ‘estimer’ chaque position possible. La théorie n’est pas ‘praticable’. Il faut savoir que d’autres jeux ont une combinatoire encore bien plus élevée (comme le jeu de Go), pour lesquels la théorie est totalement inutilisable.

De plus, la théorie ne garantit que l’espérance de gain. Souvent, cela correspond uniquement à minimiser les pertes. Un éventuel programme basé sur la théorie des jeux garantissant le nul aux échecs ne serait pas satisfaisant. On lui préférerait certainement (pratiquement et intellectuellement) un programme capable de s’adapter à son adversaire pour tenter de le vaincre quitte à prendre certains risques.

La théorie présente d’autres lacunes très importantes. Le principe de rationalité est un premier écueil qui empêche d’adapter la stratégie d’un joueur aux faiblesses éventuelles d’autres joueurs (parce que la théorie suppose que tous les joueurs sont parfaitement rationnels). D’autre part, la théorie suppose que les joueurs sont tous opposés les uns aux autres et ne peuvent pas collaborer pour maximiser conjointement leurs gains.

Par exemple, supposons un jeu dont l’arbre de Kuhn soit le suivant (Figure 4.3) :

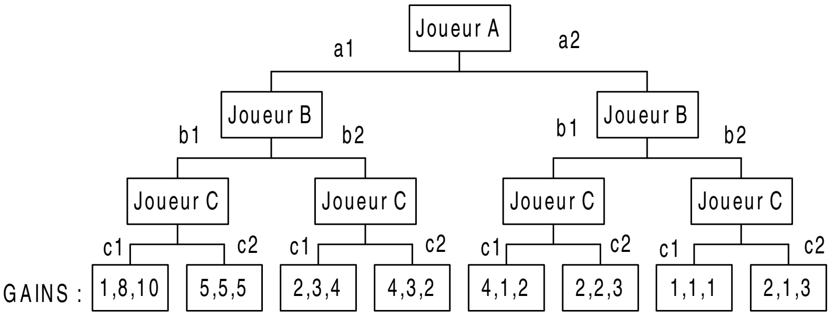


Figure 4.3. Arbre de Kuhn.

Le joueur A peut jouer l'action a1 ou bien l'action a2. Ensuite, quand son action aura été effectuée, B devra effectuer l'action b1 ou b2 et, enfin, C jouera c1 ou c2.

Le principe de la récurrence à rebours va permettre de calculer le comportement de C et ainsi de réduire l'arbre en (Figure 4.4) :

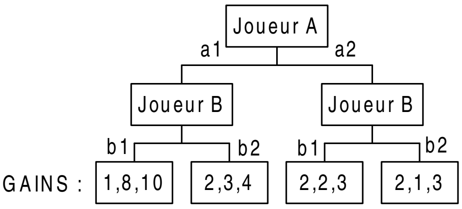


Figure 4.4. Arbre réduit – Niveau 2.

En effet, C va, suivant les différentes actions de B, maximiser son gain. On raisonne alors de la même façon pour B et on obtient l'arbre suivant (Figure 4.5) :

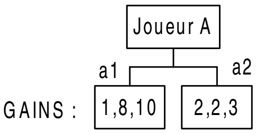


Figure 4.5. Arbre réduit – Niveau 1.

A va donc choisir l'action a2, puis B l'action b1 et enfin C l'action c2 pour un gain respectif de 2, 2 et 3 points. Alors que, si les joueurs s'entendent pour utiliser la Raison au lieu de la Théorie, ils jouent a1, b1 et c2 et gagnent tous 5 points !

4.7.2 Jeux à information complète mais imparfaite

Souvent, la matrice des gains est très complexe et le calcul peut devenir irréaliste. La théorie ne permet alors que d'affirmer que la stratégie optimale existe…

Par ailleurs, même si la matrice est simple, les valeurs qui la constituent peuvent poser des problèmes. Dans le cas des jeux à sommes non nulle par exemple, il est possible que le calcul théorique aboutisse à une solution optimale au sens de Nash mais qui ne soit pas satisfaisante. Rappelons que les équilibres de Nash permettent notamment de définir des solutions stratégiques pour lesquelles aucun des joueurs n'auraient intérêt à modifier unilatéralement sa stratégie ; pour autant, quand un jeu présente plusieurs équilibres de Nash, les solutions théoriques sont ambiguës : elles sont basées sur la rationalité des joueurs mais peuvent ne pas sembler rationnelles.

4.7.2.1 Exemple

Soit le jeu suivant :

$$\begin{pmatrix} (5,5) & (10,6) & (10,4) \\ (6,10) & (3,3) & (6,4) \\ (4,10) & (4,6) & (2,2) \end{pmatrix}$$

Un calcul rapide permet de vérifier que les solutions conduisant aux gains (6,10) et (10,6) sont des équilibres au sens de Nash (Dans les deux cas, aucune modification unilatérale de stratégie ne permet d'augmenter le gain d'un des joueurs).

Si le joueur L (ligne) joue selon sa deuxième stratégie alors que le joueur C (colonne) joue selon sa première, ils se partagent les gains 6 et 10 respectivement. De la même façon, si L joue selon sa première stratégie et C selon sa deuxième, ils se partagent les gains 10 et 6 respectivement.

Il est toutefois évident que le plus intéressant pour L comme pour C est de jouer leur deuxième stratégie. Ainsi, si chacun choisit d'adopter la stratégie qui conduit au plus favorable des deux équilibres de Nash, les gains seront ceux associés aux deuxièmes stratégies de chacun d'entre eux. Ils vont alors obtenir tous les deux un gain de 3.

Se pose alors le problème de rationalité d'une façon différente : comment peut-on dire que cette solution est guidée par la rationalité alors qu'elle fait gagner moins que chacune des deux autres ?

4.7.3 Jeux à information incomplète

Pour ces jeux, la théorie s'appuie sur les équilibres bayésiens et les systèmes de croyances. Certains modèles (tel que celui de Cournot considèrent le comportement des autres joueurs comme indépendants de son propre comportement.

En fait la théorie donne des orientations permettant, plus ou moins, de trouver les bons compromis pour considérer ces jeux comme des jeux à information complète mais imparfaite. Aucune solution n'est donnée par la théorie pour ces jeux, sauf les solutions évidentes de bon sens.

Pour ces jeux, il est courant que la combinatoire soit trop grande parce qu'il faut considérer tous les cas de figure, et notamment ceux dépendant du hasard. Pour un jeu comme le backgammon, par exemple, les techniques purement arborescentes de la théorie des jeux sont inadaptées. C'est pourquoi, le programme de Backgammon BKG 9.8 de Berliner [Berliner, 1980], effectue très peu de recherche dans les graphes d'états mais utilise des connaissances exprimées par des expert du jeu. Le succès de ce programme est impressionnant : il a battu, en 1979, le champion du monde par 7 points à 1.

4.8 Les limites des applications partielles de la théorie des jeux

4.8.1 Les fonctions d'évaluation

Comme nous l'avons décrit précédemment, lorsqu'il n'est pas possible de parcourir de façon exhaustive l'ensemble de l'arbre des coups possibles, et qu'on ne peut donc appliquer complètement la théorie, on a recours à des méthodes heuristiques de recherche basées sur des fonctions d'évaluation des positions du jeu.

La détermination de ces fonctions n'est pas toujours évidente. Chaque joueur peut avoir sa façon d'appréhender un jeu. Ainsi, il n'est pas dit que tous les joueurs évaluent de la même façon une situation (ce qui revient à dire que des joueurs peuvent avoir des fonctions d'évaluation différentes). Dans ces conditions, les éventuels calculs d'optimisation peuvent s'avérer impropres parce qu'il ne reflètent le point de vue que d'un seul joueur.

Si pour certains jeux la forme et les paramètres des fonctions d'évaluations s'imposent à la raison et devraient être partagés par tous les joueurs, dans d'autres ils sont empiriques.

Par exemple, la plupart des experts aux échecs s'accordent sur la forme des fonctions d'évaluation : elles doivent prendre en considération les valeurs des pièces (valeurs, elles-mêmes, heuristiques : 1 point pour un pion, 3 points pour un fou ou un cavalier, 5 pour une tour, etc.), la protection du roi, la mobilité des pièces, etc. Pour autant, la multitude des paramètres autorise un nombre considérable de fonctions d'évaluation, toutes raisonnables du point de vue des experts.

4.8.2 La profondeur de recherche

Pour les méthodes les plus utilisées (Minmax, alpha/béta, A*, etc.), les résultats dépendent en grande partie du nombre de situations qui peuvent être évaluées avant d'effectuer un choix.

L'exemple le plus fameux de programme utilisant ces méthodes est le programme d'échecs Deeper Blue qui a vaincu le champion du monde humain en 1997. Ce programme s'est révélé beaucoup plus efficace que sa version précédente, Deep Blue. La différence majeure d'une version sur l'autre était le nombre de coups pouvant être évalués en une seconde : Deep Blue pouvait évaluer environ 100 millions de positions à la seconde, ce qui correspond à une analyse exhaustive sur 7 coups, tandis que Deeper Blue peut analyser 200 millions de positions à la seconde. Pourtant, une telle différence ne se traduit que par la possibilité d'évaluer complètement un seul demi-coup supplémentaire !

Contre le super ordinateur Deep Blue, Garry Kasparov avait gagné 4 à 2, alors que le champion humain a perdu 3.5 à 2.5 contre Deeper Blue. Certes, il y a sans doute de nombreuses raisons à la défaite du champion mais les experts s'accordent à dire que le fait d'avoir doublé le nombre de situations évaluables en une seconde a considérablement amélioré les performances de la machine. Pour les machines, la profondeur de recherche est un élément clé, pour un joueur humain, ce n'est sans doute pas le cas. Kasparov (qui analyse en moyenne 3 coups par seconde) déclarait qu'il n'analysait peut-être qu'une position pendant que Deep Blue en analysait 30 millions, mais que lui analysait la bonne…

4.9 Intérêt du cadre théorique

Le cadre théorique nous permet de mesurer l'efficacité des techniques que nous avons développées et mises en œuvre.

Nous avons choisi de travailler sur les jeux à information complète et imparfaite d'une part parce qu'ils sont suffisamment complexes pour qu'il soit nécessaire d'apporter des solutions intelligentes et efficaces, et d'autre part parce qu'il est possible de comparer ces solutions à ce qu'aurait pu apporter la théorie.

Par ailleurs, nous estimons qu'il y a plus d'intérêt scientifique à étudier de tels jeux que des jeux dans lesquels la « force brutale » peut être appliquée. En ce sens, les jeux à information complète et parfaite nous intéressent moins même si, pour des jeux comme le Go - auxquels la « force brutale » ne peut s'appliquer-, il reste sans aucun doute beaucoup de sujets de réflexion à explorer.

Les jeux nous paraissent le meilleur moyen de travailler en même temps sur l'Apprentissage, l'Adaptation et l'Anticipation. C'est dans ce cadre de travail que nous avons développé la méthode S.A.G.A.C.E., qui apporte des solutions pratiques et originales à la détermination d'une stratégie efficace et adaptative dans des jeux à information complète mais imparfaite. Les jeux vont nous permettre d'illustrer les résultats obtenus mais ne sont, évidemment, pas le but unique de notre recherche, laquelle est susceptible de nombreuses applications comme cela sera décrit plus loin dans ce document.

Remarque :

La rédaction de ce chapitre s'est inspirée de différents ouvrages sur la question dont [Owen, 1985], [Von Neumann, 1943], [Guerrien, 1995], [Morris, 1994].

5 Apprentissage et Anticipation dans les jeux

> ... Comme je n'étudiais rien, j'apprenais beaucoup.
> Anatole France, 1844-1924

5.1 Apprentissage dans les jeux

L'apprentissage de stratégies est la méthode la plus courante permettant de pallier les limites de la théorie. Elle consiste à découvrir, puis à ajuster, des stratégies efficaces en fonction de l'expérience acquise au cours de parties réelles ou de simulations.

Les jeux se prêtent bien à l'apprentissage parce que les règles d'un jeu permettent de déterminer un bon comportement (on connaît toujours le vainqueur du jeu) et offrent donc une évaluation des stratégies dans leur globalité. Evidemment, il n'est pas directement possible de quantifier l'intérêt de tel ou tel coup et la victoire, sans une analyse précise complémentaire, ne permet que d'évaluer l'ensemble des coups.

5.1.1 Apprentissage par cœur

C'est la forme la plus simple d'apprentissage qui soit utilisée dans les jeux. Elle consiste à enregistrer sous forme de règles formelles (systèmes experts par exemple) des stratégies exprimées par des experts du jeu. C'est également l'implémentation la plus commune des résultats de la théorie. Une fois définies par calculs mathématiques les stratégies optimales, celles-ci sont formalisées et enregistrées dans le programme. Par ailleurs, quand la théorie est inapplicable mais qu'il existe de nombreux experts, ce type d'implémentation peut être relativement efficace. C'est notamment le cas pour de nombreux jeux à information incomplète, comme le poker ou le bridge, ou pour les jeux à trop grande combinatoire comme les échecs ou le go.

Un grand nombre de systèmes experts ont ainsi été développés pour la réalisation de programmes de jeu ou pour celle de bibliothèques d'ouvertures [Buro, 1997].

Les jeux qui se prêtent bien à de telles implémentations sont des jeux dont les caractéristiques tactiques ou stratégiques sont facilement identifiables et pour lesquels il peut donc exister des experts. Se pose toujours le problème connu des systèmes experts : la manière d'extraire les connaissances de l'expert pour les formaliser dans le système[1] [Meyer, 1996b].

---

[1] Nous présenterons (§ 7.5.1.2.4.) une méthode de transfert d'expertise par anticipation s'appuyant sur S.A.G.A.C.E. [Meyer, 1996b].

Parmi les implémentations sous cette forme qui ont eu un succès certain, on notera le programme de poker de Waterman [Waterman, 1970], le programme d'échecs de Wilkins [Wilkins, 1979] et les programmes de bridge de Wasserman [Wasserman, 1970] ou de Popescu [Popescu, 1983] (basé sur le moteur d'inférence Snark et sans aucun parcours d'arbre).

A l'époque où la puissance de calcul des ordinateurs était encore très faible, la programmation du jeu d'échecs était une véritable gageure pour les chercheurs en intelligence artificielle. Aucune machine n'était capable de développer un arbre assez profond pour être efficace. Wilkins a donc cherché à explorer une méthode de programmation moins calculatoire en fournissant des connaissances d'expert à son programme. Cette approche avait déjà été adoptée par Pitrat [Pitrat, 1977] dans son programme Robin. L'idée était, contrairement à l'apprentissage par renforcement des coefficients d'une fonction d'évaluation (qui était jugée trop empirique) d'utiliser des concepts spécifiques et explicites. Les stratégies développées par les programmes de Pitrat ou Wilkins sont du type « agresser le roi », « jouer les échanges quand on est en avance sur le nombre de pièces », etc.

Les principales qualités de ces programmes sont :

- La possibilité de modifier le programme très facilement en modifiant simplement les règles d'expertise.
- Les règles qui ont été utilisées pour effectuer un choix stratégique peuvent être facilement traduites en langage naturel pour expliquer les décisions du système, rendant plus faciles les éventuels ajustements.

Les grands défauts de ces programmes sont :

- La spécialisation : ces programmes sont souvent spécialisés dans les fins de parties qui sont les plus connues des experts.
- Les échecs fréquents sur des problèmes très simples, simplement parce qu'il y a des "vides" dans les connaissances fournies au système.
- La difficulté à gérer des connaissances apparemment contradictoires.

Le programme de poker de Waterman repose sur une vingtaine de règles exprimées par un expert : par exemple, "Si on a une main sûrement gagnante, si l'enjeu est important, et si l'adversaire a relancé, alors, il faut suivre". Si la plus grande partie du programme repose sur un système à base de règles figées, il tire également partie d'informations extraites du comportement de l'adversaire et utilise donc une autre forme d'apprentissage (par renforcement). De plus, Waterman utilise aussi une troisième forme d'apprentissage (supervisé) en laissant son programme jouer contre un expert humain pour lui permettre d'ajuster son système de règles. Ces variétés d'apprentissage dans les jeux font l'objet des sections qui suivent.

5.1.2 Apprentissage supervisé

Ce dernier se fonde sur des séries d'exemples pour inférer des caractéristiques permettant de les caractériser par généralisation. Cette méthode suppose qu'il est possible d'évaluer chaque exemple. En fait, quand un exemple est présenté à un système d'apprentissage supervisé, il doit être accompagné de son évaluation. L'objectif de l'apprentissage supervisé est d'être capable, après la phase d’entraînement, de généraliser afin, par exemple, d'identifier, évaluer ou catégoriser automatiquement de nouveaux exemples.

Dans un contexte de jeux, l'estimation fournie par l'expert est la valeur qu'il associe de la récompense future associée à la situation.

Exemple du Tic-Tac-Toe :

Un système d'apprentissage supervisé pour ce jeu serait sans doute construit de la façon suivante :

Un expert du jeu fournirait à un système un certain nombre de configurations du jeu et fournirait une estimation de l'intérêt de chacune d'elle pour le joueur.

Le système devrait alors lier de façon judicieuse les caractéristiques des configurations aux évaluations fournies afin de construire une fonction automatique d'évaluation cohérente avec tous ces exemples.

Par la suite, le système devrait être capable de généralisation, c'est-à-dire d'estimer une configuration non présentée précédemment dans les exemples. Cela lui permettrait d'effectuer un choix approprié entre plusieurs actions possibles à partir de situations réelles de jeu.

5.1.3 Apprentissage par renforcement

L'apprentissage par renforcement se distingue de l'apprentissage supervisé par plusieurs aspects : Il ne nécessite pas de phase d'entraînement, il repose sur le principe d'essai-évaluation (trial-error), et enfin s'appuie sur une estimation anticipée de la récompense.

Un système d'apprentissage par renforcement requiert :

- Un module lui permettant de déterminer les comportements ou actions possibles en fonction de l'environnement (système à base de règles par exemple) ;

- Une fonction de gain associant un gain objectif (reward) à chaque situation réelle ;
- Une fonction d'évaluation associant à chaque situation possible de l'environnement une estimation anticipée de la récompense finale associée à un but fixé (c'est-à-dire, de la récompense globale correspondant à tous les gains de toutes les situations réelles rencontrées jusqu'à la réalisation du but) ;
- Eventuellement, un modèle de l'environnement permettant de simuler les actions pour en estimer les conséquences.

Dans un contexte de jeux, cette forme d'apprentissage est une méthode itérative de recherche de stratégies optimales en fonction d'un but fixé et d'interactions avec l'environnement : les adversaires et les situations de jeu. En temps réel, un système d'apprentissage par renforcement ajuste ses stratégies en fonctions des résultats obtenus.

L'apprentissage par renforcement cherche ainsi à maximiser les gains attendus.

Concrètement, pour chaque coup, le système choisit généralement l’action le conduisant vers la situation dont l’évaluation est la plus forte. L’évaluation de la situation précédente est alors ajustée en fonction de celle de la situation suivante. Un facteur aléatoire d’exploration, impose de temps en temps, de ne pas choisir l’action correspondant à la situation suivante la plus favorable mais d’en choisir aléatoirement une autre. Ce facteur d’exploration permet de découvrir de nouvelles stratégies pouvant s’avérer meilleures que celles précédemment considérées comme telles.

5.1.3.1 Méthodes d’évaluation des actions

L’apprentissage par renforcement utilise plusieurs méthodes pour évaluer l’intérêt de chaque action possible dans une situation donnée. La plus répandue (et la plus simple) est d’associer à chaque action la moyenne des récompenses (rewards) qu’elle a pu recevoir dans le passé. L’évaluation d’une action est aussi appelée valeur sélective.

On définit alors :

$$V_t(a_i) = \frac{\sum_{k=1}^{k_a} r_k}{k_a}$$

Où V_t est la valeur estimée de l'action *a* à l'instant t ; k_a est le nombre de fois que l'action *a* a été choisie et les r_k sont les différentes récompenses reçues précédemment par l'action *a.*

L'intérêt de cette méthode est qu'elle converge vers la valeur objective mathématique. En effet, la loi des grands nombres garantit que si $k_a \to \infty$, V_t converge vers V_t^* la valeur réelle de V_t.

5.1.3.2 Méthodes de sélection des actions

5.1.3.2.1 Méthode du Max

L'évaluation des différentes actions possibles permet d'effectuer un choix parmi ces dernières. La méthode la plus simple est de toujours choisir l'action dont l'évaluation est la plus élevée.

On définit alors a^* la meilleure action possible telle que : $V_t(a^*) = \max_a V_t(a)$.

5.1.3.2.2 Méthode ε-greedy

Dans un contexte de jeu, la « meilleure action » n'est pas toujours celle qui semble être la plus prometteuse, notamment dans les jeux à information incomplète ou imparfaite. Il peut parfois être très astucieux de prendre son adversaire à contre-pied en effectuant un choix différent de celui qui semble être le meilleur : c'est, entre autres, la fonction du bluff. Par ailleurs, il est important, sinon crucial, dans les jeux de toujours explorer les autres possibilités : cela évite de devenir prédictible et permet, le cas échéant, d'exploiter une faiblesse particulière de l'adversaire. C'est dans cette optique que les méthodes de sélection de l'action dans les jeux laissent presque toujours la possibilité à d'autres actions que la meilleure d'être choisies.

La méthode la plus simple pour ce faire est de choisir a* avec une probabilité assez grande et une autre action a $\neq$ a* avec une faible probabilité ε. L'action a correspondante est choisie au hasard parmi les autres actions possibles. Cette méthode est appelée *ε-greedy*. La valeur de ε (en pratique souvent comprise entre 0.01 et 0.15) définit la proportion d'exploration de nouvelles stratégies par rapport aux exploitations des estimations déjà effectuées.

5.1.3.2.3 Méthode Soft-Max

Cette méthode permet d'effectuer un choix aléatoire de l'action avec des probabilités proportionnelles aux évaluations des actions.

Ainsi, comme dans le cas de la méthode *ε-greedy*, il est possible de choisir une action qui n'est pas la meilleure a priori mais la préférence est donnée tout de même aux actions censées être les meilleures (à plus forte valeur sélective).

Cette méthode utilise tout simplement une distribution de Gibbs ou de Boltzmann. Chaque action peut être choisie avec une probabilité :

$$P_{choix}(a_i) = \frac{e^{V_t(a_i)/\tau}}{\sum_{k=1}^{n} e^{V_t(a_k)/\tau}}$$

τ est un paramètre positif appelé *température*. Il définit la forme de la répartition probabiliste du choix des actions en fonction de leur évaluation. Plus la température est élevée, plus les choix des actions possibles sont équiprobables. Plus la température est basse, plus la méthode est élitiste, favorisant les actions dont la valeur sélective est la plus grande.

Comme cela a été dit précédemment, la stratégie optimale pour un jeu à information complète et imparfaite est une répartition probabiliste des stratégies pures. La méthode Soft-Max permet ainsi de choisir chaque action suivant cette répartition optimale sous réserve de choisir correctement la fonction d'évaluation V_t.

5.1.3.3 Apprentissage classique

Le calcul de la répartition probabiliste optimale n'est pas nécessairement immédiat. Il est parfois nécessaire de déterminer cette répartition par l'expérience. Généralement, cette répartition est initialisée de façon uniforme, puis est ajustée en fonction des résultats (victoires, défaites)[1].

Ainsi, on ajuste la probabilité de sélection de l'action a choisie à l'instant t de la façon suivante :

$$P_{choix}(a)_{t+1} = P_{choix}(a)_t + \alpha[1 - P_{choix}(a)_t]$$

---

[1] Nous avons implémenté une telle méthode pour en comparer les résultats avec ceux de la méthode S.A.G.A.C.E.

α dépend du succès ou de l'échec de *a* à l'instant t. Evidemment, les probabilités de choix des autres actions sont ajustées de façon à ce que la somme des probabilités reste égale à 1.

5.1.3.4 Les autres méthodes d'apprentissage par renforcement

5.1.3.4.1 Programmation dynamique

Ce type de programmation a été introduit par Bellman en 1957 [Bellman, 1957].

Dans le cas général, la programmation dynamique suppose que soit disponible un modèle parfait de l'environnement. Dans le contexte des jeux, elle requiert que le jeu soit à information complète.

Pratiquement, la programmation dynamique s'adresse à une classe de problèmes dits *Processus de Décision Markoviens*. Cela signifie que l'état à l'instant t du système considéré est entièrement défini par son état à l'instant t-1. On peut considérer que c'est le cas pour les jeux à information complète et parfaite. Ainsi, même si une position donnée aux échecs ne permet pas de reconstituer tous les coups précédents (leur ordre notamment), cette position suffit à élaborer une stratégie.

Le but de la programmation dynamique est la détermination d'une solution optimale par ajustement d'une fonction d'évaluation dont l'utilisation permet d'organiser et structurer la recherche des actions optimales dans chaque situation.

La limite la plus importante de la programmation dynamique pour les jeux et la même que celle de la Théorie des Jeux : elle peut être inapplicable quand l'arbre de décision est trop grand parce qu'elle nécessite un parcourt exhaustif de cet arbre. Pour remédier à cette contrainte, certaines méthodes de programmation dynamique asynchrone ont été développées. Elles sont censées garantir une convergence itérative asymptotique de l'optimisation des stratégies. Le problème sous-jacent est alors la vitesse de convergence.

Minsky [Minsky, 1961] a, pour la première fois, établi les relations entre l'apprentissage par renforcement et la programmation dynamique lorsqu'il a commenté le travail de Samuel concernant son programme de checkers [Samuel, 1959, 1967].

5.1.3.4.2 Méthode de Monte-Carlo

Cette méthode a été créée dans les années 40 par des physiciens de Los Alamos qui cherchaient une analogie entre les jeux de hasard et les phénomènes physiques relatifs à la bombe atomique !

Elle s'appuie non pas sur une connaissance parfaite de l'environnement mais sur l'expérience. Elle est donc adaptée à des jeux à information imparfaite ou incomplète. Une application de cette méthode par Widrow, Gupta et Maitra a concerné, par exemple, le jeu du Blackjack [Widrow, 1973]. En revanche, cette méthode ne considère aucun modèle dynamique des interactions avec l'environnement (aucun modèle de l'adversaire), ce qui paraît pourtant être essentiel dans un contexte de jeu à coups simultanés (information imparfaite) comme nous le montrerons plus loin.

La méthode de Monte-Carlo est moins restrictive que celle de la programmation dynamique en ce qui concerne l'aspect markovien des problèmes qu'elle peut traiter. Cela est dû au fait qu'elle n'ajuste pas l'estimation de la valeur sélective des actions pour une situation de jeu en fonction de l'évaluation des situations successives possibles. Comme le fait remarquer Sutton [Sutton, 1998], cette méthode n'effectue pas de bootstrap, c'est-à-dire qu'elle ne base pas une évaluation sur d'autres évaluations.

5.1.3.4.3 Apprentissage par différence temporelle

L'apprentissage par différence temporelle (TD-Learning) est une combinaison des deux précédentes méthodes. Elle évalue l'intérêt des actions possibles par expérience et ne nécessite pas de modèle de la dynamique de l'environnement. De plus, si elle utilise une forme de bootstrap en capitalisant sur les évaluations des positions successives correspondant à chaque action choisie, elles ne nécessitent toutefois pas un parcours exhaustif de l'arbre de décision (notamment des positions terminales).

L'apprentissage par différence temporelle est incrémental. Les estimations sont faites étape après étape (coup après coup dans un contexte de jeu), et non uniquement en fin d'épisode (fin de partie dans un contexte de jeu).

La plus célèbre application du TD-Learning dans un contexte de jeu est sans aucun doute le programme TD-Gammon de Tesauro [Tesauro, 1994, 1995]. Initialisé avec une expertise élémentaire du jeu, ce programme a atteint le niveau des champions humains. La méthode de TD-Learning utilisée dans ce programme repose sur une méthode d'approximation non linéaire mettant en œuvre un réseau de neurones entraîné par rétropropagation d'erreurs.

Ce programme a été inspiré d'une précédente application : celle de Samuel pour son jeu de checkers [Samuel, 1959, 1967]. A l'époque de cette application, le terme de TD-Learning n'était pas encore utilisé. Pourtant, la méthode de recherche heuristique utilisée par Samuel correspond tout à fait à l'acception du terme actuel. Ce programme s'inspirait également de la méthode du minimax de la théorie des jeux. Le parcours de l'arbre de décision ne pouvant être exhaustif en début de partie, le programme avait recours à des méthodes alpha/beta. Enfin, une variété d'apprentissage par cœur était également utilisée : certaines situations de jeu étaient enregistrées avec le meilleur coup calculé mathématiquement.

La fonction d'évaluation utilisée par Samuel était de la forme suivante :

$$\alpha_1 C_1 + \alpha_2 C_2 + \ldots + \alpha_n C_n$$

Les α_i étaient des coefficients (positifs ou négatifs) qui étaient ajustés par apprentissage, les C_i étaient des caractéristiques de la position du jeu. Au cours de l'apprentissage, la fonction modifiait les α_i. Ces derniers diminuaient si la caractéristique correspondante s'avérait être moins souhaitable et augmentaient dans le cas contraire. Les caractéristiques (au nombre de 30 environ) étaient, par exemple, le nombre de pions de l'adversaire, le nombre de pions alignés en diagonale, etc. Il est important de noter que les caractéristiques étaient fixes et qu'elles avaient été définies par Samuel lui-même. Or, le plus difficile dans la réalisation de tels programmes est sans aucun doute de définir les caractéristiques.

Aussi bien Tesauro que Samuel ont également eu recours à une méthode d'apprentissage originale : le bootstrap. Le programme apprenait à jouer contre une copie de lui-même. Cette méthode permet d'accélérer l'apprentissage parce qu'elle ne nécessite pas l'expertise d'un joueur humain intervenant par apprentissage supervisé. Les programmes sont capables d'apprendre seuls. Comme nous le verrons par la suite, nous nous sommes inspiré de cette idée pour le développement de la méthode S.A.G.A.C.E. Il faut toutefois noter que cette méthode peut être dangereuse : elle peut conduire à une dérive dans la mesure où les deux versions du même programme peuvent se spécialiser dans une stratégie commune non adaptée à d'autres adversaires. Samuel a constaté lui-même des baisses de performances de son programme après de trop grandes séries de parties contre lui-même. C'est un problème qui se pose souvent lorsqu'un programme considère systématiquement que ses adversaires possèdent les mêmes fonctions d'évaluation que lui alors que ces fonctions ne sont pas optimales. Dans le cas du programme de Samuel, les deux copies du programme étaient initialisées avec les mêmes caractéristiques mais avec des coefficients différents. Après de trop longues séries de parties, les fonctions d'évaluation des deux copies devenaient équivalentes (les coefficients égaux deux à deux).

5.1.3.4.4 Apprentissage Q-Learning

Le Q-Learning [Watkins, 89 & 92] [Sutton, 98] suppose une formulation des problèmes sous une forme markovienne (où un état du système contient toutes les informations pertinentes, i.e. la connaissance des états précédents n'est d'aucune utilité pour définir l'état courrant). Dans cette optique, un agent est défini comme un ensemble discret de tous ses états possibles (s) et de toutes ses actions possibles (a) dans chacun de ses états. Pour chaque paire (s,a), il existe un signal de renforcement R(s,a) associé au passage de l'état s à l'état s' quand l'agent a effectué l'action a.

Le Q-Learning est fondé sur une approche itérative de la résolution des équations de Bellman :

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} P_{(s,a)}(s')V(s') \qquad V(s) = \max_a (Q(s,a))$$

Ces équations définissent une fonction Q(s,a) et une fonction d'évaluation V(s). La fonction Q représente un gain attendu qui est la somme des renforcements reçus en accomplissant l'action a dans l'état s. La fonction V est une fonction d'évaluation des états qui est la somme des renforcements potentiels (c'est-à-dire présents et à venir) si l'agent adopte toujours le comportement optimal. Le coefficient γ (appelé « discount factor ») compris entre 0 et 1 sert à évaluer les récompenses retardées dans le temps et assure la convergence des fonctions.

Le Q-learning permet de résoudre itérativement ces équations par estimation successives de la fonction Q.

$$Q(s,a) = \alpha Q(s,a) + (1-\alpha)(R(s,a) + \gamma V(s'))$$

Le coefficient α représente le taux d'apprentissage.

Remarque :

L'algorithme du Bucket Brigade[1] de Holland [Holland, 1986] est une forme d'apprentissage très proche du Q-Learning dans la mesure où le renforcement est « propagé à retardement » d'état en état.

Littman a proposé un algorithme appelé Minimax Q-Learning [Litman, 1994] pour les jeux markoviens. Plutôt que de considérer l'adversaire comme un simple élément de l'environnement, Littman a intégré ses actions (ces choix de jeu) à la fonction Q, qui prend alors la forme Q(s,a_m,a_o) où a_m est l'action de l'agent et a_o l'action de son adversaire (opponent). Pour chaque état est alors enregistrée une estimation de Q sous la forme d'un jeu matriciel. L'agent minimise ses pertes potentielles en fonction de toutes les actions possibles de l'adversaire, comme le suggère la Théorie des jeux.

Littman a appliqué avec un certain succés cet algorithme à un jeu à information incomplète [Littman, 1994] (un « soccer » dans lequel chaque joueur essai d'emmener le ballon dans le but de l'adversaire, sans se le faire dérober).

Littman a suggéré l'utilisation de Minimax Q-Learning pour les jeux tels que les checkers, le backgammon, le tic-tac-toe et le Go.

---

[1] Cet algorithme est décrit dans le chapitre suivant.

5.1.4 Apprentissage par découverte

Il s'agit de méthodes de bootstrap à base d'heuristiques. De nouvelles heuristiques sont découvertes à partir d'anciennes et ce de façon itérative. Lenat [Lenat, 1984] définit une heuristique comme "une pièce de connaissance susceptible de suggérer de bonnes idées à suivre et de mauvaises à éviter". Définie ainsi, une heuristique doit permettre de spécifier, suivant le contexte (ou la situation pour les jeux), l'action la plus appropriée.

Cette forme d'apprentissage permet la découverte de nouveaux concepts. Par exemple, une heuristique du système AM de Lenat était la suivante :

$$\text{IF } f{:}A\text{x}A \rightarrow B, \text{ THEN define } g{:}A \rightarrow B \text{ as } g(x)=f(x,x).$$

En mathématiques (démonstration de théorèmes), une telle heuristique permet la découverte des concepts de double (si f est la somme), de carré (si f est la multiplication) d'identité (si f est l'union ou l'intersection), de zéro (si f est la soustraction), etc.

En ce qui concerne les jeux, Lenat a poursuivi ses recherches dans la voie tracée lors de la conception de AM et a conçu, en 1980, le programme Eurisko [Lenat, 1983, 1984] qui est resté célèbre pour avoir remporté deux années consécutives (1981 et 1982) le championnat de TCS (Traveller Trillion Credit Squadron), un jeu de simulation de bataille navale. Eurisko découvrait de nouvelles heuristiques à partir de méta-heuristiques.

Parmi les heuristiques qu'Eurisko avait découvertes, s'en trouvait une qui a changé le cours des batailles navales simulées : "Lors de la construction de la flotte, la bonne stratégie est de chercher une solution proche des solutions extrêmes". En 1981, Eurisko avait imaginé une flotte constituée de petites pirogues très peu armées mais très nombreuses et très rapides (donc difficile à détruire). Il gagna le tournoi et les règles en furent changées, passant de 100 à 200 pages ! En 1982, Eurisko avait imaginé une flotte très différente : très peu de bateaux mais très puissamment armés. Ces navires étaient très lents, mais tellement puissants, qu' Eurisko gagna encore. Eurisko avait, par ailleurs, trouvé des failles dans les règles du jeu et avait su les exploiter, contrairement aux joueurs humains.

Depuis de nombreuses années, Lenat travaille au développement de CYC, un programme "nourri" avec une somme immense de connaissances dans tous les domaines scientifiques connus sous un formalisme unifié. Peut-être que CYC sera un champion universel de tous les jeux connus malgré le sentiment général de la communauté scientifique qui est plus que réservé sur les chances de réussite de CYC et sur la méthodologie dont il est issu.

5.2 L'anticipation dans les jeux

Dans les rares cas où la Théorie des jeux apporte non seulement la solution optimale mais encore une solution gagnante, son application s'impose. C'est par exemple le cas pour le jeu du Tic-tac-toe où un joueur qui applique la théorie ne peut pas perdre. Au pire des cas, il obtient le match nul et dans le cas où son adversaire ne joue pas parfaitement, la théorie garantit la victoire.

Cependant, comme nous l'avons rapellé à plusieurs reprises, de tels jeux sont rares.

Par ailleurs, dans les cas des jeux à information complète et imparfaite, si la théorie garantit un gain optimal, elle ne garantit rien de plus. Elle ne permet par exemple pas de tirer profit d'une stratégie défaillante de l'adversaire.

Borel exprime le même point de vue : "Le joueur qui n'observe pas la psychologie de son partenaire et ne modifie pas en conséquence sa manière de jouer doit forcément perdre vis-à-vis d'un adversaire dont l'esprit est assez souple pour varier son jeu en tenant compte de celui de l'adversaire" [Borel, 1938].

L'hypothèse fondatrice de la théorie est la rationalité des joueurs. Or, quand certains joueurs sont humains, on peut douter du caractère rationnel de leurs prises de décision parce qu'aucun joueur humain ne peut prétendre être parfaitement rationnel, surtout quand le jeu est complexe.

Imaginons au jeu de Pierre/Ciseaux/Papier un joueur ayant une propension à jouer plus souvent *pierre* que *ciseaux* ou *papier*. Si son adversaire applique parfaitement la théorie, il jouera aléatoirement, et de façon uniforme, *pierre*, *ciseaux* ou *papier*. Il est pourtant évident qu'il aurait tout intérêt à jouer surtout *papier*.

Par ailleurs, un humain est-il capable d'appliquer parfaitement la théorie ? Notre conviction est qu'il est pratiquement impossible à un humain de le faire quand l'application de la théorie requiert l'utilisation du hasard. Nous reviendrons ultérieurement sur ce problème de l'aléatoire pour un joueur humain.

L'étude de l'anticipation dans les jeux peut être conduite suivant trois directions : coopérative, passive ou compétitive. Dans le modèle coopératif, il s'agit d'anticiper le comportement de ses partenaires pour rendre l'association plus performante. Dans le modèle passif, on suppose que l'anticipé n'aide, ni ne tente de gêner, l'anticipant ; c'est le cas de la plupart des réalisations. Enfin, le modèle compétitif suppose que le modélisé fait tout son possible pour demeurer le moins prévisible possible. Ce dernier modèle concerne particulièrement la théorie des jeux, ou bien la vision pessimiste selon laquelle la nature (l'environnement) est toujours plus ou moins hostile.

Ces trois directions de l'étude de l'anticipation sont les point-clés de la théorie des jeux. Nous y reviendrons par la suite.

5.2.1 Le modèle coopératif :

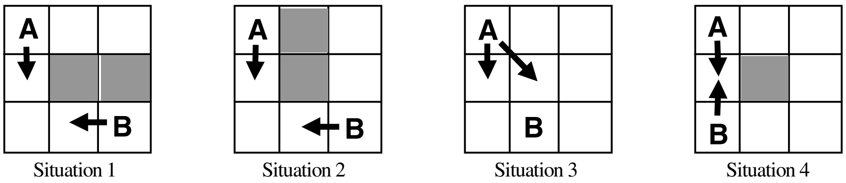


Figure 5.1. Modèle coopératif.

On suppose que les agents A et B ont pour but de se rencontrer sur une même case. Ils ne peuvent pas communiquer (ou, ne le peuvent plus à la suite d'un incident). Ils se déplacent dans toutes les directions mais ne peuvent franchir les cases noires. Le but pour eux est de minimiser le nombre de déplacements avant de se rencontrer. Pour ce faire, ces agents sont amenés à prendre des décisions en anticipant sur celles de leur partenaire.

Si on se place vis-à-vis de A, la situation 1 (Figure 5.1) est la plus facile à gérer. En effet, sans avoir à conjecturer sur l'attitude de B, il sait, en considérant uniquement les contraintes de l'environnement, que B ne peut qu'aller à gauche (ou, au pire, rester bloqué). De ce fait, A va descendre, anticipant facilement sur le fait que B va aller à gauche.

La situation 2 (Figure 5.1), vis-à-vis de A, est un peu différente. En effet, B a la possibilité matérielle de monter et, de ce fait, d'agir de façon irrationnelle (c'est-à-dire en choisissant une trajectoire inadaptée à son but). A doit anticiper en tenant compte du fait que B et lui-même sont dans un état coopératif. Ainsi, il paraît logique pour A de supposer que B, cherchant à optimiser le 'rendez-vous' va aller à gauche. Au pire, si B est défaillant et choisit de monter, A n'a toujours qu'une possibilité pour le retrouver, c'est de le poursuivre. Ainsi, l'anticipation de A est encore aisée, il va descendre. (Le fait que A n'ait, matériellement pas d'autre choix, est une raison supplémentaire de sa descente, cela peut très bien lui éviter d'avoir à anticiper sur les actions de B...).

Dans la situation 3 (Figure 5.1), le problème de l'anticipation est clairement posé. A n'a, a priori, aucun moyen matériel ou planifié de déterminer l'action de B. Il va donc devoir anticiper sur le comportement de celui-ci avant de prendre sa décision. Les performances du système ne dépendent, dans cet exemple, que de la qualité de l'anticipation. Une anticipation par apprentissage consisterait, dans ce cas, à rechercher dans les situations précédentes une configuration semblable et à agir en conséquence. A supposer que le rendez-vous soit un échec, il se peut que la résolution du problème soit encore longue. En effet, A et B peuvent choisir ensuite de ne plus bouger ou de bouger simultanément dans des directions inadéquates, ce qui ne résoudrait rien.

La situation 4 (Figure 5.1) est la plus complexe. Dans le cas ou tout se passe 'logiquement', A devrait descendre et B, monter ce qui résout le problème directement. Supposons que B soit défaillant et qu'il parte sur la droite ; si A n'a pas des procédures (techniques) robustes et adaptatives, il est probable qu'il va poursuivre B. Si leur vitesse est la même, le rendez-vous sera toujours un échec. Par contre, si A est anticipant et adaptatif, il va pouvoir analyser différemment la situation. On peut espérer qu'il se rendra rapidement compte de la défaillance de B et de ses conséquences. Ainsi, il pourra décider de partir dans l'autre sens, comprenant qu'alors il ne pourra plus rater B (excepté si B est tellement défaillant que son but soit devenu celui de fuir A).

5.2.2 Le modèle passif :

Il s'agit de l'approche la plus simple de l'anticipation qui ne nécessite pas de prendre en compte les conséquences de ses propres actions sur le comportement d'autrui. Le problème se résume alors uniquement à effectuer des prédictions et à agir en conséquence, sans avoir à gérer une quelconque influence de son propre comportement sur l'objet des prédictions.

5.2.3 Le modèle compétitif :

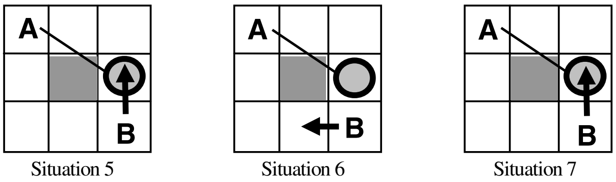


Figure 5.2. Modèle compétitif.

On suppose, dans ce modèle, que A tente de détruire B. Pour ce faire, il doit anticiper sur le déplacement de celui-ci en détruisant la case correspondant à la prochaine position de B.

A la différence du modèle coopératif, une réussite de A signifie un échec de B. De ce fait, B, s'il est adaptatif, va tirer un enseignement des réussites de A. Supposons que A anticipe correctement et gagne (Situation 5 ; Figure 5.2). Dans un programme d'IA classique, cela va le conforter dans l'idée qu'il faut, dans cette configuration, tirer au-dessus de B. Or, pour B, le résultat est inverse, un programme classique adaptatif va le pousser à choisir l'autre direction la prochaine fois (Situation 6 ; Figure 5.2). Alors, A va perdre et va devoir modifier l'idée qu'il a de B et affiner ses anticipations (Situation 7 ; Figure 5.2). Toujours est-il qu'il va avoir des performances bien moindres que dans le modèle coopératif. Si A n'est pas suffisamment adaptatif, il va 'rater' B une fois sur deux, car à chaque fois ce dernier va modifier sa direction.

La seule solution pour A est de posséder des techniques d'anticipation adaptatives et performantes. Il faudrait qu'il prenne en compte les conséquences de ses anticipations réussies sur le comportement de B, l'anticipé.

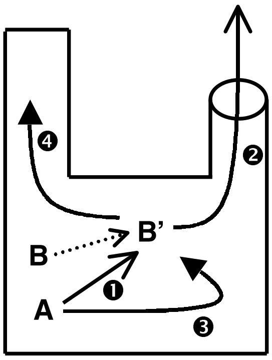
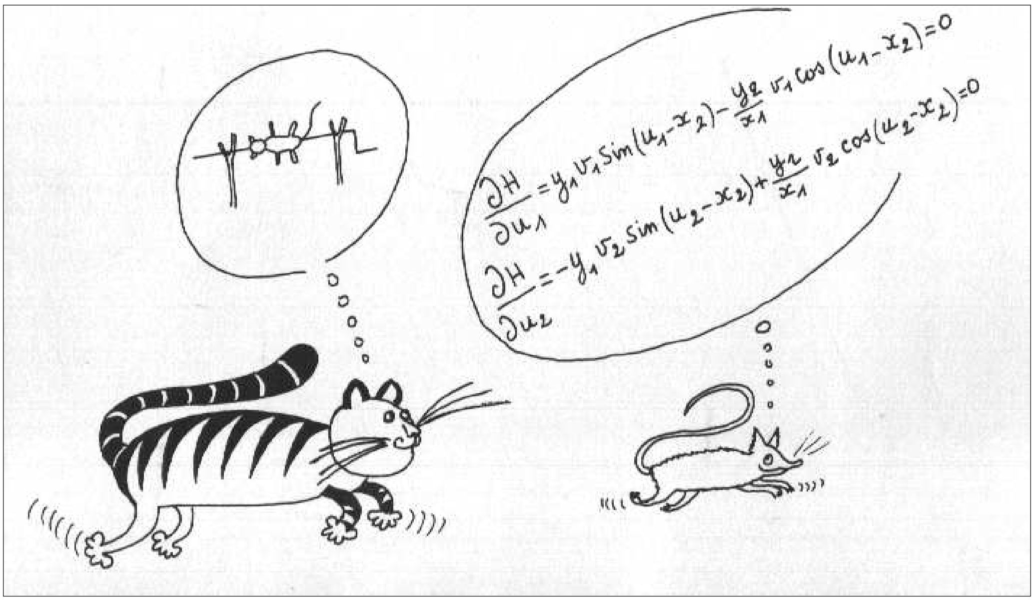


Figure 5.3. Modèle compétitif - le couloir -

Dans cet exemple d'anticipation raisonnée (Figure 5.3), on suppose que A cherche à attraper B en le bloquant au bout du couloir de gauche, qu'il se déplace plus vite que lui et qu'il l'effraie. On suppose également que le couloir de gauche est fermé et qu'une sortie de B par la droite signifie une victoire pour ce dernier. On suppose enfin, que, au départ de l'action, B se dirige vers la droite (flèche en pointillés sur la figure).

Si A fait une anticipation simple sur le déplacement de B, il va se diriger vers le haut à droite (❶) c'est-à-dire, à un point de rendez-vous calculé. Or, ce faisant, il va effrayer B qui va se diriger vers le couloir de droite (❷) et va donc gagner. Une anticipation efficace nécessite la prise en compte de sa propre influence. Ainsi, A doit tenir compte du fait que son propre déplacement anticipé va modifier celui de B. La meilleure anticipation va donc consister à partir à droite dans un premier temps puis vers le haut à gauche (❸) afin de rabattre B dans le couloir de gauche (❹).



©JAMAG, 80

5.2.4 Modélisation de l’adversaire

La majorité des systèmes capables de jouer correctement à des jeux à information complète et imparfaite, ou à information incomplète, intègrent, d’une façon ou d’une autre, des processus de modélisation de leurs adversaires afin d’anticiper leurs stratégies.

5.2.4.1 Qu’est-ce qu’un modèle de l’adversaire

Il s’agit d’une fonction ou d’un algorithme permettant d’estimer (avec une précision ou une confiance donnée) la stratégie de son adversaire en fonction d’un état du jeu, c’est-à-dire les choix qu’il effectuerait dans chaque situation.

Typiquement, l’algorithme du MinMax intègre de façon implicite un modèle simpliste de l’adversaire : « l’adversaire joue exactement avec la même stratégie que le joueur » (sous-entendu : « il a les mêmes connaissances, la même analyse du jeu et tire les mêmes conclusion »).

5.2.4.2 A quoi sert un modèle de l’adversaire

Janson [Janson, 1990] a défini deux situations dans les jeux où il est très important de considérer les capacités de jeu de l’adversaire. La première, qu’il appelle « swindle » (escroquerie), concerne une situation dans laquelle le joueur a des raisons de penser que son adversaire va sous-estimer certains coups et va donc jouer, à leur place, un coup médiocre. La seconde, qu’il nomme « trap » (piège), est une situation dans laquelle le joueur pense que son adversaire va surestimer un coup et va donc le jouer au détriment d’autres coups meilleurs. Avec ces définitions, il est subtil d’orienter sa stratégie de telle façon qu’elle conduise à de telles situation. Pour ce faire, il faut, bien évidemment, posséder un modèle de l’adversaire.

Une autre situation où il est utile de posséder un modèle de l’adversaire est le cas où la plus grande valeur renvoyée par la fonction d’évaluation concerne plusieurs coups possibles. Dans ce cas, les algorithmes sans modèle choisissent aléatoirement un coup. Avec un modèle de l’adversaire, il est possible de choisir le coup censé poser le plus de problème à l’adversaire, même si ce coup conduit à la même évaluation (sur les critères d’une fonction d’évaluation classique) que les autres.

5.2.4.3 Comment construire un modèle de l'adversaire

Les quelques systèmes présentés dans la section suivante donnent des exemples de la façon dont on peut construire un tel modèle.

Généralement, un joueur construit un modèle de son adversaire en se basant sur des coups observés de ce dernier (en cours de partie contre lui ou en cours de parties contre d'autres adversaires).

5.2.5 Quelques systèmes utilisant un modèle de l'adversaire

5.2.5.1 Jeux à information complète et parfaite

L'idée d'intégrer un modèle a principalement concerné les jeux à information complète et parfaite[1]. Carmel et Markovitch, convaincus par les différents travaux (dont [Berliner, 1977] ; [Knuth, 1975] ; [Korf, 1989] ; [Schaeffer, 1992] et [Samuel, 1967]) confirmant l'importance de l'utilisation d'un modèle de l'adversaire pour la réalisation de programmes de jeux, ont constaté que cela n'avait pas eu d'influence dans les réalisations effectives. Ils ont développé en 1993 un algorithme (M*) [Carmel, 1993] permettant d'intégrer un modèle de l'adversaire dans une procédure de recherche de type MinMax. L'algorithme M* se définit comme une généralisation de MinMax dans laquelle les deux joueurs ne sont pas supposés utiliser la même stratégie.

Ils ont conduit des expériences dans lesquelles ils ont comparé les performances de M* et de l'algorithme classique MinMax. Par ailleurs, ils ont effectué des expériences d'auto-modélisation et ils ont testé des algorithmes qui utilisent les coups de l'adversaire pour estimer sa profondeur de recherche et sa fonction d'évaluation.

M* pose comme hypothèses que l'adversaire effectue ses choix stratégiques à l'aide d'un algorithme MinMax simple, qu'il évalue les position à l'aide d'une fonction d'évaluation constante, que sa profondeur de recherche est également constante (il n'utilise aucun algorithme d'optimisation de profondeur de recherche en fonction des branches) et enfin qu'il n'utilise, lui, aucun modèle de son adversaire (de M* donc).

---

[1] Sans doute parce qu'il s'agit des jeux les plus étudiés et les plus faciles à appréhender. Le modèle, pour ces jeux, est utile lorsqu'il n'est pas possible d'exploiter la théorie (comme pour le Go) parce que la combinatoire est trop importante ou lorsqu'on pense pouvoir exploiter d'éventuelles faiblesses de l'adversaire.

L'algorithme fonctionne comme un MinMax mais il utilise deux fonctions d'évaluation (f_{min} et f_{max}) et deux profondeurs de recherche (d_{min} et d_{max}). f_{min} et d_{min} représentent le modèle actuel de l'adversaire (ce que le programme estime être sa fonction d'évaluation et sa profondeur de recherche) et f_{max} et d_{max} sont des paramètres de l'algorithme et définissent la stratégie du programme lui même.

M* attribue à chaque nœud deux valeurs (v_{min} et v_{max}) calculées respectivement à l'aide de (f_{min} , d_{min}) et (f_{max} , d_{max}).

Korf a proposé en 89 [Korf, 1989] un algorithme très similaire avec presque les mêmes hypothèses. La principale différence résidait dans la prise en compte du fait que l'adversaire possède également un modèle du joueur, dont le joueur possède un modèle, dont l'adversaire… Cette récursivité infinie des modèles avait des conséquences lourdes dans les calculs sans apporter de gain de performance par rapport à M* développé plus tard.

Les hypothèses conditionnant la validité de ces algorithmes sont fortes. C'est en particulier le cas de l'hypothèse que la fonction d'évaluation est constante qui suppose donc que l'adversaire n'est pas capable d'apprendre. Pour cette seule raison, l'algorithme semble inadapté contre des joueurs humains. L'hypothèse de non-utilisation d'un modèle de l'adversaire (pour M*) est également forte et incompatible avec l'application de la méthode contre des adversaire humains.

Pour construire le modèle de l'adversaire, M* utilise deux méthodes : la première pour estimer la profondeur de recherche de l'adversaire et la seconde pour estimer sa fonction d'évaluation.

Les deux algorithmes correspondants s'appuient sur des exemples de parties jouées par l'adversaire. L'espace de recherche pour la profondeur est petit et, si on considére que la fonction d'évaluation de l'adversaire est peu différente de celle du joueur, il est aisé d'estimer la profondeur de recherche d'un adversaire. Une erreur importante sur le modèle de la fonction d'évaluation de l'adversaire peut, bien entendu, induire une erreur dans l'estimation de la profondeur de recherche, mais la méthode est tout de même relativement satisfaisante.

L'algorithme d'estimation de la fonction d'évaluation est très semblable aux algorithmes utilisés pour Deep Blue [Hsu, 1990] ou pour Chinook [Schaeffer, 1992] afin de déterminer une bonne fonction d'évaluation à partir de bases d'exemples de parties d'échecs et de checkers jouées par des experts. Il y a toutefois une différence entre les deux approches liée au fait que, pour M*, la fonction recherchée ne sera pas utilisée par le programme lui même mais comme modèle de l'adversaire qui a joué les parties "exemples". Quoiqu'il en soit, la mise en œuvre de l'algorithme d'estimation nécessite que soit connu l'ensemble des paramètres qui peuvent être pris en compte par un joueur lorsqu'il prend une décision. La fonction d'évaluation recherchée est alors supposée être de la forme :

$$f(b) = \overline{w}.\overline{h}(b) = \sum_i w_i h_i(b)$$

où b est l'état du jeu (board) et $h_i(b)$ le $i^{ème}$ critère de cet état. Un critère est un aspect caractéristique d'un état du jeu pouvant influencer la prise de décision d'un joueur. C'est, par exemple, le cas du nombre de pions restant pour le jeu de Checkers, la liberté du roi aux échecs, ou le nombre d'alignements encore possibles pour le joueur à connect4 (puissance4)…

Ini encore, ces approches reposent sur des hypothèses fortes incompatibles, à notre sens, avec les stratégies développées par des joueurs humains.

Cela étant dit, M* permet de produire un modèle très efficace d'adversaires utilisant effectivement des stratégies de type MinMax et des fonctions d'évaluation de cette forme. M* se montre très efficace contre de tels joueurs et s'avère meilleur que MinMax.

M* a été testé contre des adversaires déterministes pour des jeux à information complète et parfaite. Carmel et Markovitch reconnaissent que les hypothèses faites sur l'adversaire (utilisation d'un algorithme MinMax à fonction d'évaluation et profondeur constantes) sont très restrictives. Les différentes améliorations de l'algorithme initial[1] n'ont pas apporté de réponses adéquates aux jeux à information complète et imparfaite ou aux jeux contre des joueurs humains.

---

[1] En 1994, Carmel et Markovitch ont élaboré un nouvel algorithme M_ε^* [Carmel & Markovitch, 1994] capable de prendre en compte un modèle récurent (l'adversaire peut avoir également un modèle du joueur et le joueur un modèle de ce modèle et…) et pouvant être imparfait (à ε près).

Samuel (en 1967) puis Schaeffer (en 1992) ont également proposé l'utilisation d'un modèle de l'adversaire pour un jeu à information complète et parfaite : le jeu de checkers [Samuel, 1967] et [Schaeffer, 1992]. Ils ont développé des arguments équivalents à ceux présentés par Janson [Janson, 1990] décrits dans le paragraphe §5.2.4.2..

Collins, Birnbaum, Krulwich et Freed ont proposé une utilisation de la modélisation pour le jeu des Echecs mais il s'agissait plus d'auto-modélisation que d'un modèle de l'adversaire [Collins et al., 1993].

5.2.5.2 Jeux à information complète et imparfaite

Shannon a développé un algorithme d'anticipation du comportement d'un joueur humain pour le jeu Pair/Impair (les règles en sont décrites dans le chapitre 7) [Shannon, 1953].

Cet algorithme est suffisamment simple pour avoir été intégré dans une machine totalement mécanique. Il est décrit en détails dans le chapitre des expérimentations (chapitre 8) parce qu'il a été implémenté pour être comparé à S.A.G.A.C.E. contre des joueurs humains.

Bien que l'algorithme de Shannon soit extrêmement simple et qu'une martingale particulière (déterminée par Shannon lui-même) permette de le battre dans un rapport de 3 :1, la machine bat, en moyenne, les joueurs humain de façon remarquable. Sur environ 10.000 (!!) coups joués contre des joueurs humains, la machine de Shannon en a remporté environ 53%[1].

Minasi a écrit un court programme pour jouer contre des joueurs humains au jeu de Pierre/Ciseau/Papier. Le modèle de l'adversaire développé par cet algorithme est simpliste mais n'empêche pas le programme de se révéler meilleur que les joueurs humains sur des parties de longueur supérieure à 20 coups.

L'algorithme correspondant est décrit en détail dans le chapitre des expérimentations (pour les mêmes raisons que celui de Shannon).

---

[1] Une machine purement aléatoire aurait obtenu ce résultat avec une probabilité infime, inférieure à 1 sur 10 billions.

Parmi les jeux de ce type, mais à somme non nulle, on compte notamment le Dilemme itéré des prisonniers. La majorité des programmes ayant été écrits pour ce jeu ont seulement été opposés les uns aux autres, presque jamais à des joueurs humains. Cependant, la plupart de ces programmes prennent explicitement en compte la façon de jouer de l'adversaire et ajustent leur stratégie en conséquence. En revanche, ils ne créent pas, à proprement parler, un modèle de l'adversaire pour anticiper ses coups. Ils se « contentent » d'intégrer le comportement de l'adversaire lors des coups précédents pour choisir le coup suivant. La différence est fondamentale. Par exemple, Axelrod [Axelrod, 1984] a proposé l'utilisation d'un algorithme génétique pour la réalisation d'un système capable de générer des statégies à ce jeu. Dans cet algorithme, les solutions sont implémentées sous la forme de chaînes de 64 bits codant les trois précédentes confrontations. Les résultats obtenus sont bons dans la mesure où, en moyenne, les solutions proposées sont aussi performantes que Tit-for-tat (une très bonne stratégie consistant uniquement à reproduire le coup précédent de l'adversaire : Tit-for-tat se traduit par « œil pour œil »).

De nombreuses études ont concerné ce jeu. On citera notamment [Rapoport, 1965], [Delahaye, 1996a,1998] et [Knoblauch, 1994].

L'approche la plus répandue consiste à supposer que l'adversaire détermine sa stratégie par l'intermédiaire d'un automate à états finis [Hopcroft, 1979]. Il s'agit d'un formalisme particulier permettant d'exprimer que l'adversaire exhibe une rationalité limité [Simon, 1947]. Sous cette hypothèse, un certain nombre de théorèmes [Gilboa, 1989], [Fortnow, 1994] permettent de déterminer une stratégie optimale contre de tels adversaires. Les stratégies correspondantes sont construites récursivement (en temps polynomial). L'hypothèse est cependant forte et ne s'applique sans doute pas à des joueurs humains. En effet, la capacité d'« obéir » à un automate à état fini demande une mémoire proportionnelle au nombre d'états et au nombre de relations entre états. Il semble plus raisonnable de supposer qu'un joueur humain applique des heuristiques qu'il évalue en permanence, plutôt que de supposer qu'il « obéit » à un automate déterministe.

Les mêmes études [Gilboa, 1989], [Fortnow, 1994] ont d'ailleurs conclu que, pour la plupart des jeux à information imparfaite, si un joueur agit suivant un automate probabiliste[1] particulier, il n'existe pas de stratégie optimale rationnelle à lui opposer.

---

[1] Les transitions d'un état à un autre sont, au moins en partie, déterminées par un choix probabiliste.

5.2.5.3 Jeux à information incomplète

C'est pour la réalisation de programmes capables de jouer à de tels jeux que la modélisation de l'adversaire pour l'anticipation de ses coups a été la plus utilisée.

Uther et Veloso [Uther, 1997] ont proposé une amélioration de l'algorithme de Littman [Littman, 1994] présenté dans le paragraphe §5.1.3.4.4.. Leur algorithme OM Q-Learning (pour Opponent Modelling Q-Learning) reprend l'idée d'associer à la fonction Q les actions de l'adversaire, mais remplace la stratégie Minimax (inspirée de la Théorie des jeux) par une stratégie moins pessimiste[1]. La stratégie retenue est celle définie par « le jeu fictif » d'Owen [Owen, 1995] : les choix passés de l'adversaire dans chaque état du système sont enregistrés pour calculer la distribution probabiliste de ses actions. En fonction de cette distribution et des renforcements liés à chaque paire de choix (a_m, a_o) , il est possible de calculer les espérances de gain des différentes actions a_m et de choisir celle qui correspond à la plus grande.

S.A.G.A.C.E. s'oppose en partie à cette méthode qui conduit à une stratégie déterministe (elle permet de découvrir - finalement - les coefficients probabilistes de la stratégie optimale mais se construit de façon déterministe). Il est donc possible pour un adversaire « sagace » de connaître systématiquement le prochain coup d'un tel adversaire.

Uther et Veloso ont défini les règles d'un jeu de soccer (légèrement différent de celui de Littman), le Hexcer. Pour un tel jeu, le OM Q-learning s'avère plus efficace que le Q-Learning (mais avec un apprentissage plus lent) et encore plus efficace que Minimax Q-Learning (et avec un apprentissage plus rapide).

OM Q-Learning pallie deux défauts de MiniMax Q-Learning. Premièrement, contrairement à ce dernier, OM Q-Learning utilise un taux d'apprentissage constant au lieu de dégressif ce qui, si cela lui retire la garantie de convergence à l'infinie, lui permet de s'adapter en permanence à un adversaire qui apprend. Deuxièmement, OM Q-Learning ne nécessite pas (contrairement à Minimax Q-Learning) que l'adversaire ait déjà essayé toutes ses actions possibles dans un état donné avant que l'algorithme n'y soit exploitable [Uther, 1997].

---

[1] La méthode Minimax utilisée par Littman suppose que l'adversaire adopte sa stratégie optimale. Uther et Veloso suggèrent d'utiliser un modèle de l'adversaire pour tirer éventuellement parti de ses faiblesses.

Uther et Veloso ont comparé OM Q-Learning à l'algorithme Prioritized Sweeping de Moore et Atkeson [Moore, 1993]. Cet algorithme effectue deux phases d'ajustement des valeurs de Q par étape de renforcement. La première phase est classique, la seconde consiste à modifier les valeurs de Q correspondant aux états prédécesseurs et successeurs des états les plus souvent mis à jour (qui sont donc des états souvent atteints). Le nombre d'ajustements faits lors de la deuxième phase est un paramètre de l'algorithme qui peut varier en fonction du jeu ou de la durée de l'apprentissage.

Prioritized Sweeping s'avère meilleur que Q-Learning, MiniMax Q-Learning et OM Q-Learning pour le jeu Hexcer [Uther, 1997].

Les algorithmes mis en jeu dans ces expérimentations ont tous un défaut intrinsèque, celui de ne pas intégrer un processus de généralisation. Uther et Veloso ont donc réalisé un nouvel algorithme « continuous U tree » basé sur l'approche « U tree » de McCallum [McCallum, 1995] qui utilise des arbres de décision [Quinlan, 1986] pour la généralisation. La modification majeure de l'approche consiste à généraliser « U tree » pour qu'il puisse s'appliquer à des espaces à états continus (et non discrets). Cet algorithme se révèle plus performant que les précédents [Uther, 1997].

Tous ces algorithmes présentent toutefois un défaut majeur : ils nécessitent un grand nombre d'exemples afin d'être exploitables (pour que l'apprentissage soit cohérent). Uther et Veloso ont alors eu l'idée d'utiliser, dans le cas où l'adversaire est meilleur que le programme, les coups de ce dernier pour « bootstrapper » le système. En clair, il s'agit d'utiliser les données correspondants aux coups d'un adversaire performant pour accélérer l'apprentissage. Si cette technique s'avère effectivement efficace en face d'un adversaire de bon niveau, elle peut se révéler dangereuse lorsque l'adversaire est adaptatif (quand il apprend également). Cela est dû, comme nous l'avons signalé précédemment, au fait que cette forme d'anticipation (par modélisation simple de l'adversaire) ne tient pas compte de ses conséquences sur l'adversaire lui même. Il faut en effet se donner les moyens d'utiliser les données liées à ces observations sans « tromper » son propre modèle. Cette méthode de bootstrap a néanmoins inspiré un module de S.A.G.A.C.E. qui s'avère ainsi capable d'apprendre par imitation.

Un jeu à information incomplète se prête particulièrement bien à l'étude de la modélisation de l'adversaire : le Poker. Tout joueur de Poker sait que les seules façons d'y bien jouer prennent en considération un modèle des adversaires (une idée de la façon dont ils pensent) et l'image que l'on donne de soi-même (la notion de bluff). Plus généralement, le Poker est considéré comme un excellent terrain d'investigation pour l'Intelligence Artificielle parce qu'il s'agit d'un jeu qui exploite de nombreuses activités intellectuelles humaines : la prise de décision face à des situations évolutives dont certains éléments sont cachés, la prise de décision en situation de risques, la gestion d'un capital, l'estimation des caractéristiques d'une situation, etc.

Findler [Findler, 1961,1977,1978] a écrit que « pour apprendre à un ordinateur à jouer au Poker, il faut comprendre le processus humain d'acquisition de l'information ». Findler a réalisé un système appuyé sur un modèle des joueurs humains. Son but n'était pas de concevoir un bon joueur de Poker capable de modéliser ses adversaires mais de reproduire en simulation les processus cognitifs mis en œuvre par les joueurs humains. Malheureusement, ces différents travaux ont abouti à des joueurs informatiques dont le niveau était peu élevé par rapport à de véritables joueurs humains.

A la tête du groupe « The University of Alberta GAMES Group », Schaeffer s'intéresse maintenant (après le jeu de checkers) au Poker [Billings, 1995, 1998], [Papp, 1998]. Schaeffer suggère que les études sur la Théorie des jeux doivent être maintenant orientées vers les jeux à information complète et imparfaite ou à information incomplète.

En effet, les études précédentes sur les jeux à information complète et parfaite ont conduit à la réalisation de systèmes « de force brute » parfois extrêmement performants (comme Deep Blue) [Hsu, 1990] mais elles n'ont pas grand intérêt pour la recherche relative aux processus de décision humains par rapport à des problèmes concrets.

Le groupe « Poker » de l'université d'Alberta a proposé le tableau suivant résumant la problématique générale du Poker [Billings, 1998] :

Problème général	Aspects relatifs du Poker
Information imparfaite.	Les cartes des adversaires sont ignorées.
Compétition entre plusieurs intervenants.	Jeu à plusieurs joueurs.
Management du risque.	Stratégies des mises et leurs conséquences.
Modélisation des intervenants.	Identification de la stratégie des adversaires et exploitation des faiblesses.
Tromperie.	Bluff, changement de stratégie,…
Non fiabilité de l'information.	Prise en compte de la tromperie de l'adversaire.

Un bon joueur de Poker doit être capable de gérer tous les aspects du jeu. Une excellente gestion du risque ne saurait compenser une grande « lacune en tromperie » parce que les adversaires humains sont prompts à exploiter les joueurs prédictibles.

A l'évidence, la réalisation d'un programme de niveau international au Poker permettra assurément d'apprendre beaucoup sur les processus de décision humains et aura naturellement de grandes applications dans les activités qui requièrent des facultés équivalentes.

Le groupe de Schaeffer développe un programme (Loki). Pour l'instant les modules d'estimation des mains, des stratégies simples de mise et d'estimation statistique de la valeur du pot sont opérationnels et relativement performants. En revanche, au moment où ce manuscrit s'écrit[1], Loki est très prédictible : il rejouera exactement de la même façon dans des circonstances semblables. Les préoccupations majeures actuelles des concepteurs de Loki sont la modélisation de l'adversaire, le bluff et la stratégie de mises.

---

[1] Ou plutôt « se fait écrire » !

La faculté de modéliser ses adversaires au Poker fait la différence entre deux joueurs de même niveau d'expertise du jeu. Le succès limité des approches de Carmel et Markovith [Carmel, 1996] et de Iida et al. [Iida, 1997] concernant la modélisation de l'adversaire par généralisation de l'algorithme minimax ont convaincu l'équipe de développement de Loki qu'il fallait s'orienter dans une autre direction. L'approche retenue se fonde sur une analyse du comportement passé de chaque adversaire ; plus exactement sur l'historique de ses mises en fonction de ses mains quand elles sont connues[1].

A tout moment, Loki détermine une probabilité de distribution des cartes de l'adversaire en fonction de son modèle (le modèle que Loki a de lui) et des mises qu'il effectue. Quand Loki n'a pas encore de modèle d'un adversaire particulier, il utilise un modèle minimal générique (correspondant à un joueur moyen).

Le modèle maintenu par Loki pour chaque adversaire consiste en deux tableaux :

- Un tableau (T_1) du poids affecté à chaque ensemble possible de deux cartes initiales[2] ;
- Un tableau (T_2) de fréquences des actions choisies.

T_1 permet une estimation de la main de l'adversaire. T_2 est une répartition des fréquences de chaque action de l'adversaire dans chaque situation. T_2 est construit en supposant, en première approximation, qu'un joueur n'accepte de miser (quand il a le choix) que si sa main fait partie des 15% meilleures mains possibles. Ensuite, le modèle est ajusté en fonction du comportement observé.

Le système complet de modélisation est relativement complexe. Il prend en compte un certain nombre de connaissances d'expert sur le Poker et suppose (indirectement) que les adversaires de Loki sont de bons joueurs.

Ce système comprend également un système d'abstraction des modèles. Quand Loki énumère les mains possibles de ses adversaires, il prend en compte ses propres cartes. S'il possède 2 As et, s'il y en a un autre parmi les cartes communes, il sait qu'aucun adversaire ne peut avoir également une paire d'As. Lorsqu'il doit estimer la stratégie de ses adversaires, il doit prendre en compte le fait que, eux, ne savent pas qu'il a 2 As. Ainsi, Loki possède une forme simplifiée de modèle de lui-même afin d'imaginer sur quel(s) critère(s) chaque adversaire mise (ou pas).

---

[1] Lokibot joue à la variante « Hold'em Poker » (utilisée pour les championnats du monde de Poker). Dans cette variante, comme dans d'autres, si un joueur reste seul (les autres ont abandonné la manche), il ne montre pas ses cartes et sa main initiale reste inconnue ce qui ne permet pas d'estimer sa stratégie de mise durant les manches correspondantes.

[2] Au Hold'em, chaque joueur reçoit 2 cartes qu'il est le seul à connaître avant la première phase de mises. Ensuite, il devra obtenir la meilleure main possible en utilisant 5 cartes parmi un ensemble constitué de ses 2 cartes et de 5 cartes communes à tous les joueurs. La seule information inconnue des adversaires est donc la valeur des 2 cartes que chaque joueur a reçues en début de manche.

Pour l'instant (à l'heure ou s'écrit ce manuscrit), Loki n'utilise les tableaux T_2 que pour l'ajustement des tableaux T_1. Il ne les utilise pas directement pour ajuster sa propre stratégie en fonction de celle de ses adversaires. Par ailleurs, l'auto-modélisation (le modèle que Loki gère de lui-même) n'est que peu exploitée et Loki ne tente pas de tromper ses propres adversaires en leur fournissant délibérément des informations fallacieuses[1].

Les expériences conduites ont montré [Papp, 1998] que la modélisation générique[2] (GOM) de l'adversaire dans Loki lui permettait de battre constamment, et avec un net avantage, des copies de lui même ne n'exploitant pas la modélisation. Elles n'ont, en revanche, pas montré de différence notable en qualité entre la modélisation générique et la modélisation spécifique[3] (SOM).

Enfin, il convient de remarquer que les séries d'expériences menées l'ont été contre d'autres joueurs informatiques qui ne variaient pas leurs stratégies (particulièrement des copies « tronquées » de Loki lui-même). Cela ne permet donc pas de se faire une idée précise des performances de Loki face à des joueurs humains. Il n'est pas non plus évident que l'approche de l'anticipation (le modèle de l'adversaire) soit adaptée contre les joueurs humains. Cependant, la méthodologie est originale et intelligente et les expériences en cours (Loki est maintenant un joueur officiel en jeu IRC[4]) devraient confirmer son « talent ».

Pour l'équipe de développement de Loki, le procédé d'amélioration du programme est cyclique. La modélisation actuelle de l'adversaire semble satisfaisante en l'état et les prochains efforts vont concerner d'autres aspects du jeu [Billings, 1998]. Quand ces derniers seront « traités » de façon satisfaisante, l'équipe reprendra son travail sur l'anticipation par modélisation des adversaires.

5.2.6 Les contre-mesures

« L'avenir n'est plus ce qu'il était » P.Valery

L'Anticipation peut être utilisée, dans un contexte de jeu, comme une contre-mesure à l'Adaptation. Par exemple, si un joueur apprend à modifier ses actions en s'adaptant à son adversaire, il devient, a priori, meilleur que ce dernier. Pourtant, si l'adversaire en question est doté de techniques ou procédures anticipatives performantes, il pourra prévoir comment se fera l'adaptation du joueur. Connaissant la façon dont le joueur s'adapte, il lui sera facile de le contrer.

---

[1] Au Poker, contre de nouveaux adversaires, il est parfois très subtil de perdre de façon délibérée (en jouant visiblement mal) des manches sans grand intérêt (pot de petite valuer) afin de donner une image faussée de ses capacités de jeu.

[2] Elle ne maintient pas de statistiques des actions et l'ajustement des tableaux T_1 traite toutes les actions d'un adversaire sans tenir compte expressément de ses stratégies.

[3] Elle distingue les différents types de joueurs en fonction de leur façon de jouer (utilisation des tableaux T_2).

[4] Internet Relay Chat (espace du Web réservé aux jeux en ligne).

Cela pose un nouveau problème : comment 'lutter' contre un adversaire anticipant ? En effet, si celui-ci est en mesure, d'une façon ou d'une autre, de prévoir les actions de son adversaire, il prend un ascendant absolu.

La première idée, certainement trop simpliste, consiste à s'en remettre plus ou moins au hasard pour le choix de ses actions afin d'être imprévisible. Il est évident que lorsque la situation demande une réflexion intense ou une planification intelligente, le hasard ne peut être performant (sauf, évidemment s'il s'agit de la stratégie mixte optimale au sens de Nash, mais avec les limitations déjà évoquées dans le chapitre consacré à la Théorie des Jeux).

L'idée de s'en remettre au hasard est astucieusement émise par E.A. Poe au sujet de sa nouvelle 'La lettre volée'. Dans cette histoire, il est question d'un garçon qui gagnait toutes les billes de son école au jeu pair/impair dans lequel il s'agit de deviner si le nombre de billes que cache l'adversaire dans sa main est pair ou impair. La stratégie du garçon consistait uniquement à forcer les traits de son propre visage pour calquer ceux de son adversaire dans le but de ressentir les sentiments, les émotions correspondantes - qui lui venaient alors naturellement à l'esprit - pour déterminer ceux de l'adversaire. Ainsi, connaissant l'état d'esprit de l'adversaire, le garçon était en mesure de savoir comment il pensait et, donc, comment il jouait. (Il s'agit d'une des méthodes cognitives dont nous avons parlé pour l'anticipation dans le paragraphe §3.3). Poe écrit « Il s'agit purement et simplement d'une identification de la part du raisonneur de ses facultés intellectuelles avec celles de son adversaire... la façon de faire mieux consiste à ne pas raisonner du tout ! » (et de s'en remettre donc au hasard pour choisir le nombre de billes que l'on tient dans la main).

« ...Mais il erre sans cesse par trop de profondeur ou par trop de superficialité pour le cas en question, et plus d'un écolier raisonnerait mieux que lui. J'ai connu un enfant de huit ans, dont l'infaillibilité au jeu de pair ou impair faisait l'admiration universelle. Ce jeu est simple ; on y joue avec des billes. L'un des joueurs tient dans sa main un certain nombre de ses billes, et demande à l'autre : pair ou non ? Si celui-ci devine juste, il gagne une bille ; s'il se trompe, il en perd une. L'enfant dont je parle gagnait toutes les billes de l'école. Naturellement, il avait un mode de divination, lequel consistait dans la simple observation et dans l'appréciation de la finesse de ses adversaires. Supposons que son adversaire soit un parfait nigaud, et levant sa main fermée, lui demande pair ou impair ? Notre écolier répond : impair, - et il a perdu. Mais à la seconde épreuve, il gagne, car il se dit en lui-même : le niais avait mis pair la première fois, et toute sa ruse ne va consister qu'à lui faire mettre impair à la seconde ; Je dirai donc : impair ; - il dit impair, et il gagne.

Maintenant, avec un adversaire un peu moins simple, il aurait raisonné ainsi : ce garçon voit que, dans le premier cas, j'ai dit impair, et, dans le second, il se proposera, - c'est la première idée qui se présentera à lui, - une simple variation de pair à impair comme a fait le premier bêta ; mais une seconde réflexion lui dira que c'est là un changement trop simple, et finalement il se décidera à mettre pair comme la première fois. - Je dirai donc pair. - Il dit pair, et gagne. Maintenant ce mode de raisonnement de notre écolier, que ses camarades appellent la chance, - en dernière analyse, qu'est-ce que c'est ?

- C'est simplement, - dis-je, - une identification de l'intellect de notre raisonneur avec celui de son adversaire.

- C'est cela même, - dit Dupin ; - et quand je demandai à ce petit garçon par quel moyen il effectuait cette parfaite identification qui faisait tout son succès, il me fit la réponse suivante :

« Quand je veux savoir jusqu'à quel point quelqu'un est circonspect ou stupide, jusqu'à quel point il est bon ou méchant, ou quelles sont actuellement ses pensées, je compose mon visage d'après le sien, aussi exactement que possible, et j'attends alors pour savoir quels pensers ou quels sentiments naîtrons dans mon esprit ou dans mon cœur, comme pour s'appareiller et correspondre avec ma physionomie. »

Cette réponse de l'écolier enfonce de beaucoup toute la profondeur sophistique attribuée à La Rochefoucaud, à La Bruyère, à Machiavel et à Campanella.

- Et l'identification de l'intellect du raisonneur avec celui de son adversaire dépend, si je vous comprends bien, de l'exactitude avec laquelle l'intellect de l'adversaire est apprécié.

- Pour la valeur pratique, c'est en effet la condition, - répliqua Dupin.

(Histoires extraordinaires « La lettre volée » Edgar Allan Poe - Traduction de Charles Baudelaire).

Une méthode plus efficace que l'utilisation du pur hasard comme contre-mesure à l'anticipation est basée sur la devise de Socrate « Connais-toi toi même » écrite sur le fronton du temple de Delphes ou bien sur les écrits de Suntzu dans l'art de la guerre[1] « Connaissez l'ennemi et connaissez-vous vous-même, en cent batailles vous ne courrez jamais aucun danger ». Il s'agit de se donner les moyens de pouvoir anticiper soi-même sur ses propres actions afin de savoir toujours à l'avance lorsque l'on devient trop prévisible. Cela permet de modifier sa stratégie lorsque l'on pense que l'adversaire est en mesure de prédire ce que l'on jouerait 'normalement'.

Se pose alors une nouvelle fois le problème de la profondeur de l'anticipation : l'adversaire peut chercher un moyen de prédire quand on va changer de stratégie...

Ces considérations, ainsi que l'histoire de Poe, ont inspiré la définition des fonctionnalités de S.A.G.A.C.E.

---

[1] Nous suggérons la lecture du livre « The art of war » de Griffith (Oxford University Press).

6 Les stratégies humaines

> La mémoire est souvent la qualité
> de la sottise.
> François René de Chateaubriand,
> 1768-1848

6.1 Introduction

Il n'est pas raisonnable de prétendre décrire ni même connaître les processus cognitifs qui régissent le développement de stratégies pour les humains. Pourtant, une simple introspection, ou l'interrogation d'un expert de jeu peuvent donner quelques pistes sur les caractéristiques ou les spécificités des stratégies humaines.

Pour des jeux aussi simples que le Tic-tac-toe on peut penser que la stratégie d'un humain[1] est purement déterministe. Il sait ce qu'il faut jouer parce qu'il a déjà joué des dizaines de fois et que le jeu ne présente plus aucune difficulté pour lui.

Pour les jeux à information complète et parfaite, il semblerait que la mémoire des parties déjà disputées joue un grand rôle. C’est aussi le cas de la mémoire de coups particuliers longuement étudiés comme le font les grands champions d'échecs, souvent en fonction de traités d'autres experts. Il faut noter que l'expertise des humains est parfois erronée même lorsqu'elle a été transmise année après année. Ainsi, le programme Chunker [Berliner, 1983] a permis de découvrir de nombreuses erreurs dans la littérature des échecs. A l’évidence, tant que les experts humains jouaient entre eux, ils s'accordaient sur le résultat de ces fins de parties et ne terminaient pas le jeu.

Pour les jeux à information incomplète, la mémoire semble également jouer un grand rôle. Par exemple, pour les jeux de cartes, les grands joueurs disent estimer les chances d'apparition des cartes distribuées dans les jeux des adversaires en début de partie en fonction des parties précédentes.

---

[1] Par humain nous entendons humain doué d'une intelligence commune.

Enfin, dans la plupart des cas, les joueurs humains disent jouer différemment suivant leurs adversaires, suivant l'idée qu'ils se font d'eux. Cette notion de modèle de l'adversaire est sans doute celle qui différencie le plus les stratégies humaines de la plupart des stratégies « informatiques » actuelles. Les joueurs de poker notamment adaptent leur stratégie en fonction de l'idée qu'ils se font de leurs adversaires (*bluffeurs*, « *épiciers* », *téméraires*…). Les meilleurs joueurs parviennent à fausser l'image qu'ils donnent d'eux en jouant, dans les coups de moindre importance, d'une façon qui ne correspond pas à leur stratégie normale. De même, certains entêtements dans les surenchères ne s'expliquent chez les bons joueurs que par la volonté de confirmer le modèle qu'ils ont de leurs adversaires[1].

Un autre aspect majeur des stratégies développées par les joueurs humains est le recours au hasard. Un joueur ayant l'impression d'être devenu prédictible au cours des parties a tendance à s'en remettre au hasard pour tromper ses adversaires. Le fait est que l'Homme éprouve de grandes difficultés à manipuler le hasard.

6.2 La mémoire humaine

Les psychologues considèrent que la mémoire humaine se décompose en plusieurs sous-mémoires auxquelles sont associées des caractéristiques et des processus distincts. Ils distinguent trois types de mémoires particuliers :

- La mémoire sensorielle :

  Capable de stocker une petite quantité d'information pour une période très brève.

- La mémoire à court terme :

  Egalement de faible capacité, elle peut néanmoins maintenir l'information plus longtemps que la mémoire sensorielle.

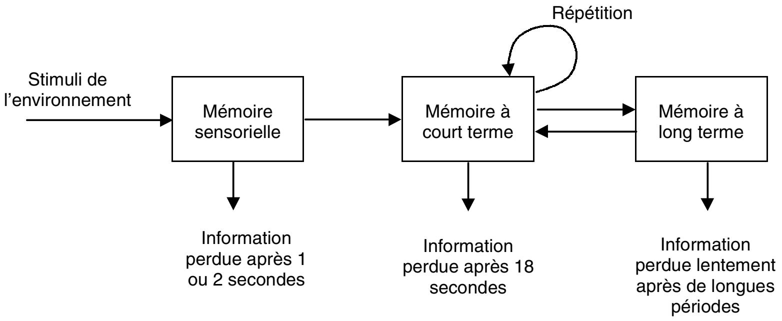
- La mémoire à long terme :

  Elle est de grande capacité et peut conserver des informations pour de longues durées. Certaines informations finissent par être oubliées, d'autres jamais.

---

[1] Cette technique courante au poker s'appelle « l'achat d'information ».

Dans les années 60, les psychologues ont proposé un modèle de cette décomposition [Waugh, 1965] et [Shiffrin, 1969] :



D'après ce modèle, une information peut être simplement "perçue" puis oubliée presque aussitôt ; elle peut "transiter" par la mémoire à court terme où elle peut rester présente tant qu'un effort de mémorisation particulier est effectué (par répétition d'après le modèle[1]) ; suivant l'importance que lui accorde le sujet et l'effort qu'il y consacre, elle peut être conservée à long terme. La notion d'*effort* est subjective, un événement important étant retenu apparemment sans effort.

6.2.1 Mémoire sensorielle

Il s'agit simplement d'un stockage ponctuel des sensations liées aux stimuli reçus de l'environnement. On peut considérer la période avant la perte de l'information comme une réminiscence involontaire. Toutes les informations qui n'ont pas retenu l'attention sont perdues. Les autres, sont stockées dans la mémoire à court terme.

6.2.2 Mémoire à court terme

Cette mémoire est particulièrement mise à contribution lors de la réalisation de tâches répétitives sur une courte période de temps. Par exemple, recopier une adresse inconnue à la main sur de nombreuses enveloppes est une tâche faisant appel à la mémoire à court terme. Pour les premières enveloppes, on relit souvent l'adresse (processus de répétition) puis, au bout d'un moment, on se souvient de plus en plus de mots. Si cette tâche est répétée pour de nombreuses enveloppes, il est clair que l'adresse sera finalement apprise par cœur et sera stockée dans la mémoire à long terme.

---

[1] Stern prend l'exemple d'un numéro de téléphone lu et répété jusqu'à sa composition puis oublié ensuite. [Stern, 1985].

Comme Stern le fait remarquer [Stern, 1985], il est très difficile d'assigner une capacité à la mémoire à court terme. Cependant, Miller [Miller, 1956] qui fait autorité dans ce domaine a estimé à sept plus ou moins deux (7±2) le nombre d'informations élémentaires (chunks[1]) pouvant être stockées dans la mémoire à court terme.

Les chunks présents en mémoire à court terme peuvent faire référence à d'autres chunks présents, eux, en mémoire à long terme. C'est pour cela qu'il est plus facile de retenir des phrases cohérentes que des suites de mots sans lien les uns avec les autres. Dans le premier cas, les chunks en mémoire s'apparient avec des chunks présents en mémoire à long terme (construction grammaticale, sens, logique, etc.).

Les capacités de la mémoire à court terme dépendent évidemment de chaque individu, qu'il soit jeune ou âgé, homme ou femme, etc. Parkinson (1982) puis Pollack, Johnson et Knaff (1959) ont fait de nombreuses expériences pour mesurer plus précisément la capacité de la mémoire à court terme.

6.2.3 Mémoire à long terme

Cette mémoire contient les informations retenues consciemment ou pas, volontairement ou pas. Les psychologues ne proposent aucune mesure de la capacité de cette mémoire tant elle pose de questions. Certains affirment qu'elle a une capacité infinie.

Certains neurophysiologues ont proposé des mesures en fonction du nombre de neurones dans le cerveau humain et des connections entre eux [Milner, 1970]. D'autres, tels que Hebb [Hebb, 1949] ont cherché à modéliser les processus de mémorisation afin d'estimer le nombre de cellules corticales nécessaires pour le stockage d'un chunk.

6.2.4 Mémoire et stratégies humaines

Pour un joueur, la mémoire joue un rôle primordial. Degroot a effectué des études sur les capacités des champions au jeu d'échecs [Degroot, 1965]. Il s'est particulièrement intéressé à l'importance des chunks dans la mémoire à long terme des joueurs experts pour comprendre ce qui les distingue des joueurs moins expérimentés. Plus particulièrement, il a cherché à comprendre le rôle de l'expérience dans le développement de ces chunks.

---

[1] Le mot « chunk » est souvent traduit en français par « morceaux ». Ce peut être des mots, des images… par extension, le « chunk » permet une « représentation unifiée unique » d'éléments similaires.

Degroot [Degroot, 1965] a considéré le nombre de coups envisagés par l'expert par rapport à celui du joueur « normal ». Après une série de tests sur les capacités de mémorisation, il est arrivé à la conclusion que les capacités de mémoires des experts étaient beaucoup plus importantes que celles des joueurs normaux. Il s'est alors naturellement demandé si ces capacités avaient un rôle plus important que d'autres facultés pour le niveau de jeu aux échecs (calcul, intuition, etc.). D'après lui, la principale distinction entre un expert et un novice est la capacité de mémorisation de chunks (liés à des configuration précises de parties de l'échiquier) beaucoup plus grande chez les experts.

Chase et Simon ont conduit des expériences similaires [Chase, 1973]. Ils ont présenté pendant 5 secondes des échiquiers (contenant 24 à 26 pièces) à trois catégories de joueurs (experts, amateurs, débutants). L'arrangement des pièces sur les échiquiers correspondait à des milieux de parties décrites dans de nombreux ouvrages de référence. Chase et Simon ont constaté que le pourcentage de pièces replacées correctement dépendait directement du niveau des sujets (62% pour les experts, 34% pour les amateurs et 18% pour les débutants).

Afin de mesurer l'importance des capacités de mémoires pour les joueurs d'échecs Chase et Simon (1973), ont conduit de nouvelles expériences (par rapport à celles de Degroot). Les échiquiers présentés aux différentes catégories de joueurs étaient, cette fois, remplis au hasard. Les performances ont été à peu près les mêmes pour chaque catégorie. La conclusion est que la taille de la mémoire n'est pas déterminante mais que l'aptitude à mettre en correspondance des patterns perçus avec des chunks mémorisés en mémoire à long terme l'est. Ainsi, le joueur expert est, semble-t-il, capable de reconnaître des configurations d'échiquiers qui se sont révélées (dans le passé) être efficaces ou pas[1].

Dans les jeux très simples à information complète et imparfaite (*Pierre/Ciseaux/Papier* ; *Pair/impair* ; etc.), les joueurs humains adaptent leurs stratégies en fonction de leurs adversaires de façon très réactive. Il semblerait qu'un joueur humain prenne en considération uniquement les derniers coups pour effectuer son choix. Dans des jeux à information complète et imparfaite légèrement plus complexes (*Alésia*[2] notamment), il semblerait que certaines configurations particulières retiennent l'attention des joueurs et qu'ils réagissent en fonction des conséquences passées éventuelles de leurs choix. Il s'agit d'une constatation que nous avons pu faire par l'observation de joueurs humains au cours des séries d'expérimentations que nous avons menées.

---

[1] Cette aptitude a été nommée « chunking ability » par les psychologues.

[2] Ce jeu sera décrit ultérieurement. Il a fait l'objet d'une implémentation de la méthode S.A.G.A.C.E.

6.3 Le hasard

Dans les jeux à information imparfaite, le hasard joue un grand rôle. La Théorie des jeux indique, comme nous l'avons signalé, que les stratégies optimales pour ces jeux correspondent à des répartitions probabilistes des stratégies pures. Pour « suivre » une telle stratégie, il faut se donner les moyens de « générer » des nombres aléatoires qui vont déterminer, pour chaque coup, le choix à effectuer.

Dans l'exemple du §4.6.7.3., la stratégie optimale du joueur ligne est la suivante :

$$x = \left(\frac{3}{4}, \frac{1}{4} \right)$$

Cela signifie qu'il doit choisir la première action possible trois fois sur quatre et l'autre une fois sur quatre, et ce, aléatoirement.

6.3.1 Les problèmes du hasard

Il paraît facile de générer des suites de nombres aléatoires. Pourtant, la définition d'une suite aléatoire est stricte. Intuitivement, une suite est uniformément aléatoire si « elle est dépourvue de structure interne » comme le définit Chaitin [Chaitin, 1996] et si « chaque élément est en moyenne autant représenté que les autres ».

Une première définition formelle est la suivante : une suite est dite uniformément aléatoire si elle a été engendrée par un processus purement aléatoire :

Si on considère les trois suites binaires suivantes :

```
1.     1 1 1 1 1 1 0 0 1 1 1 1 0 1 1 1 1 1 0 1
2.     1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
3.     1 1 1 0 1 0 0 0 1 0 1 0 0 0 1 0 1 1 0 1
```

En utilisant un générateur aléatoire idéal, les trois suites précédentes ont la même probabilité d'apparition parmi les 2^{20} possibles[1]. Or, la première contient beaucoup plus de 1 que de 0 et la seconde présente une structure interne évidente (un 1 est toujours suivi d'un 0 et inversement). Seule la troisième semble répondre aux deux critères intuitifs.

Une définition plus satisfaisante est la suivante : une suite est dite aléatoire si :

1. la probabilité d'apparition d'un élément particulier est égale à celle des autres,
2. la somme des probabilités d'apparition de tous les éléments est égale à 1,
3. chacun des éléments qui la constituent est indépendant des autres.

---

[1] Les suites contiennent 20 éléments qui peuvent être 0 ou 1ce qui représente 2^{20} possibilités.

Si on considère les trois suites binaires précédentes, les suites 1. et 2. sont écartées ce qui répond aux critères intuitifs.

Pour une suite aléatoire au sens large du terme (sans la restriction d'uniformité), cette définition peut être exprimée sous la forme suivante :

Une suite est aléatoire si elle est incompressible[1]. Cela signifie qu'il n'est pas possible de définir une suite aléatoire sous une forme contenant moins d'éléments d'information que la suite elle-même. La suite 2. peut être décrite par : 10 x [1,0] (dix fois le couple 01). Elle est compressible et ne satisfait donc pas cette dernière définition.

On trouvera dans [Chaitin, 1996] une présentation détaillée de la notion de suites aléatoires. Cette définition concernant l'incompressibilité des suites aléatoires pose un problème important : une suite binaire aléatoire ne peut être définie par un programme (sinon, on utiliserait le programme correspondant pour obtenir une version compressée de ses chiffres binaires [Delahaye, 1995]). Ainsi, un ordinateur ne peut générer que des suites imparfaitement aléatoires.

Cela peut sembler gênant pour la réalisation de programmes capables de jouer à des jeux à information complète et imparfaite puisque, par définition, aucun programme ne pourra « suivre » idéalement la Théorie des jeux, c'est-à-dire la solution optimale. Bien évidemment, la plupart des algorithmes générateurs de suites aléatoires mis en œuvre dans les ordinateurs sont tout à fait satisfaisants pour cet usage[2].

6.3.2 L'humain est-il capable de manipuler le hasard ?

Si un ordinateur est, en théorie, incapable de générer des suites de nombres idéalement aléatoires, qu'en est-il pour les humains ?

Une simple expérience, facilement renouvelable, illustre la difficulté que nous avons à générer de telles suites : voici une suite binaire (censée être uniformément aléatoire) générée d'un trait par un sujet humain[3] :

`1 0 1 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 1 1 0 0 0 1 0 1 0 0`

---

[1] Kolmogorov et Chaitin ont proposée cette définition en 1965 sous la forme suivante : « une suite de chiffre est aléatoire quand le plus petit algorithme nécessaire pour l'introduire dans un ordinateur contient à peu près le même nombre de bits que cette suite » [Chaitin, 1996].

[2] A titre de curiosité, on trouvera des exemples de « mauvais» générateurs dans « Aléas du hasard informatique » (Delahaye *Pour la science*. Mars 98).

[3] Cette série a été simplement tapée au clavier pendant la rédaction de ce texte par l'auteur lui-même…

On peut y compter 18 fois le chiffre 0 et 13 fois le chiffre 1. On peut décompter 11 fois le couple ‘10’, 10 fois ‘01’, 7 fois ‘00’ et 2 fois ‘11’. Le critère d’uniformité n’est absolument pas respecté.

Ce n’est qu’une simple expérience qui ne prétend pas se renouveler à l’identique systématiquement mais qui illustre la difficulté qu’ont les humains à générer spontanément des suites aléatoires.

En prenant le temps de la réflexion, il est tout à fait possible de générer une suite plus satisfaisante. En revanche, si la série doit être générée de façon discontinue (en effectuant d’autres tâches en même temps), un autre problème apparaît, lié à celui de la capacité de la mémoire à court terme… Il est très difficile de se souvenir des nombres précédemment générés pour en générer d’autres satisfaisant globalement les critères d’une suite aléatoire.

Dans un contexte de jeu, un joueur humain voulant générer des nombres aléatoires le fait de façon discontinue (il attend le coup de l’adversaire, la confrontation des choix, etc.). Cela explique sans doute en partie le succès de programmes fondés sur la reconnaissance de patterns face à des joueurs humains[1].

Une méthode satisfaisante pour un humain est d’utiliser un artefact (comme un dé) pour l’aider à générer des nombres aléatoires. Delahaye suggère une méthode originale et efficace pour effectuer des tirages au sort mentaux [Delahaye, 1995c] : pour générer de façon uniforme un nombre binaire aléatoire, il suffit de penser à un mot moyennement long (comme « anticipation ») puis d’en compter le nombre de lettres. Si ce nombre est pair, le nombre généré est 0 et 1 sinon.

Reste à vérifier que, en moyenne, les mots usuels français ont la même parité par rapport à leur nombre de lettres… Les nombres entiers peuvent être générés à partir des nombres binaires mais cela demande un certain nombre de tirages. Delahaye donne également une méthode pour générer des nombres suivant des répartitions non uniformes. Il s’agit tout de même de méthodes difficiles à mettre en œuvre pendant un jeu !

---

[1] Quand nous avons programmé la méthode de Minasi [Minasi, 1991] (qui repose sur une technique de reconnaissance de patterns) pour le jeu *Pierre/Ciseaux/Papier,* nous avons pu constater à quel point un joueur humain éprouve de grandes difficultés à joueur aléatoirement, même lorsqu’on lui demande de le faire.

Une autre méthode est utilisable par un individu particulier : il existe au Japon une personne capable de réciter les 10.000 premières décimales de π. Certes π n'est pas un nombre aléatoire au sens fort du terme (il est simplement transcendant), cependant, comme cette personne est sans doute la seule à connaître autant de décimales de π, si elle utilise cette source d'inspiration, elle sera imprévisible pour un autre humain !

6.4 L'adaptation

La principale caractéristique de l'intelligence humaine est *l'Adaptation*. Dans un contexte de jeu, il est rare de voir un joueur humain persévérer dans un comportement le menant inéluctablement à la défaite. Même si les solutions alternatives ne sont pas plus efficaces, il s'agit de la forme la plus simple d'adaptation : ne pas renouveler ce qui est mauvais. Cette adaptation est très réactive.

Les bons joueurs sont capables de s'adapter en imaginant de nouvelles stratégies plus efficaces et de s'adapter à chacun de leurs adversaires.

6.4.1 Adaptations réactives

Les adaptations réactives sont celles qui ne demandent pas d'effort intellectuel de la part du joueur. Le joueur réagit instantanément à un échec ou une réussite. Typiquement, au jeu *Pierre/Ciseaux/Papier*, un joueur ayant perdu trois fois de suite en jouant *Pierre* pourrait décider de façon réactive de jouer *Ciseaux* ou *Papier.*

Ce type d'adaptation n'est efficacement possible que pour des jeux très simples ou dans des configurations de jeux bien particulières. Aux échecs, par exemple, les débuts de parties sont très particuliers. Les grands joueurs vont définir le type de la partie en choisissant, par leurs premiers coups, une ouverture (pour l'un) et une défense appropriée (pour l'autre). Certaines ouvertures peuvent être contrées par différents systèmes de défense.

Par ailleurs, de nombreuses ouvertures et défenses commencent de la même façon. Ainsi, les joueurs réagissent très rapidement aux coups de l'adversaire pour définir, ensemble, l'orientation de la partie. A tout moment le premier joueur adapte son choix d'ouverture à celui de défense du second et inversement de façon réactive. Quand les deux joueurs parviennent à un schéma de jeu qui échappe à leur connaissance commune des ouvertures et défenses, le rythme de la partie change, les joueurs deviennent moins réactifs.

6.4.2 Adaptations cognitives

Un bon joueur est capable d'ajuster sa stratégie en fonction des résultats qu'elle provoque. Une compréhension du jeu est alors indispensable pour que l'adaptation soit faite de façon efficace. La compréhension d'un jeu s'acquiert par apprentissage : apprentissage des règles dans un premier temps, puis des mécanismes sous-jacents.

Un bon joueur humain sera également capable d'inventer des stratégies en fonction de la situation et de sa compréhension du jeu. Dans les jeux à information complète et imparfaite, le joueur s'adapte également en fonction de l'idée qu'il se fait de son adversaire.

6.5 L'apprentissage

L'apprentissage est une faculté intellectuelle particulièrement mise en avant dans les jeux.

6.5.1 Apprentissage par cœur

Un joueur humain désirant jouer aux échecs, par exemple, doit avant tout en *apprendre* les règles. Chaque jeu possède ses règles et chaque joueur doit les connaître.

Les bons joueurs connaissent par cœur certaines configurations du jeu et la façon optimale de jouer dans ces situations (Les champions d'échecs connaissent par cœur un grand nombre d'ouvertures et de fins de partie).

Pour certains jeux très simples (*Tic-tac-toe* notamment) certains joueurs humains connaissent parfaitement toutes les situations possibles et tous les coups à jouer (pour gagner ou obtenir le nul).

Les jeux à information complète et imparfaite ne permettent pas aux joueurs humains d'apprendre par cœur les coups optimaux à jouer dans chaque situation : au mieux, un joueur peut apprendre par cœur les répartitions probabilistes optimales… Il lui restera à déterminer chaque coup aléatoirement en conséquence.

6.5.2 Apprentissage supervisé

L'apprentissage supervisé dans les jeux pour des humains consiste en l'apprentissage, sous la direction d'un joueur confirmé, des règles du jeu et de ses particularités (bonnes stratégies générales, erreurs à ne pas commettre, etc.).

6.5.3 Apprentissage par imitation

Un joueur humain est capable d'imiter les stratégies qu'il apprend en observant ses adversaires ou partenaires. Il apprend ainsi par imitation.

Les grands joueurs eux-mêmes apprennent les stratégies qui ont été reconnues par les experts comme particulièrement efficaces. L'apprentissage des ouvertures aux échecs peut aussi bien être assimilé à un apprentissage par imitation qu'à un apprentissage par cœur.

6.5.4 Apprentissage par renforcement

L'humain apprend à partir de ses erreurs. Dans le cadre des jeux, un humain apprend au cours des parties à ajuster sa stratégie. Il apprend quels sont les coups efficaces, quels sont ceux qui ne le sont pas.

Une stratégie qui s'est révélée souvent efficace sera d'autant plus utilisée et inversement.

6.5.5 Apprentissage par découverte

Un joueur humain découvre de nouvelles stratégies au fur et à mesure qu'il joue. Pour peu que chaque partie soit différente, il apprend de nouvelles façons de jouer.

Pour les jeux les plus simples, un humain finit par connaître par cœur toutes les stratégies possibles et finit par ne plus apprendre par découverte.

6.6 La rationalité

Le manque de rationalité des humains ne fait aucun doute. Il explique notamment notre position sur l'inadaptation de la Théorie des jeux pour la réalisation de programmes se fondant sur elle pour jouer contre des humains. Non que la théorie soit moins efficace contre un humain que contre un autre type d'adversaire[1], mais elle ne tire aucun parti de cette irrationalité des joueurs humains.

Dans le cadre des jeux, la notion de rationalité est relativement claire : un joueur rationnel minimise ses pertes en calculant les conséquences les plus pessimistes de ses propres choix confrontés à tous ceux de ses adversaires. La théorie des jeux repose totalement sur la rationalité parfaite de tous les joueurs.

---

[1] La Théorie des jeux permet de garantir l'espérance de gain (théorique) contre tout adversaire.

Prix Nobel d'économie en 1978, Herbert A. Simon, a constaté que les comportements observés des acteurs économiques humains n'étaient absolument pas rationnels dans le sens généralement admis par les économistes et intégré dans les modèles. Cette hypothèse de rationalité, base de la Théorie des jeux, supposait que les comportements des acteurs économiques visaient une forme de maximisation (gains, intérêts, etc). Ainsi, Simon a substitué la «rationalité limitée» et l'«effort d'assouvissement» à l'hypothèse de rationalité et à l'effort de maximisation en se basant sur le fait que les humains sont, par nature, incapables de prendre en considération tous les paramètres (choix et conséquences) d'une décision parfaitement rationnelle [Simon, 1947], [Gilboa, 1989].

Certaines expériences illustrent tout à fait significativement cette absence de rationalité. Delahaye a proposé l'expérience suivante [Delahaye, 1996b] :

« *Imaginez qu'on vous propose le jeu suivant : tu me donnes 152.449 €*[1] *; ensuite, tu tires un billet dans cette urne qui contient 1 billet rouge et 99 billets bleus. Si tu tires le rouge, je te donne 152.449.017 € ; sinon, je ne te donne rien.*

En acceptant de jouer à ce jeu, vous gagnerez 152.449.017 € une fois sur cent, donc, en moyenne, vous gagnerez 152.449.017 € pour 15.244.902 €, ou 1.524.490 € pour 152.449 € engagés. L'espérance mathématique du jeu est donc de 1372041 € par partie jouée.

Cela semble extrêmement intéressant.

Posez-vous maintenant honnêtement la question (en imaginant que vous disposiez de cent cinquante deux mille quatre cent quarante neuf euros d'économie) : est-ce que je jouerai ? Je suis certain que, comme moi, vous répondriez non ».

Les joueurs humains semblent éprouver de grandes difficultés à appréhender la notion d'espérance de gain.

Les jeux coopératifs (comme *le dilemme itéré des prisonniers*) illustrent souvent les problèmes de rationalité des humains [Delahaye, 1995a & 1995b]. En effet, la rationalité d'un joueur particulier est incompatible avec la rationalité de l'ensemble. Le dilemme est très délicat pour un joueur humain.

Pour certains jeux simples à information complète et parfaite, un joueur humain peut être parfaitement rationnel : il suffit qu'il connaisse la stratégie *pure* optimale.

---

[1] Le texte original parlait d'un million de francs (une ancienne monnaie n'ayant plus cours aujourd'hui…) nous avons pris la liberté de traduire pour facilité la compréhension…

Par ailleurs, la rationalité dans certains types de jeux requiert des outils : pour être parfaitement rationnel (jouer de façon à optimiser ses gains) à un jeu à information complète et imparfaite, un joueur doit pouvoir jouer suivant des répartitions probabilistes… nous avons déjà mentionné les problèmes correspondants (§6.3).

6.7 Les fonctions d'utilité

Tous les théoriciens du jeu connaissent le manque de rationalité des joueurs humains. C'est pour cela qu'a été introduite la notion de *fonction d'utilité*. Plutôt que d'établir des stratégies par rapport à la rationalité des joueurs humains, la Théorie des jeux pour les jeux à information incomplète se fonde sur l'*utilité*. La fonction d'utilité d'un joueur indique sa préférence. Dans l'exemple précédent de Delahaye, on considère que la fonction d'utilité d'un joueur le conduit à refuser le jeu alors que fonction rationnelle l'aurait poussé à accepter le jeu.

La notion d'utilité explique l'engouement des humains pour les jeux des organismes publics tels que La *Française des jeux*. Les joueurs préfèrent perdre peu (un ou deux euros) en espérant gagner beaucoup, même si l'espérance de gain est négative, plutôt que de risquer de perdre beaucoup en pouvant gagner énormément avec une espérance de gain positive.

L'Utilité est un indicateur numérique cohérent permettant, idéalement, d'exprimer les préférences de façon cohérente et non-ambiguë. Si un décideur préfère A à B, l'utilité de A est plus grande que celle de B.

Formellement[1], on écrit : $U(A) \geq U(B)$. La préférence stricte s'indiquant alors $U(A)>U(B)$ et l'indifférence : $U(A)=U(B)$.

L'utilité obéit aux règles élémentaires suivantes :

La transitivité : si $U(A)>U(B)$ et $U(B)>U(C)$ alors $U(A)>U(C)$.

Le caractère total de la relation : si $U(A) \geq U(B)$ et $U(B) \geq U(A)$ alors $U(A)=U(B)$.

Samuelson [Samuelson, 1947] et Debreu [Debreu, 1984], notamment, ont formalisé la notion d'Utilité et ont participé au développement de la Théorie de l'Utilité.

---

[1] On trouvera une introduction claire et simple de la notion d'utilité dans [Boursin, 1996].

6.8 Anticipation

Lorsqu'il joue à un jeu à information complète et imparfaite, un joueur humain tente de prédire l'action de ses adversaires pour choisir sa propre stratégie. Cela nous a été confirmé par les différentes expériences (dont nous reparlerons par la suite) que nous avons pu mener lors de l'implémentation de la méthode S.A.G.A.C.E.

Dans ces expériences, les joueurs humains, quand on les interrogeait sur leur façon de jouer, déclaraient qu'ils jouaient en fonction des actions présumées de leurs adversaires.

Cependant, curieusement, les joueurs humains continuaient à tenir ces propos quand ils jouaient contre des stratégies déclarées aléatoires de l'ordinateur.

6.8.1 Modélisation

C'est le processus cognitif par lequel un joueur humain ou informatique crée un modèle de ses opposants.

Un joueur humain se crée une représentation de ses adversaires. Il les modélise et utilise le modèle correspondant pour ajuster ses stratégies.

Bien évidemment, le succès d'un joueur modélisant dépend de la pertinence de son modèle et de la façon dont il l'utilise. Par ailleurs, un modèle peut s'avérer inutilisable : si un joueur « découvre » que son adversaire joue aléatoirement, cela ne lui permet pas nécessairement d'être plus efficace.

Plus particulièrement, dans un jeu à information complète et imparfaite, s'il « découvre » que son adversaire utilise la Théorie des jeux, il saura seulement que son propre comportement n'aura aucune influence sur le résultat des parties (son adversaire se garantissant l'espérance de gain).

6.8.2 Réflexivité de l'anticipation

Il s'agit d'un problème général. Quand deux joueurs humains jouent l'un contre l'autre, chacun modélise l'autre. Dans chaque modèle, il faudrait idéalement, intégrer le fait que l'adversaire modélise également…

On doit alors considérer le degré de « conscience » de l'anticipation. Doit-on anticiper le fait que l'adversaire anticipe, qu'il anticipe sachant qu'on anticipe soi-même, etc. ?

Il paraît impossible de savoir jusqu'où s'effectue l'anticipation d'un joueur humain. Cependant, les joueurs interrogés font souvent mention de l'idée qu'ils ont de l'idée que se font les adversaires d'eux. Cela laisse supposer un degré d'ordre deux.

6.9 Prise en compte des stratégies humaines

Les différents systèmes décrits au chapitre précédent ont largement participé à l'inspiration de S.A.G.A.C.E. concernant l'utilisation d'un modèle de l'adversaire. Cependant, ces systèmes ne tiennent pas explicitement compte des stratégies humaines (des paramètres les conditionnant), exception faite de la machine de Schannon [Shannon, 1953] dont le mécanisme des mémoires présuppose les limites de la mémoire humaine. Gobet et Jansen ont cependant envisagé un programme capable de jouer aux Echecs basé sur un modèle explicite de la mémoire humaine [Gobet, 1994].

Ricaud a proposé, pour le jeu de Go, une approche originale consistant à modéliser la stratégie humaine par un mécanisme d'abstraction [Ricaud, 1995, 1997]. Le système GOBELIN, construit autour de cette approche, n'exploite pas de modèle d'un adversaire pour anticiper ses coups, mais exploite une forme de modélisation du jeu humain pour la construction de ses propres tactiques.

A travers d'autres jeux à information complète et parfaite : Tic-Tac-Toe ; lose- Tic-Tac-Toe (le réciproque du précédent : il faut faire aligner 3 pions à l'adversaire) et Achi (un Tic-Tac-Toe avec déplacement des pions après leur pose), Rattermann et Epstein ont comparé les façons de jouer d'humains experts ou débutants, d'une part, et d'une machine, d'autre part [Rattermann, 1995]. Ces jeux sont cependant trop simples pour prendre en considération des paramètres comme la mémoire ou l'anticipation. Cependant, l'étude correspondante permet de se faire une meilleure idée de la façon dont un joueur humain appréhende un jeu. Elle donne une idée de la différence entre un bon joueur et un mauvais : le bon joueur n'a pas nécessairement de plus grandes facultés mentales (concentration , mémoire, QI), il ne fait pas non plus un parcours mental plus grand des arbres de décision, mais il sait focaliser son attention sur les points clés d'une position. Il organise correctement son savoir et possède de meilleures procédures pour utiliser ce savoir.

Il semblerait, d'après cette étude, que les bons joueurs humains fassent beaucoup appel à des méta-règles comme « Regarder deux coups en avant pour choisir le coup à jouer » (30%), « Sélectionner un coup pour se défendre contre un plan simple de l'adversaire » (17%), « Répéter les coups précédents qui ont fait gagner » (11%) alors que les joueurs débutants font plutôt appel aux méta-règles de défense « Sélectionner un coup pour se défendre contre un plan simple de l'adversaire » (30%) (contre 17% pour les bons joueurs).

La solution S.A.G.A.C.E., présentée dans le chapitre suivant, prend en considération les aspects des stratégies humaines évoqués dans ce chapitre afin de construire et mettre à jour un modèle cohérent de l'adversaire dans les jeux à information complète et imparfaite. Ce modèle permet de prédire les coups de l'adversaire afin de les anticiper. Il tient également compte de sa propre influence sur le comportement de l'adversaire.

©Yom, 99

Dans ce jeu très classique de 'Pierre/Ciseaux/Papier', l'humain, convaincu qu'un ordinateur ne peut penser très finement, a deviné - pour d'obscures raisons - que celui-ci allait jouer le papier, alors, il décide de jouer les ciseaux. Pourtant, le robot ayant lu et analysé le document que vous lisez en ce moment sait comment anticiper : il sait que l'humain pense qu'il va jouer le papier et qu'il va donc lui-même jouer les ciseaux. Il choisit alors de jouer la pierre et... gagne !

S.A.G.A.C.E.

(**S**olution **A**lgorithmique **G**énétique pour l'**A**nticipation de **C**omportements **E**volutifs)

Sagacité :	Pénétration faite d'intuition, de finesse et de vivacité d'esprit « Une sagacité froidement cruelle qui devait lui permettre de tout deviner, parce qu'il savait tout supposer ». Balzac.
Sagace :	Du latin Sagax, Sagagis, qui a l'odorat subtil. Adj : qui a de la sagacité, avisé, clairvoyant, fin, perspicace, subtil. « Restait ce mari qui, pour peu qu'il fut sagace, devait se douter de leur liaison ». Huysmans.

7 La méthode S.A.G.A.C.E.

> «Le joueur qui n'observe pas la psychologie de son partenaire et ne modifie pas en conséquence sa manière de jouer doit forcément perdre vis-à-vis d'un adversaire dont l'esprit est assez souple pour varier son jeu en tenant compte de celui de l'adversaire» Borel, 1924

7.1 Introduction

La solution S.A.G.A.C.E. est une solution générique pour l'élaboration de programmes destinés à affronter des joueurs humains dans des jeux à information complète et imparfaite. S.A.G.A.C.E. pallie les défauts des approches existantes en tirant explicitement partie des caractéristiques des stratégies généralement développées par les humains dans les jeux de ce type. Plus particulièrement, S.A.G.A.C.E. tient compte :

- des faibles capacités de la mémoire humaine,
- de la difficulté qu'éprouvent les humains à utiliser le hasard,
- des particularités de l'apprentissage et de l'adaptation humains,
- de l'irrationalité humaine.

Comme le prix Nobel H.A. Simon, nous sommes convaincu qu'aussi bien dans la collecte des données que dans leur traitement les capacités intellectuelles des hommes sont limitées. Ainsi, l'Homme a une rationalité limitée qui, éventuellement, tend à maximiser ses choix, mais uniquement dans l'horizon des ses facultés d'analyse, de calcul et de compréhension.

S.A.G.A.C.E. est une approche modulaire : elle est constituée de différents modules qui peuvent être utilisés (implémentés) indépendamment les uns des autres. Cette modularité lui permet d'avoir des applications plus larges que celles du jeu à information complète et imparfaite. S.A.G.A.C.E. peut s'appliquer non seulement à d'autres types de jeux (information complète et parfaite, information incomplète) mais aussi à de nombreux problèmes d'interactions entre Homme et Machine. Cette généralité est notamment due au fait que la méthode prend en compte sa propre influence sur celle de l'humain qu'elle modélise et qu'elle permet, parfois, d'anticiper un comportement qui n'a jamais été observé précédemment (la plupart des autres méthodes décrites précédemment dans ce texte ne sont capables que de prédire un comportement déjà observé dans des circonstances semblables).

S.A.G.A.C.E. est fondée sur la modélisation des comportements des systèmes auxquels elle s'applique. Si elle est particulièrement adaptée aux interactions avec les humains, elle permet également d'appréhender les systèmes artificiels.

Elle repose sur une approche purement comportementale. Elle permet de prédire et d'anticiper le comportement d'un système (vivant ou artificiel) et surtout l'évolution de celui-ci au cours du temps. En revanche, elle ne prétend pas découvrir les mécanismes cognitifs ou algorithmiques sous-jacents.

Cette méthode algorithmique s'implémente sous forme de plusieurs systèmes de classeurs [Holland, 1986] ; [Wilson, 1989], dont les règles s'adaptent par apprentissage et par évolution génétique [Holland, 1975] ; [Goldberg, 1989] ; [Meyer, 1996a].

7.2 Les systèmes de classeurs

Sous sa forme la plus simple, un système de classeurs (S.C.) est constitué de quatre modules (Figure 7.1) qui communiquent entre eux et avec l'extérieur (généralement appelé l'Environnement) par l'intermédiaire de messages formatés.

7.2.1 Les différents modules d'un système de classeurs

Les quatre modules qui constituent un S.C. sont :

- Une base de règles ❶
- Une liste de messages ❷
- Une interface d'entrée ❸
- Une interface de sortie ❹

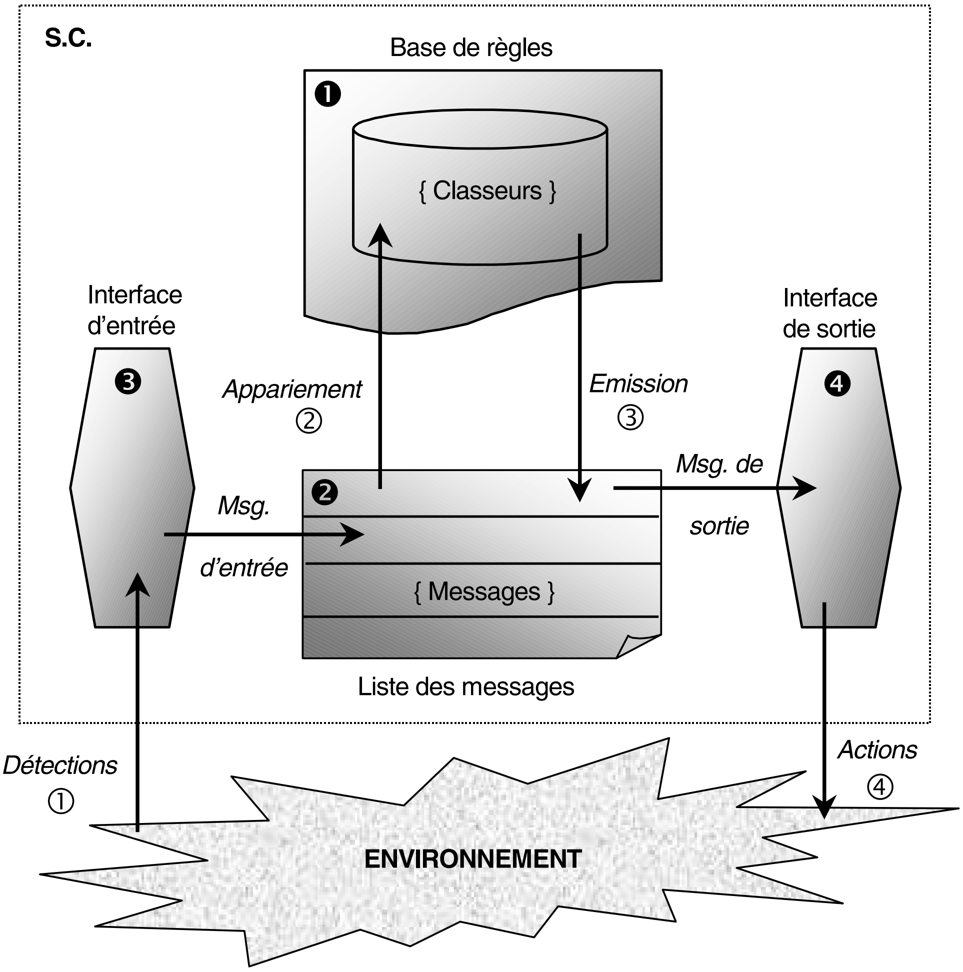


Figure 7.1. : Un système de classeurs

7.2.1.1 La base de règles

La base de règles contient des règles (les classeurs) de type Condition → Action de la forme :

$$C_i : \quad \textbf{SI } Sont\_présents\_messages(x_0, x_1, \ldots, x_n)$$

$$\textbf{ALORS } Emettre\_message(\ C_i(x_0, x_1, \ldots, x_n)).$$

Le classeur C_i est dit « déclenchable » si le message X_i est présent dans la liste des messages. Dans ce cas, le message $C_n(x)$ est déposé sur la liste des messages.

A chaque classeur est associée une valeur sélective (fitness) qui est une indication valuée de son intérêt par rapport aux autres. En effet, plusieurs classeurs peuvent se déclencher en même temps et chacun déposer un message sur la liste des messages ; dans le cas où certains de ces messages sont incompatibles, le S.C. arbitre en fonction de la fitness des classeurs correspondants.

7.2.1.2 La liste des messages

Les messages présents dans la liste des messages sont de trois types :

- Les messages d'entrée

  Ces messages sont construits par l'interface d'entrée à partir des informations sensorielles détectées dans l'environnement ①. Ces messages sont susceptibles de s'apparier ② avec la partie condition de certains classeurs qui pourront se déclencher en émettant ③ un (des) message(s) qui seront déposés dans la liste.

  Dans un contexte de jeu, les messages d'entrée sont, a priori, constitués à partir de l'état du jeu.

- Les messages internes

  Ces messages émis par les classeurs de la base de règles ont une fonction particulière : ils sont, comme leur nom l'indique, à usage interne. Ils servent à déclencher d'autres classeurs. Cet usage permet de déclencher des classeurs en série.

Dans un contexte de jeu, les messages internes peuvent être utilisés pour la planification stratégique, c'est-à-dire pour définir une stratégie à long terme. En effet, certains messages pouvant subsister dans la liste des messages pendant plusieurs coups (cf. Cycles d'un S.C.).

- Les messages de sortie

Ces messages sont émis par les classeurs de la base de règles. Ils sont à usage externe, codant les actions externes du S.C. ④.

Dans un contexte de jeu, les messages d'action représentent, a priori, les choix effectués par le système (les coups joués).

7.2.1.3 L'interface d'entrée

L'interface d'entrée représente la perception externe du S.C. Les informations détectées dans l'environnement ① sont formatées sous forme de messages d'entrée qui sont déposés sur la liste des messages.

7.2.1.4 L'interface de sortie

L'interface de sortie permet au S.C. de spécifier ses actions ④ externes en direction de l'environnement.

C'est l'interface de sortie qui arbitre, éventuellement, entre des messages d'actions contradictoires[1].

7.2.2 Cycles d'un système de classeur

Le fonctionnement d'un S.C. est géré sous forme de cycles. Chaque cycle comprend plusieurs étapes séquentielles :

1. gestion de l'interface d'entrée,
2. gestion de la liste de message,
3. gestion de l'interface de sortie.

---

[1] « Contradictoires » ne signifie pas « simultanées ». En effet, plusieurs messages d'action peuvent être émis simultanément s'ils ne sont pas fonctionnellement contradictoires. Par exemple, si le S.C. contrôle un robot, celui-ci pourrait parfaitement effectuer certaines actions en même temps (tourner la tête et avancer) , mais pas d'autres (tourner à droite et à gauche).

7.2.2.1 Gestion de l'interface d'entrée

Il s'agit principalement de la création des messages d'entrée en fonction des perceptions. C'est aussi durant cette étape que sont gérés les éventuels conflits entre perceptions incompatibles.

7.2.2.2 Gestion de la liste de message

Cette étape est souvent récursive. Tant qu'il existe un message qui peut s'apparier avec un classeur, l'appariement est effectué. Si de nouveaux messages sont émis, la gestion de la liste continue.

Certains S.C. intègrent une durée de vie pour les messages. Cette durée de vie est souvent comptée en nombre de cycles. Un message non apparié lors d'un cycle de fonctionnement du S.C. voit sa durée de vie diminuée d'une unité (un cycle). Lorsque la durée de vie d'un message en fin d'étape d'appariement devient nulle, le message correspondant est effacé[1].

Quand la liste ne contient plus que des messages d'action ou des messages internes qui ne peuvent s'apparier, la gestion de la liste est terminée.

7.2.2.3 Gestion de l'interface de sortie

Les messages de sortie présents dans la liste des messages sont gérés à cette étape.

Les actions effectuées dans l'environnement vont sans doute modifier ce dernier et sa perception par le système s'en trouvera donc modifiée, permettant d'engager un nouveau cycle de fonctionnement. L'environnement peut, par ailleurs, être dynamique et se trouver naturellement modifié d'un cycle du S.C. à un autre.

7.2.3 Apprentissage dans les systèmes de classeurs

L'apprentissage dans les systèmes de classeur peut se faire de différentes façons dont certaines peuvent être utilisées au sein d'un même système.

Les deux grandes catégories d'apprentissage sont :

1. l'adaptation de la fitness des classeurs,
2. la création de nouvelles règles adaptées au problème.

---

[1] Ainsi, un message peut subsister un certain nombre de cycles dans la liste des messages. Dans un contexte de jeu, un message (par exemple du type : « jouer défensif » ou « jouer telle ouverture ») peut être émis avec une certaine pérennité pour effectuer des choix stratégiques à long terme.

7.2.3.1 *L'adaptation de la fitness des classeurs*

Un classeur ayant émis un message de sortie s'étant avéré particulièrement adapté à la tâche du système peut être facilement « récompensé » par un système d'attribution positive de crédit (qui va augmenter sa fitness) ou « puni » par un système d'attribution négative de crédit (qui va diminuer sa fitness).

Il est beaucoup plus difficile de juger de l'efficacité des classeurs qui émettent des messages internes. En effet, la pertinence des messages émis ne pourra être évaluée qu'après que les actions induites aient été effectuées et jugées. Pour cette raison, différents algorithmes particuliers d'apprentissage par renforcement [Sutton, 1984 & 1998] ; [Barto, 1990], peuvent être mis en œuvre dans les systèmes de classeurs.

7.2.3.1.1 *Le Profit Sharing Plan (plan de partage des profits)*

Le Profit Sharing Plan (PSP) [Grefenstette, 1988] est un algorithme qui répartit directement le renforcement (positif ou négatif) perçu par les actions effectuées en fin de cycle sur tous les classeurs ayant été déclenchés pendant le cycle de fonctionnement.

Pour mettre en œuvre cet algorithme, il faut donner les moyens au S.C. d'adresser tous les classeurs qui ont participé à la réalisation de chaque action. Chaque appariement et déclenchement de classeur doivent donc être mémorisés pendant chaque cycle.

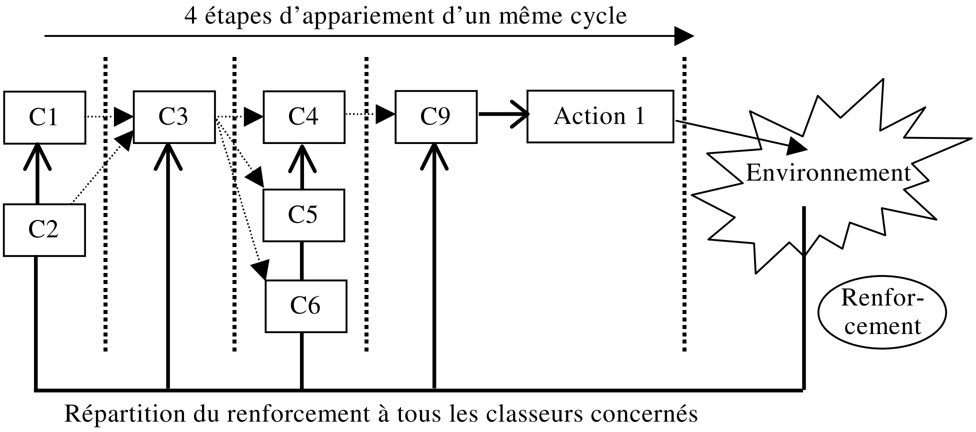


Figure 7.2. : Le profit Sharing Plan

Pendant les différentes étapes d'appariement d'un même cycle, plusieurs classeurs peuvent avoir été déclenchés simultanément et peuvent avoir, à leur tour, permis le déclenchement d'autres classeurs jusqu'au déclenchement d'un classeur ayant émis un message d'action ayant conduit à l'attribution d'un renforcement par l'environnement. La répartition du renforcement va impliquer tous les classeurs correspondants.

7.2.3.1.2 Le Bucket Brigade (Chaîne des porteurs d'eau)

L'Algorithme du Bucket Brigade (BBA) [Holland, 1986] permet d'anticiper le renforcement que recevra chaque classeur lorsque l'action déclenchée en fin de cycle aura été évaluée et aura reçu le renforcement correspondant de la part de l'environnement. Pour décrire l'algorithme, Holland utilise une analogie avec *les intermédiaires* en économie : ces personnes touchent une commission sur les affaires qu'ils apportent et paient une commission à leurs propres intermédiaires. Effectivement, le BBA fonctionne sur ce principe : chaque classeur paie un tribut (bid) immédiat aux classeurs qui lui ont permis d'être déclenché (il s'agit des classeurs dont le(s) message(s) émis sur la liste s'apparient à l'un des messages de la partie condition du classeur considéré). Le classeur ayant émis le(s) message(s) d'action en fin de cycle reçoit tout le renforcement correspondant.

Quand une suite d'activations des classeurs est répétée, le BBA est semblable au PSP. En effet, la chaîne de propagation du renforcement final étant répétée, le renforcement immédiat d'un classeur par un autre s'apparente à une propagation répartie en fin de cycle.

L'avantage majeur du BBA est que ses transferts de renforcement locaux ne nécessitent pas de mémoriser tous les classeurs déclenchés pendant chaque cycle de fonctionnement du S.C.

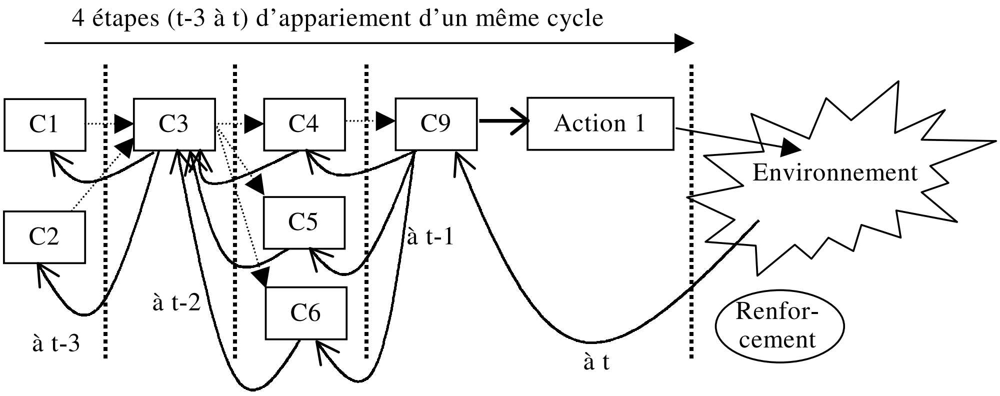


Figure 7.3. : Le Bucket Brigade

7.2.3.2 la création de nouvelles règles

Généralement, les nouveaux classeurs dans un S.C. sont créés par adaptation structurelle ou fonctionnelle des classeurs existants.

La méthode la plus souvent employée dans les systèmes de classeur pour générer de nouvelles règles procède par algorithme génétique [Goldberg, 1989], [Mitchell,1996] et [Michalewicz, 1996].

7.2.3.3 Les algorithmes génétiques

Les algorithmes génétiques sont des algorithmes d'exploration qui manipulent des populations de solutions potentielles à un problème donné. Génération après génération, les AG tentent d'optimiser les populations de solutions en manipulant les individus par des mécanismes qui s'apparentent à ceux de la sélection naturelle ou de la génétique.

L'analogie avec la sélection naturelle concerne le choix des solutions potentielles qui sont conservées d'une génération à l'autre. Entre chaque génération, un certain nombre d'individus (solutions potentielles) sont autorisé à se « reproduire » en produisant un certain nombre de descendants tandis que d'autres sont tout simplement éliminés. Dans la nature, la survie et le taux de reproduction d'un individu dépendent de son degré d'adaptation à son environnement. Pour les algorithmes génétiques, la « survie » d'un individu solution dépend de sa contribution à la résolution du problème général. L'intérêt de chaque individu est donné par une fonction dite d'évaluation (fitness) qui renseigne sur son degré d'adaptation. C'est en fonction de la fitness d'un individu qu'on détermine le nombre de ses descendants dans la génération suivante.

En règle générale, on détermine à l'avance le nombre N d'individus de la prochaine génération suivant la stratégie de reproduction qu'on a choisie (population de taille fixe, taux de croissance défini, etc.). Ensuite, on effectue N tirages aléatoires pour déterminer les individus sélectionnés dans la population précédente qui seront recopiés dans la nouvelle population. Les tirages sont biaisés en fonction de la fitness des individus de la population précédente. Un même individu peut-être sélectionné plusieurs fois et donc recopié plusieurs fois.

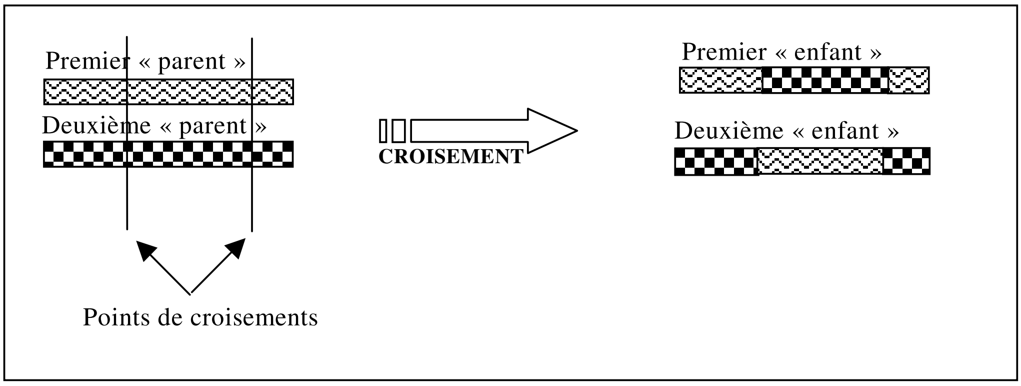
L'analogie avec la génétique concerne l'évolution des populations. Une fois la nouvelle génération établie (reproduction à l'identique en un ou plusieurs exemplaires des individus les meilleurs et élimination des plus mauvais), les individus sont susceptibles d'être modifiés par l'application d'opérateurs génétiques :

- le croisement,

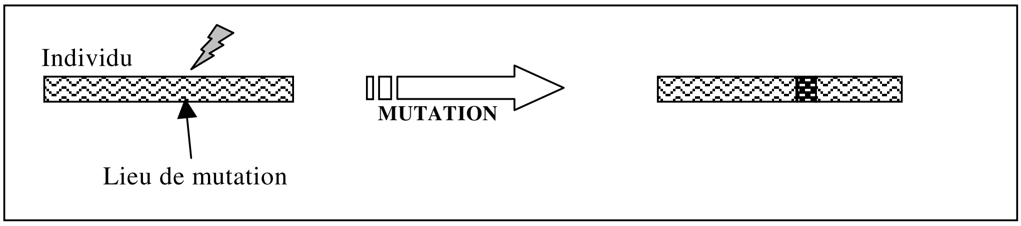
  Il consiste à mélanger le code génétique de deux individus pour en créer un troisième.
- la mutation.

  Elle consiste à modifier aléatoirement des parties du code génétique d'un individu.

Croisement :



Mutation :



Le choix des opérateurs génétiques (croisements et mutations) dépend du problème traité et plus particulièrement du codage des solutions. Les croisements et les mutations pour les chaînes binaires sont très simples : il s'agit, pour les croisements, d'échange de sous-chaînes et, pour les mutations, du remplacement d'un bit par un autre (0 ⇔ 1). Les opérateurs utilisés pour des individus codés sous forme réelle sont très différents [Belew, 1991], [Janikow, 1991].

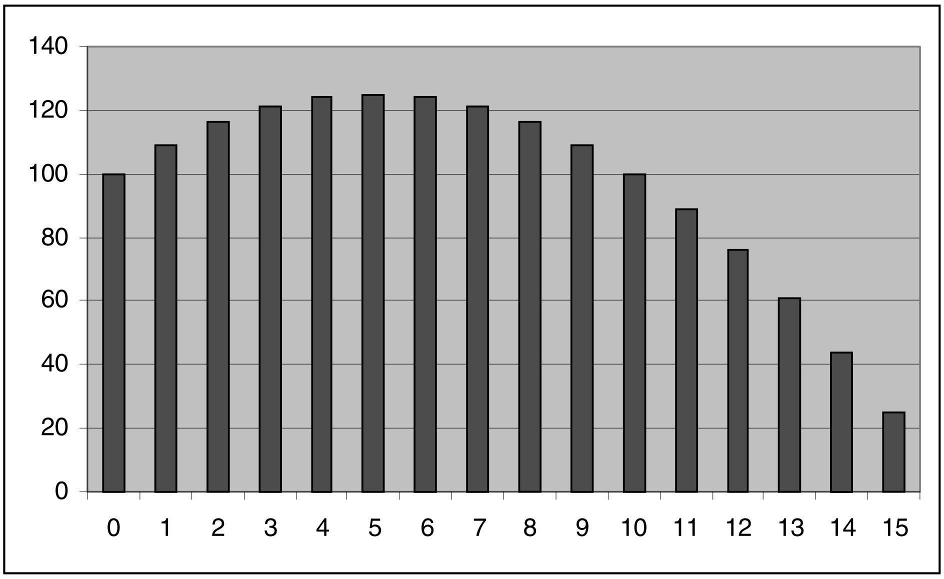
De la même façon, le taux de mutation dans la population, l'impact de chacune d'elles, le nombre de points de croisements, etc. sont autant de facteurs qui varient d'une application à une autre.

Exemple de mise en œuvre d'un AG :

Imaginons un système dont le but serait de maximiser par algorithme génétique la fonction entière :

$$f : [0..15] \Rightarrow N$$
$$f(i) = 100 - i(i - 10)$$

Comme code génétique des individus (solutions potentielles) nous prenons leur représentation binaire.



Evidemment, pour un exemple si simple, un rapide calcul ou un graphique permettrait de déterminer que la solution est 5. Cependant, dans cet exemple, on considère qu'on ne dispose que d'un moyen d'évaluer la fitness en chaque point.

Considérons une population de 4 solutions potentielles générées aléatoirement :

①	1 0 1 1
②	1 1 1 1
③	0 0 0 1
④	0 0 0 0

Comme fonction d'évaluation, on choisit naturellement f puisque l'on cherche à maximiser cette même fonction.

On détermine ainsi les fitness des 4 solutions potentielles :

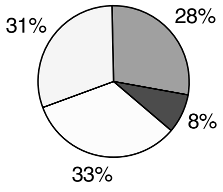
①	1 0 1 1	→ f(11) = 89
②	1 1 1 1	→ f(15) = 25
③	0 0 0 1	→ f(1) = 109
④	0 0 0 0	→ f(0) = 100

L'individu ③ est celui qui a la plus forte fitness, c'est celui qui a le plus de chances d'être reproduit à la génération suivante. Réciproquement, l'individu ② est le moins efficace et aura moins de chances d'être reproduit.

On effectue les tirages aléatoires avec la répartition probabiliste suivante :

①	1 0 1 1	→ f(11) = 89	→ reproduction = 89/(Σf) =	28 %
②	1 1 1 1	→ f(15) = 25	→ reproduction = 25/(Σf) =	8 %
③	0 0 0 1	→ f(1) = 109	→ reproduction = 109/(Σf) =	33 %
④	0 0 0 0	→ f(0) = 100	→ reproduction = 100/(Σf) =	31 %

Σf est la somme des fitness des individus de la population. Dans cet exemple, Σf vaut 323 (89+25+109+100). Cette méthode de reproduction aléatoire est souvent appelée méthode de la roulette étant donnée l'analogie avec une roue de loterie…



Imaginons que les tirages aient désigné les individus ①, ③, ③ (une $2^{\text{ème}}$ fois) et ④.

Si les croisements concernent les individus deux par deux (①&③ et ③&④) et s'il n'y a qu'un point de croisement au centre des allèles, on obtient la population suivante :

❶ 1 0 | 0 1 = 1 0 0 1 ①&③ premier « enfant »
❷ 0 0 | 1 1 = 0 0 1 1 ①&③ deuxième « enfant »
❸ 0 0 | 0 0 = 0 0 0 0 ③&④ premier « enfant »
❹ 0 0 | 0 1 = 0 0 0 1 ③&④ deuxième « enfant »

Il se trouve que les nouveaux individus ❸ et ❹ sont les mêmes que les précédents individus ③ et ④. C'est un hasard…

Imaginons maintenant qu'une mutation intervienne sur l'individu ❹ au niveau du $2^{ème}$ bit. ❹ devient se transforme alors : 0 0 0 1 → 0 1 0 1.

La population finale est la suivante :

❶ 1 0 0 1 ①&③ premier « enfant » → f(9) = 109
❷ 0 0 1 1 ①&③ deuxième « enfant » → f(3) = 121
❸ 0 0 0 0 ③&④ premier « enfant » → f(0) = 100
❹ 0 **1** 0 1 ③&④ deuxième « enfant » muté. → f(5) = 125

Globalement, la population s'est améliorée ($\Sigma f = 455$). De plus, l'individu codant la solution optimale est apparu. En effet, l'individu ❹ est la solution au problème posé. Sa fitness est de 125.

Dans un problème réel, la solution n'est pas nécessairement identifiable. En effet, sans calculs préalables, rien ne permettait d'affirmer que ❹ était l'individu solution. C'est pourquoi, en général, on interrompt l'algorithme génétique quand la population ne s'améliore plus depuis un certain temps et on considère que son meilleur individu est une bonne approximation de la solution au problème (sinon la solution elle-même).

7.3 Les jeux utilisés

Pour illustrer la solution S.A.G.A.C.E. et les différentes méthodes originales qui la composent, nous utiliserons les jeux à information complète et imparfaite qui ont fait l'objet d'une implémentation. Ces mêmes jeux nous permettront de comparer l'approche S.A.G.A.C.E. à d'autres, au moyen de tableaux comparatifs de performance.

Chacun de ces jeux concerne deux joueurs, est symétrique à somme nulle, et est itéré (on effectue plusieurs parties et le vainqueur est celui qui remporte le plus de victoires).

7.3.1 Pair / Impair (ou « matching pennies »)

7.3.1.1 Marche du jeu

Les deux joueurs (notés J1 et J2) choisissent pour chaque coup entre « pair » et « impair ». Si les deux joueurs ont choisi la même chose, le joueur J1 marque un point, si les deux joueurs ont fait un choix différent, c'est le joueur J2 qui marque un point.

7.3.1.2 Particularités de ce jeu

Il s'agit du jeu à information complète et imparfaite le plus simple qu'on puisse imaginer. Il a inspiré Shannon qui a réalisé une machine mécanique capable d'y jouer contre un humain [Shannon, 1953]. L'approche S.A.G.A.C.E. sera comparée à celle de Shannon.

7.3.2 Pierre / Ciseaux / Papier

7.3.2.1 Marche du jeu

Les deux joueurs choisissent simultanément et en secret entre *Pierre*, *Ciseaux* ou *Papier*. Si les deux joueurs ont choisi la même chose, le coup est nul, sinon, un joueur ayant choisi *Papier* gagne contre un joueur ayant choisi *Pierre* (le papier enveloppe la pierre) et perd contre un joueur ayant joué *Ciseaux* (les ciseaux coupent le papier). Un joueur ayant choisi *Pierre* gagne contre un joueur ayant choisi *Ciseaux* (la pierre casse les ciseaux).

7.3.2.2 Particularités de ce jeu

Ce jeu est extrêmement simple. Minasi l'a utilisé pour illustrer sa méthode d'élaboration de stratégie par recherche de patterns [Minasi, 1991]. L'approche S.A.G.A.C.E. sera comparée à celle de Minasi.

7.3.3 ALESIA[1]

7.3.3.1 Marche du jeu

Chaque joueur reçoit au départ un capital de 50 points. Chaque coup du jeu consiste en un combat de points. Chaque joueur décide secrètement du nombre de points (au moins un) qu'il affecte à ce coup, et l'inscrit sur un papier. Puis, les papiers sont découverts. Le joueur qui a écrit le plus grand nombre bouge le marqueur (initialement au centre) d'un pas (une ligne verticale) vers la citadelle adverse. Si les deux nombres sont égaux, le marqueur n'est pas bougé. Les nombres écrits par les joueurs sont ensuite retranchés de leurs capitaux respectifs.

Lorsqu'un joueur seul a épuisé son capital, la partie continue et il est forcé de jouer le nombre zéro à chaque tour, perdant donc contre tout choix de son adversaire. En pratique, le joueur adverse pourra donc avancer le marqueur d'autant de pas qu'il lui reste de points (en jouant un point à chaque coup).

7.3.3.2 Fin de la partie et gain

Un joueur gagne s'il parvient à déplacer le marqueur jusqu'à la citadelle adverse (située à trois pas de la ligne centrale de départ).

Si les capitaux des deux joueurs sont épuisés avant qu'aucun n'ait réussi à atteindre la citadelle adverse, la partie est nulle.

7.3.3.3 Particularités de ce jeu

Les règles de ce jeu en font un sujet d'étude particulièrement riche :

1) Les règles sont simplissimes mais la combinatoire immense,

2) Il n'existe pas de stratégie pure générale[2] garantissant la victoire,

3) Il n'existe pas de stratégie pure générale garantissant le nul,

4) certaines positions particulières offrent des stratégies pures optimales.

On trouvera en Annexe C une démonstration du fait que ce jeu n'admet pas de stratégie optimale non mixte.

---

[1] Jeu librement inspiré du jeu « CITADELLE » [Pingaud, 1984].

[2] qui s'applique quelle que soit la configuration du jeu.

7.3.4 Le jeu des trois pierres[1]

7.3.4.1 Marche du jeu

Deux joueurs s'affrontent. Ils possèdent chacun trois pierres au début du jeu. L'un des joueurs est désigné premier joueur. A chaque tour de jeu, les deux joueurs choisissent simultanément et en secret un nombre de pierres compris entre zéro et le nombre de pierre qu'il leur reste (donc trois au départ). Ensuite, le premier joueur annonce un nombre correspondant à ce qu'il pense être la somme des deux nombres de pierres choisis par les joueurs. Le deuxième joueur annonce alors également un nombre correspondant à ce qu'il pense être la somme mais ce nombre doit être différent de celui du premier joueur.

Les deux joueurs dévoilent alors le nombre de pierres qu'ils avaient choisi (sans mensonge).

Si le premier joueur a annoncé la somme correcte, il gagne et peut écarter définitivement une de ses pierres.

Si le second joueur a annoncé la somme exacte, c'est lui qui gagne et qui peut écarter une de ses pierres.

Si un des deux joueurs a gagné le tour, il sera le premier joueur pour le tour suivant.

Si les deux joueurs ont échoué, un autre tour commence mais le second joueur devient le premier et inversement.

7.3.4.2 Fin de la partie et gain

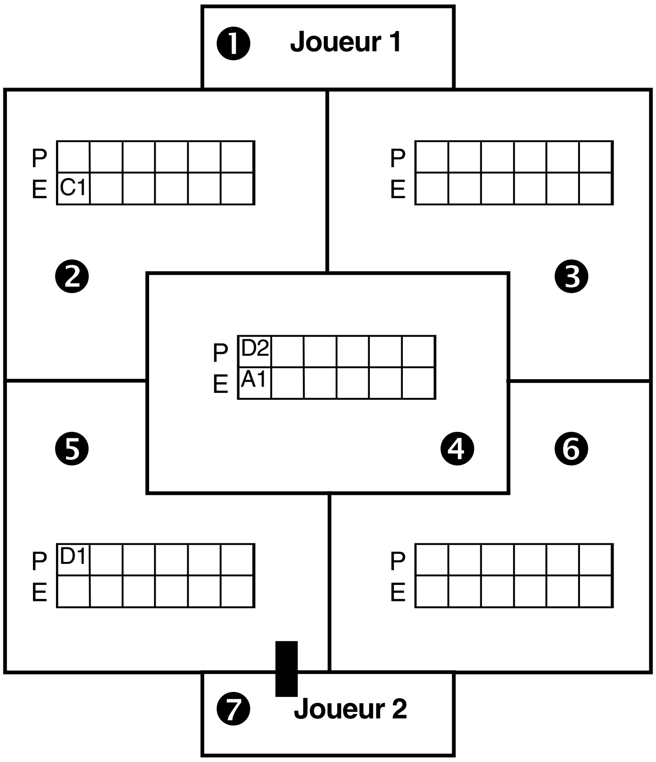
Le vainqueur est celui qui parvient à se débarrasser de toutes ses pierres.

---

[1] ce jeu m'a été décrit par le romancier B. Werber avec lequel j'ai eu plaisir à y jouer.

7.3.5 SUNTZU[1]

SUNTZU est un jeu économico-politique qui opposent deux joueurs.



Premier tour de jeu :

Joueur 1 annonce :
Dictature sur le pays ❺

Joueur 2 annonce :
Chemin de fer vers le pays ❺

Joueur 1 annonce :
Charbon sur le pays ❷

Joueur 2 annonce :
République sur le pays ❺
(instruction inapplicable)

Joueur 1 annonce :
Argent sur le pays ❹

Joueur 2 annonce :
Dictature sur le pays ❹

Le joueur 1 ne marque rien,
Le joueur 2 marque 1 point…

7.3.5.1 Marche du jeu pour deux joueurs :

La partie se joue en 8 tours de jeu. A chaque tour, les deux joueurs inscrivent secrètement 3 instructions sur une feuille personnelle dans un ordre bien clair. Les instructions sont ensuite annoncées en alternance : le joueur 1 annonce sa première instruction, qui est exécutée sur la feuille commune de jeu (dessinée ci-dessus) ; puis le joueur 2 annonce sa première instruction qui est exécutée ; puis le joueur 1 annonce sa deuxième instruction etc.

Au tour suivant, ce sera le joueur 2 qui commencera à annoncer (et en alternance ainsi à chaque tour).

---

[1] Jeu librement inspiré du jeu « PAYS » [Pingaud, 1984].

7.3.5.1.1 Les instructions sont de trois types :

- Augmentation de la richesse économique d'un pays : le joueur indique un pays et une richesse économique, qui est inscrite immédiatement de la couleur du joueur et dans une des cases correspondantes sur le pays (2ème ligne du tableau des pays). Il y a trois richesses possibles : argent (A), blé (B), charbon (C). Pour être inscrite, la richesse indiquée doit être différente de la richesse précédemment inscrite dans le même pays ; sinon, l'instruction n'est pas exécutée (et elle n'est pas remplacée).

- Construction d'un chemin de fer : le joueur indique entre quels pays est construite la ligne qui est alors tracée sur la feuille. Une ligne se place sur la frontière entre deux pays y compris ceux des deux joueurs. Il ne peut y avoir qu'une ligne par frontière commune à deux pays donnés ; si elle existe déjà, l'instruction n'est pas exécutée (et n'est pas remplacée).

- Modification du régime politique d'un pays : le joueur indique un pays et un régime politique qui est inscrit immédiatement sur la feuille de la couleur du joueur et sur une case correspondante du pays (1ère ligne du tableau). Il y a trois régimes possibles qui se suivent dans un ordre circulaire strict : dictature, guérilla, république, dictature, etc. A une dictature ne peut succéder qu'une guérilla, à celle-ci seulement une république... Le premier régime politique imposé à un pays peut être n'importe lequel. Par la suite, une instruction qui n'est pas conforme à la règle de succession n'est pas exécutée (et n'est pas remplacée).

Les instructions de richesses économiques et de régime politique ne peuvent que concerner les pays centraux (Numéro 2 à 6) ; les pays des deux joueurs n'y participent pas.

Les trois instructions d'un joueur pour un tour ne doivent pas mentionner deux fois le même pays, quelles que soient ces instructions.

Un pays sans régime politique (pas encore affecté par un des deux joueurs) ne compte pas lors du décompte des points à la fin de chaque tour de jeu.

7.3.5.2 Fin de la partie et gain :

Après chaque tour de jeu, le score des joueurs est calculé de la façon suivante : Chaque richesse vaut un point sauf dans un pays en guérilla où les richesses ne sont pas comptées. Dans un pays en dictature, seuls sont comptés l'argent et le charbon, et seulement au joueur qui a instauré la dictature (même si ce n'est pas lui qui a déposé ces richesses). Dans un pays en république, seuls sont comptés le blé et le charbon pour les deux joueurs (chaque joueur marque un point par richesse de sa couleur). Le charbon, quel que soit le régime politique, n'est compté pour un joueur que s'il peut être transporté par chemin de fer depuis le pays où il est inscrit jusqu'à celui du joueur ; il doit exister une succession de lignes de chemin de fer qui relie les deux pays, sans passer par un pays en guérilla ou un pays en dictature instituée par l'autre joueur.

Les scores obtenus par les joueurs sont additionnés. A la fin de la partie, le plus fort total gagne.

7.3.5.3 Particularités de ce jeu

L'intérêt principal de ce jeu réside dans :

1) le fait que les joueurs effectuent trois choix simultanément,

2) le fait qu'on puisse marquer des points en utilisant des ressources posées par l'adversaire,

3) le fait qu'on puisse changer un régime politique en spéculant sur un changement causé dans le même tour par l'adversaire,

4) le fait que l'ordre dans lequel on effectue (ou place) ses choix est déterminant.

7.4 Architecture générale de S.A.G.A.C.E.

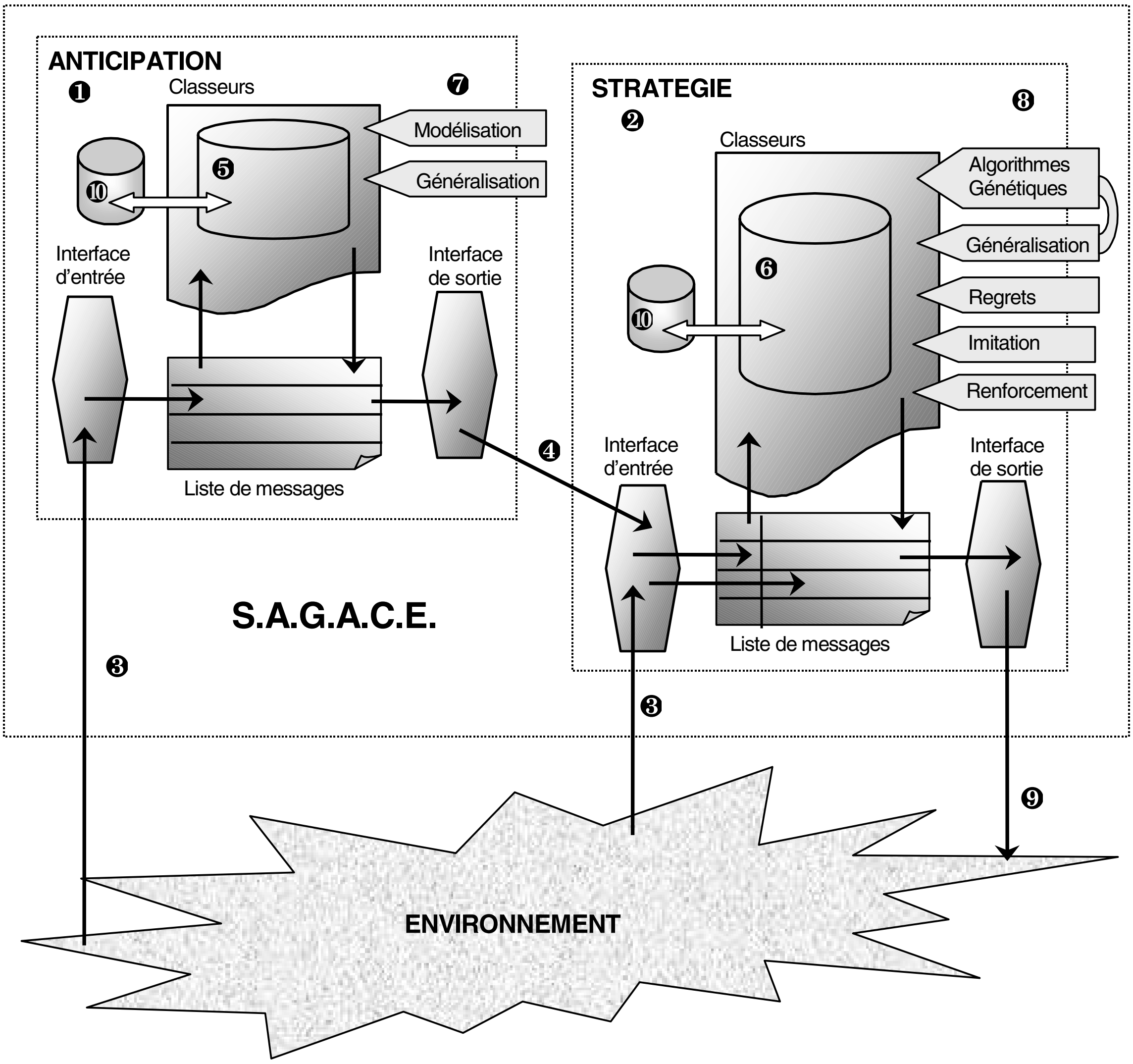


Figure 7.4. : Architecture générale de S.A.G.A.C.E.

S.A.G.A.C.E. se compose principalement de deux systèmes de classeurs indépendants qui communiquent par l'intermédiaire de l'interface de sortie de l'un et l'interface d'entrée de l'autre.

Le premier système de classeurs ❶ permet la modélisation de l'adversaire et l'anticipation de ses actions. Le deuxième système ❷ de classeurs est affecté aux choix stratégiques en fonction de l'état du jeu et du modèle généré par le premier système.

Les deux systèmes reçoivent des messages de l'environnement qui concernent l'état général du jeu ❸ (la situation actuelle).

Le premier système renseigne le second sur le modèle dynamique de l'adversaire (qu'il actualise en temps réel) par l'intermédiaire de son interface de sortie ❹.

Les classeurs du système de modélisation/anticipation ❺ sont des règles de type :

SI la_situation_est(S_i) **ALORS** l'adversaire_va_jouer(J_j)

A ces classeurs sont associés, parmi d'autres, des paramètres de confiance dans la conclusion donnée, une fitness correspondant à la valeur du coup correspondant pour l'adversaire, et des indices particuliers qui seront explicités ultérieurement.

Ces classeurs sont générés puis évoluent par apprentissage et par la mise en œuvre de techniques originales de modélisation et de généralisation ❼.

Les classeurs du système de stratégie ❻ sont des règles de type :

SI la_situation_est(S_i) **ET SI** l'adversaire_va_jouer(Jj) **ALORS** jouer(A_k)

La partie « l'adversaire_va_jouer(Jj) » peut-être laissée vide, signifiant que le classeur n'est pas concerné par les prédictions ou estimations du S.C. d'anticipation.

A chacun de ces classeurs est associée une fitness indiquant l'intérêt estimé de ce dernier.

Ces classeurs sont générés puis évoluent par apprentissage par renforcement, par évolution génétique, et par différentes méthodes originales ❽.

Les classeurs du S.C. stratégique émettent des messages de sortie qui représentent les choix du système (le(s) coup(s) à jouer) ❾.

A chaque S.C. sont associées des bases de métaconnaissance [Pitrat, 1990] qui régissent l'évolution et l'adaptation des classeurs ❿. Elles contiennent notamment les règles de paramétrage des méthodes d'adaptation utilisées (modélisation, généralisation, imitation, génétique, renforcement, etc.).

L'originalité du travail présenté réside principalement dans l'architecture générale de S.A.G.A.C.E. et dans les différentes méthodes originales de création et d'adaptation de classeurs introduites dans les deux systèmes de classeurs.

Chacun des modules, chacune des méthodes d'apprentissage, de création de classeurs, de modélisation, de généralisation est présenté(e) en détails dans les sections suivantes.

7.5 Implémentation de S.A.G.A.C.E.

Les sections suivantes vont détailler l'implémentation de S.A.G.A.C.E. pour des jeux à information complète et imparfaite.

7.5.1 Les bases de règles[1] du S.C. Stratégique

7.5.1.1 Forme des classeurs

S.A.G.A.C.E. va manipuler des classeurs lui permettant de choisir une action en fonction de la configuration du jeu.

Ces classeurs sont génériquement de la forme suivante :

SI la_situation_est(S_i) **ET SI** l'adversaire_va_jouer(Jj) **ALORS** jouer(A_k)

Il est important que la forme implémentée des classeurs du S.C. stratégique permette de décrire toutes les stratégies pures du jeu.

Plus simplement, il est impératif que le type de la partie *condition* des classeurs (la_situation_est(S_i)) permette de décrire toute situation possible du jeu et que la partie *conclusion* des classeurs (jouer(A_k)) permette de définir tous les choix de jeu possibles.

7.5.1.1.1 Exemple d'ALESIA

Une situation à ALESIA est entièrement définie par :

- *Le nombre de points d'un joueur,*
- *Le nombre de points de son adversaire,*
- *La position du jeton.*

Les actions des joueurs concernent

- *le nombre de points misés*

---

[1] Nous utiliserons désormais indifféremment le terme « règle » et le terme « classeur ».

Ainsi, la forme minimale des classeurs pour ALESIA est la suivante :

Condition			***Action***
Nombre points	*Nombre points de l'adversaire*	*Position jeton*	*Mise*
[1..50]	*[1..50]*	*[-3..3]*	*[1..50]*

Pour tenir compte du coup anticipé de l'adversaire, une condition supplémentaire est ajoutée à la forme minimale précédente :

Condition			***Prédiction***	***Action***
Nb points	*Nb points de l'adv.*	*Position jeton*	*Mise de l'adv.*	*Mise*
[1..50]	*[1..50]*	*[-3..3]*	*[1..50]*	*[1..50]*

7.5.1.1.2 Exemple de Pierre / Ciseaux / Papier

Si le jeu n'était joué qu'une seule fois, il n'y aurait pas de partie condition pour les classeurs. En effet, la notion de situation ne correspond à rien dans un jeu à coup unique.

En revanche, si on considère qu'on va effectuer un certain nombre de parties, ce qui s'est passé lors des parties précédentes (choix des deux joueurs) crée la situation en cours. En fait, les joueurs vont agir en fonction des coups précédents. La forme des classeurs peut alors intégrer, par exemple, les choix effectués par les deux joueurs lors des trois derniers coups.

Remarque : Convention d'écriture : Pierre = R, Ciseaux = S, Papier = P.

On obtient alors des classeurs de la forme :

Condition						***Prédiction***	***Action***
Coup i-3		*Coup i-2*		*Coup i-1*		*Choix adversaire*	*Choix*
Soi	*Adv.*	*Soi*	*Adv.*	*Soi*	*Adv.*		
R/S/P	*R/S/P*	*R/S/P*	*R/S/P*	*R/S/P*	*R/S/P*	*R/S/P*	*R/S/P*

Dès la conception, il est particulièrement indiqué de tenir compte de la nature des actions qui seront effectuées les classeurs :

- Généralisation
- Spécification
- Simplification
- Etc.

Pour ce faire, il faut qu'un même classeur puisse s'appliquer dans plusieurs situations, et donc que certains paramètres de sa partie condition ou d'anticipation puisse être moins contraints que d'être assignés à une valeur fixe. Pour les paramètres numériques, on utilise naturellement des intervalles et, pour des éléments spécifiés, on utilise des ensembles.

7.5.1.1.3 Exemple d'ALESIA

Les classeurs dans ALESIA pourraient donc prendre la forme suivante :

Condition						***Prédiction***		***Action***
Nb points		*Nb points de l'adv.*		*Position jeton*		*Mise de l'adv.*		*Mise*
Inf	*Sup*	*Inf*	*Sup*	*Inf*	*Sup*	*Inf*	*Sup*	
[1..50]	*[1..50]*	*[1..50]*	*[1..50]*	*[-3..3]*	*[-3..3]*	*[1..50]*	*[1..50]*	*[1..50]*

Cela permet d'écrire des règles de la forme :

***SI** (il me reste entre 3 et 10 points) & (il reste entre 10 et 15 points à l'adversaire) & (le jeton est au milieu ou à une case du milieu)*

***ET SI** (je pense que l'adversaire va jouer entre 9 et 11 points)*

***ALORS** (jouer 1 point).*

Une fois la forme des classeurs déterminée, il reste à définir la forme de l'enveloppe des classeurs. L'enveloppe est le classeur lui-même accompagné de tous les paramètres qui permettent de lui attribuer une valeur sélective (pour arbitrer entre les classeurs à considérer lors d'un choix stratégique) et de tous les paramètres de gestion des bases (date de création, provenance, etc.). La forme de l'enveloppe est conditionnée par le choix des méthodes d'apprentissage, de gestion, d'élimination, etc.

Certains exemples d'enveloppes seront tirés des implémentations de S.A.G.A.C.E.

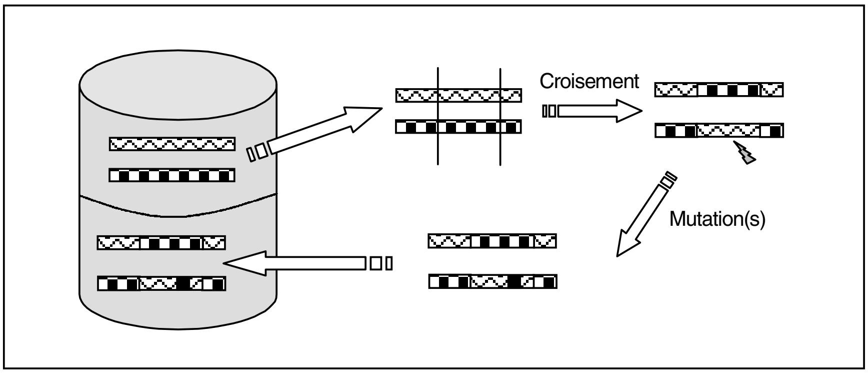
Remarque: on trouvera la forme des règles de SUNTZU et un exemple en Annexe B.

7.5.1.2 Méthodes de création de règles

S.A.G.A.C.E. utilise plusieurs méthodes de création de règles :

- Les algorithmes génétiques ;
- L'introspection ou l'amorçage par modélisation d'un joueur humain performant ;
- La généralisation ;
- L'imitation ;
- La méthode originale des regrets.

7.5.1.2.1 Les algorithmes génétiques (AG)



Nous avons déjà présenté la méthodologie générale des algorithmes génétiques. Cependant, nous avons développé pour S.A.G.A.C.E. une méthode paramétrable de création de règles par algorithmes génétiques.

De nombreux moyens permettent d'ajuster la création de règles par AG, particulièrement :

- Croisements
  - Choix des parents
  - Nombre de points de croisement
  - Fitness des enfants
- Mutations
  - Taux de mutation
  - Forme des mutations

Dans S.A.G.A.C.E., tous ces moyens sont paramétrables. Les paramètres correspondants sont contenus dans les bases de métaconnaissances liées au S.C. stratégique.

L'ajustement des différents paramètres de l'A.G. de création des règles du S.C. stratégique est à la charge d'un second A.G. !

Dans les implémentations que nous avons faites de S.A.G.A.C.E., cet A.G. d'ajustement est extrêmement simplifié. Il permet en fait de tester sur un très grand nombre de parties (ce qui le rend difficilement exploitable en vérité) des affectations de valeurs aux paramètres.

Exemple d'ALESIA

Pour l'implémentation d'ALESIA, nous avons défini les paramètres suivants :

- *Choix des parents :*

  parmi les 1%, 5%, 10%, 25%, 50%, 75% ou 100% meilleurs individus

  Ce paramètre détermine comment sont choisis les parents. 1% signifie qu'on ne prend que les meilleurs individus, 100% signifie que tous peuvent être parents. Dans l'implémentation, nous avons séparé en deux ce facteur (un pour chacun des deux parents) ce qui permettait d'effectuer des croisements avec, par exemple, toujours au moins un des parents parmi les 1% meilleurs et l'autre parmi les 50% meilleurs.

- *Nombre de points de croisement :*

  1, 2, 3, 4, 5 ou aléatoire.

  Ce paramètre est simple, il définit simplement en combien de points vont s'effectuer les échanges de valeurs entre deux individus.

- *Pourcentage de règles mutées*

  Définit, pour chaque génération, le pourcentage de règles créées qui seront mutées.

- *Pourcentage de mutations dans une règle mutée*

  Définit, pour chaque règle mutée le pourcentage de parties mutées.

- *Importance de la variation des mutations*

  10%, 25%, 50% ou total

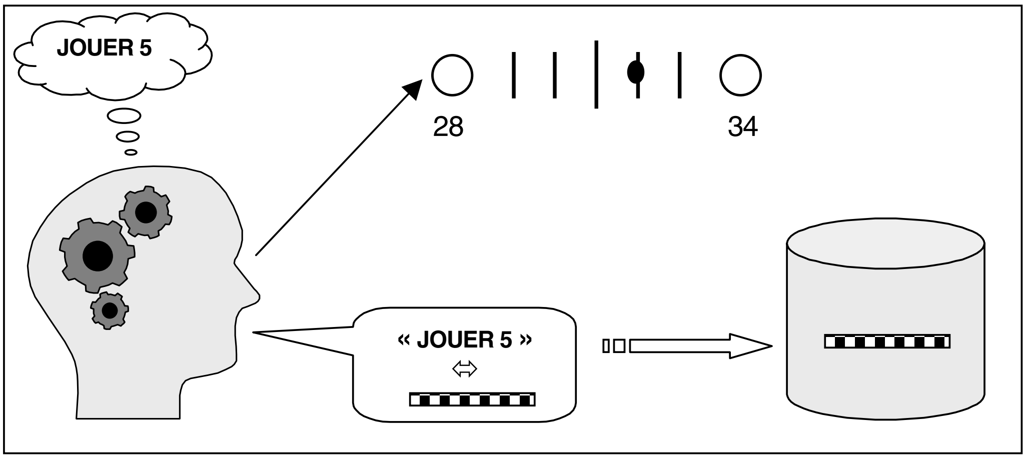
  Définit, pour chaque mutation l'importance de la mutation, c'est-à-dire à quel point elle modifie la valeur mutée. Les règles dans ALESIA ne contiennent que des entiers (hors enveloppe) et cette importance de variation correspond à une distance maximum (un intervalle centré autour de la valeur avant mutation) entre l'ancienne valeur et la nouvelle.

  L'introduction de ce facteur permet de créer des règles à la manière des méthodes d'exploration par « Hill climbing » (qui effectuent des petits sauts d'exploration avec uniquement de petites variations par mutation).

L'algorithme génétique de paramétrage utilisé pour l'implémentation d'ALESIA ne fait que déterminer de nouvelles séries de paramètres par recombinaison de séries anciennes.

Un sérieux travail devrait permettre de généraliser cette approche pour en faire une méthode générique de S.A.G.A.C.E.

7.5.1.2.2 L'Introspection



Cette méthode est sans doute la plus simple à décrire mais la plus difficile à mettre en œuvre. Elle consiste simplement, pour un humain, à décrire des règles qui lui semblent judicieuses, sous le formalisme imposé par les règles.

Cette idée d'amorçer un système par la modélisation d'un joueur humain a conditionné une grande partie de l'approche S.A.G.A.C.E. C'est, en effet, pour réaliser un tel amorçage qu'a été développée l'idée de modélisation par un S.C. dédié qui fait l'originalité de S.A.G.A.C.E. [Meyer, 1996b].

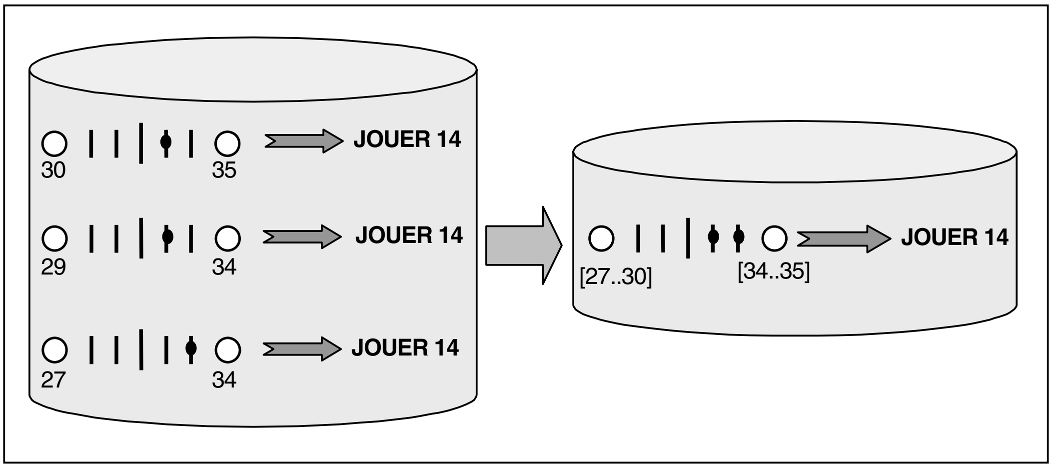
Exemple d'ALESIA

Lors de l'implémentation d'ALESIA, nous avons introduit de cette façon (c'est-à-dire manuellement) un petit nombre de règles nous semblant judicieuses. Par exemple :

« Au départ, jouer 10 » ou « si je suis à un pas de la victoire et si j'ai plus de points que l'adversaire, jouer tout », etc.

Certaines de ces règles se sont avérées très efficaces, d'autres très mauvaises (elles ont été éliminées par le système après apprentissage).

7.5.1.2.3 La généralisation



Dans S.A.G.A.C.E. , la généralisation est utilisée, au sein du S.C. stratégique, pour en améliorer les performances globales en généralisant les bonnes règles (les bons classeurs) trop spécifiques.

Pour généraliser des règles telles que celles utilisées dans les implémentations que nous avons faites (ALESIA, SUNTZU, etc.) il ne suffit pas simplement de regrouper des règles ayant la même conclusion et des parties conditions similaires pour en faire une seule. En effet, se pose le problème de la valeur sélective (fitness) des règles. Même si deux règles sont très proches du point de vue de leur forme, elles peuvent avoir des performances très différentes qui rendraient illogique le fait de les combiner.

En fait, le problème se pose en ces termes :

Partant d'une règle stratégique performante mais peu utilisée par le système, comment la rendre plus souvent applicable sans lui faire perdre de son efficacité ? (si elle est efficace et peu utilisée, cela signifie qu'elle n'est pas souvent applicable).

Une règle performante et peu applicable tire bien souvent sa force de sa spécificité. De ce fait, on comprend à quel point le problème de la généralisation est délicat parce que, en la généralisant, on risque de faire perdre à une règle ce qui la rendait efficace.

La méthode originale que nous avons développée pour S.A.G.A.C.E. est la suivante :

Etant donnée une règle R à généraliser, on construit un arbre de recherche de performance associé R. La racine est constituée de R et chaque nœud d'une généralisation de R. Dans les implémentations de S.A.G.A.C.E. que nous avons faites, les règles sont définies à partir d'entiers et de booléens, mais la méthode est facilement généralisable. Pour généraliser sur un intervalle, on en augmente la taille (on éloigne les bornes inférieures et supérieures) ; pour généraliser sur un booléen, on le transforme en un joker (indifféremment Vrai ou Faux).

La stratégie de généralisation (choix des parties des règles à généraliser) se fait suivant des heuristiques contenus dans les bases de Métaconnaissances telles que :

- Commencer par généraliser les parties les plus générales (par exemple, l'intervalle le plus grand) car, a priori, cela préserve la spécificité de la règle.

La construction de l'arbre obéit à d'autres heuristiques de la forme :

- Si, après tests[1], une règle généralisée s'avère moins efficace qu'elle ne l'était auparavant, remonter d'un nœud et couper définitivement la branche correspondante.

Avec cette méthode, il est possible de couper une branche qui aurait pu, après quelques étapes supplémentaires, s'avérer très fructueuse. En effet, une règle généralisée à un nœud précis peut avoir une efficacité inférieure à celle de R mais une de ses descendantes éventuelles peut être, elle, plus efficace.

En fait, une simple observation du fonctionnement de la méthode montre que le fait de couper une branche n'empêche pas de retrouver certaines de ses feuilles par un autre chemin (figure 7.5.).

---

[1] Nous avons également créé une méthode originale permettant de tester très rapidement un grand nombre de règles. Cette méthode de génération de situation sera décrite ultérieurement (§7.5.4.).

Exemple de généralisation :

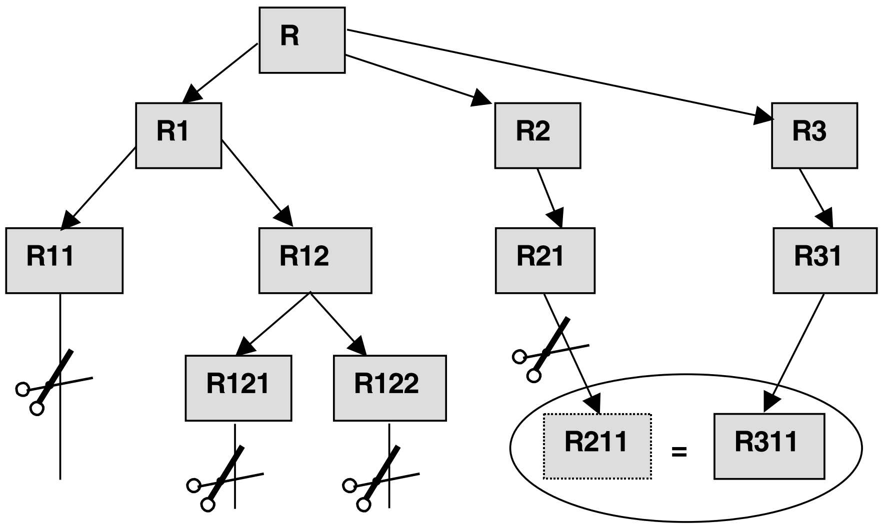


Figure 7.5. : Généralisation dans S.A.G.A.C.E.

- R1 : 1er essai
- R11 : 2ème essai ; échec, retour à R1
- R12 : 3ème essai ; surgénéralisation
- R121 : 4ème essai ; échec ; retour en R12
- R122 : 5ème essai ; échec ; retour en R12 puis en R1 puis en R
- R2 : 6ème essai
- R21 : 7ème essai ; échec ; retour en R2 puis en R

  Une autre généralisation de R21 aurait donné R211
- R3 : 8ème essai
- R31 : 9ème essai
- R311 : 10ème essai

  La règle R211 n'a pas été atteinte parce que la branche R2 a été coupée lorsque que R21 s'est révélée moins efficace que R. Cependant, l'exploration de la partie droite de l'arbre permet de tester R311 qui, parce qu'elle s'avère être identique à R211, permet de retrouver cette dernière.

Cette méthode de généralisation, si elle peut permettre la création de règles très efficaces (comme nous l'avons parfois observé) a toutefois des inconvénients : elle nécessite beaucoup de temps en calculs et beaucoup d'espace mémoire. En particulier, elle demande d'autant plus de temps qu'il faut tester chaque règle sur une série d'exemples afin d'estimer son efficacité. Pour cette raison, elle n'a été utilisée en pratique que lorsque le système ne jouait pas (longues procédures d'essais pendant la nuit).

On trouvera en annexe B un exemple de généralisation d'une règle pour le jeu SUNTZU.

Cette méthode ne s'oppose pas complètement au point de vue de Holland concernant la généralisation dans les systèmes de classeurs [Holland, 1986]. Ce dernier explique qu'il n'est pas utile d'ajouter des procédures de généralisation ou de spécialisation dans un système de classeurs qui utilise des algorithmes génétiques parce que ceux-ci intègrent de telles méthodes.

En effet, la création de règles par algorithmes génétiques permet une forme simple de généralisation et spécialisation. Dans le cas où les règles sont codées avec un alphabet de trois lettres (0, 1 et #), cela se comprend facilement. # représente un joker codant à la fois pour 0 et 1. Une mutation qui changerait un 0 dans la prémisse d'une règle pour le transformer en # créerait une règle généralisée par définition :

Exemple : $R_0 = 1\ 0\ 1\ \#\ 0\ 1 \rightarrow Action_0 \quad \Rightarrow \quad R_1 = 1\ \#\ 1\ \#\ 0\ \ 1 \rightarrow Action_0$

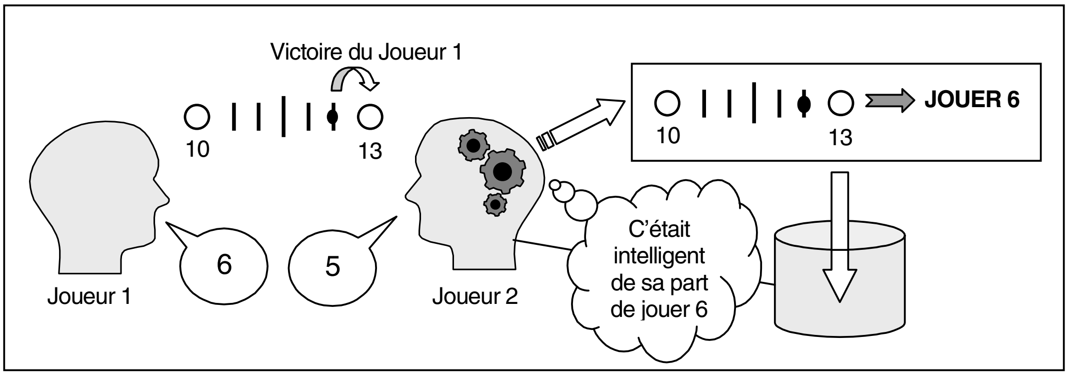
R_1 est créée à partir de R_0 par mutation du deuxième élément (les éléments sont appelés taxons). La prémisse de R_1 est plus générale que celle de R_0 puisque # représente à la fois 0 et 1.

A l'inverse, la mutation d'un # en 0 ou 1 spécialise une règle.

Cette forme de généralisation est simple et suffisante pour de nombreuses applications. Toutefois, la généralisation n'est pas orientée : les règles à généraliser ne sont pas déterminées spécifiquement. Par ailleurs, pour les applications aux jeux de S.A.G.A.C.E., la forme des règles n'est jamais aussi simple. Les règles contiennent des intervalles d'entiers ou de réels et il existe plusieurs façons de généraliser un taxon. Il ne suffit pas de changer un 1 ou un 0 en #, il faut changer un intervalle en un intervalle plus grand. Le choix des nouvelles bornes est, à lui seul, problématique.

En fait, l'utilisation d'un algorithme génétique pour la création des règles dans S.A.G.A.C.E. permet une généralisation et une spécialisation simple et implicite comme celles envisagées par Holland, mais l'introduction de la méthode particulière que nous venons de décrire permet, en plus, une généralisation explicitement dirigée et contrôlée.

7.5.1.2.4 L'imitation

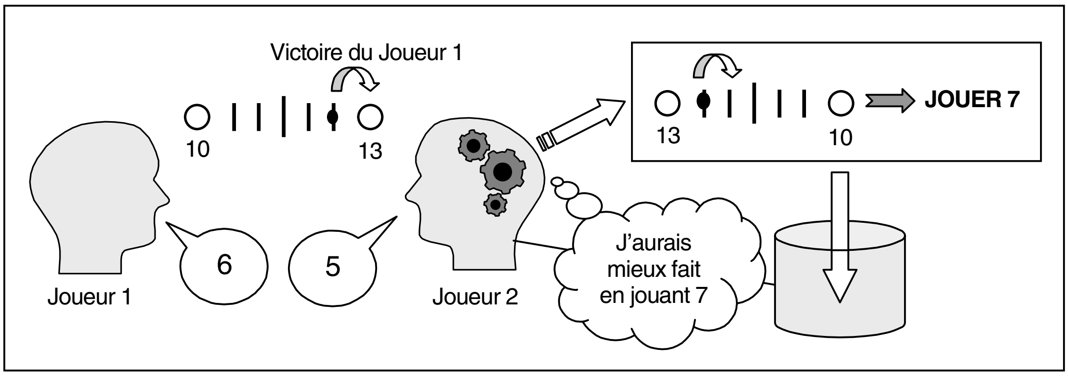


Lors de certaines parties, un coup de l'adversaire peut s'être révélé très judicieux. La méthode d'Imitation imaginée pour S.A.G.A.C.E. permet de tirer profit d'un bon coup de l'adversaire. Il faut cependant remarquer qu'un bon coup de l'adversaire peut être uniquement lié à une erreur personnelle. Les bons coups doivent donc être validés sur une base de cas appropriée avant d'être intégrés dans la base d'un S.C. stratégique (cf. méthode de génération de situations).

Le S.C. d'anticipation contient des descriptions formelles des coups de l'adversaire. Ces descriptions ont un formalisme identique à celui des règles du C.S. stratégique (cf. §7.5.2). Il est ainsi possible, de faire « passer » une description d'un bon coup de l'adversaire dans la base de règles du C.S. stratégique (quelques aménagements sont souvent nécessaires, mais ils sont mineurs).

Cette méthode d'apprentissage par imitation est également utilisée (conjointement à l'Introspection) pour l'amorçage du S.C. stratégique. Il est en effet souvent plus facile pour un expert de jouer contre un élève pour que ce dernier tire ses propres enseignements (avec ses critères d'appréciation propres), plutôt que de tenter d'expliciter les raisons d'un choix avec un vocabulaire ou un formalisme imposé (par la forme des règles du C.S. stratégique). [Meyer, 1996b].

7.5.1.2.5 La méthode des regrets



Les jeux à information complète et imparfaite ont une caractéristique primordiale que nous avons cherché à exploiter :

Quand les deux joueurs ont annoncé leur(s) choix et qu'on a établi les conséquences de ces choix pour les deux joueurs, il est possible, à partir de la connaissance du coup de l'adversaire, de déterminer par le calcul le coup qu'il aurait fallu lui opposer. En fait, quand les joueurs ont annoncé leur coup, on peut raisonner comme pour un jeu à information complète et parfaite pour déterminer la meilleure stratégie de réponse.

La méthode des regrets que nous avons intégrée à S.A.G.A.C.E. est fondée sur cette remarque. Après chaque coup, le système calcule le meilleur coup à opposer à celui de l'adversaire, puis il crée le classeur correspondant qu'il intègre à la base de règle du S.C. stratégique. Les paramètres définissant, par exemple, combien de nouveaux classeurs créer après chaque coup sont contenus dans les bases de métaconnaissance.

Exemple de SUNTZU

Après chaque coup, le S.C. évalue, parmi toutes les actions possibles, celles qui auraient été les plus efficaces en regard de ce qu'a joué l'adversaire. Cette méthode s'est avérée particulièrement redoutable contre les humains ou les joueurs informatiques déterministes dans cette application.

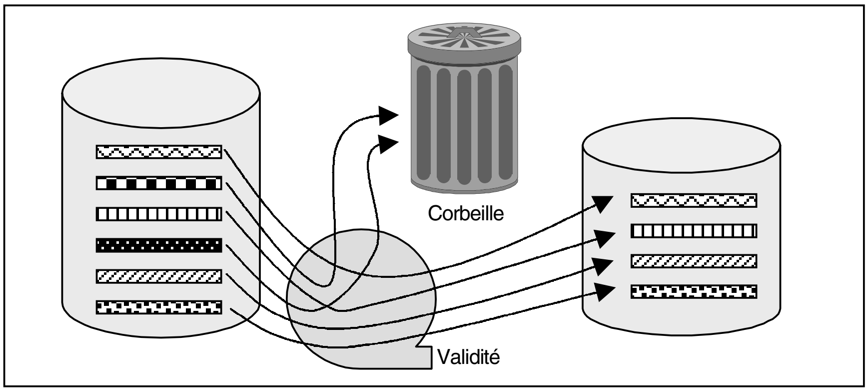
Cela est sans doute lié au fait que les règles du jeu sont relativement complexes pour un humain et que celui-ci est souvent tenté de rejouer les trois mêmes actions si une situation déjà vue se représente.

En règle générale, dans toutes les applications, cette méthode s'avère redoutable contre les joueurs déterministes. La raison est évidente : un adversaire déterministe joue toujours la même chose dans des circonstances identiques. Par conséquent, si le système connaît déjà les meilleurs coups opposés correspondants et s'il les joue, il optimise ses choix. Cela revient, dans ce cas, à utiliser une stratégie optimale pure dans un jeu à information complète et parfaite.

Cette méthode s'apparente à celle définie par « le jeu fictif » d'Owen [Owen, 1995] : les choix passés de l'adversaire dans chaque états du système sont enregistrés pour calculer la distribution probabiliste de ses actions. Owen a démontré que si on choisit une stratégie qui optimise en permanence la façon de répondre à ces observations, la stratégie correspondante converge vers la stratégie optimale. Cette méthode demande cependant qu'il soit possible de déterminer (ou plutôt de calculer) cette meilleure réponse possible. La méthode des regrets de S.A.G.A.C.E. permet de déterminer les éléments nécessaires à l'élaboration de cette stratégie (les stratégies pures entrant dans la stratégie mixte optimale). L'exploitation des règles découvertes par la méthode des regrets est laissée à l'apprentissage, lequel permet, par ajustement de la valeur sélective des règles ainsi créées, de déterminer la stratégie mixte optimale.

Comme la méthode des regrets permet de déterminer toutes les stratégies pures nécessaires et comme l'apprentissage permet d'ajuster les valeurs sélectives des règles correspondantes, cela assure la convergence à l'infini de S.A.G.A.C.E. vers la solution optimale contre tout adversaire. Bien évidemment, cette convergence n'est que théorique, comme l'est celle de Owen. Cependant S.A.G.A.C.E. ne se résume pas à cette convergence à l'infini qui est inexploitable en pratique.

7.5.1.3 Méthodes d'élimination de règles

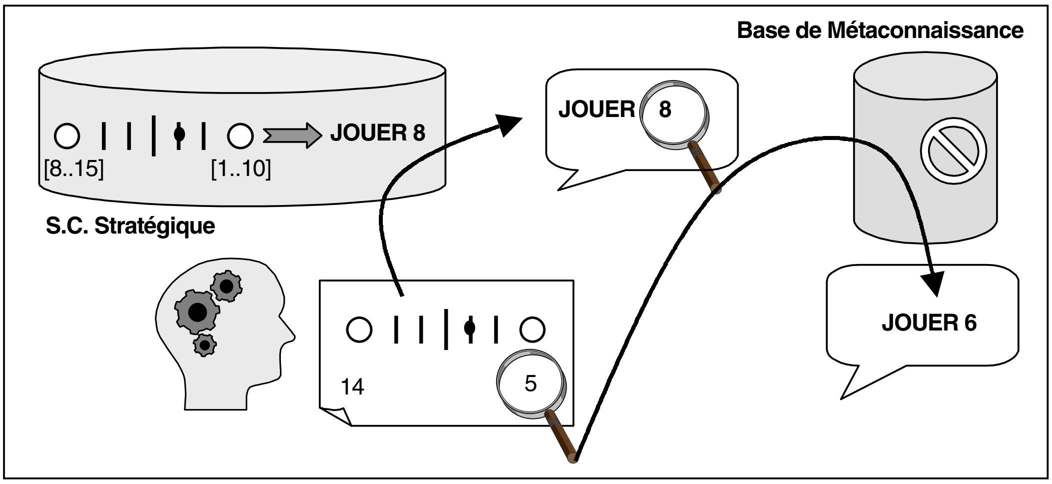


S.A.G.A.C.E. intègre une procédure d'élimination des règles invalides qui s'appuie sur une description au seins des bases de métaconnaissances, de la forme que doivent prendre les règles valides.

Les méthodes de création de règles étant nombreuses, cette méthode permet de vérifier que toute règle créée est valide.

Normalement, de telles procédures de création ne devraient engendrer que des règles valides. Cependant certaines procédures (comme la généralisation) sont parfois très complexes et il est utile d'avoir un moyen sûr de « nettoyer », a posteriori les bases de règles des C.S.

7.5.1.4 Les bases de métaconnaissances



Nous avons souvent fait mention de ces bases et avons déjà décrit une grande partie de leur utilisation par S.A.G.A.C.E. (paramétrage des méthodes, vérifications, éliminations, etc.).

Les bases peuvent être également utilisées pour contraindre les choix des joueurs. Au-delà de l'élimination des règles invalides, elles permettent de modifier le choix effectué par le S.C. stratégique si celui-ci s'avère sûrement inadapté.

Exemple d'ALESIA

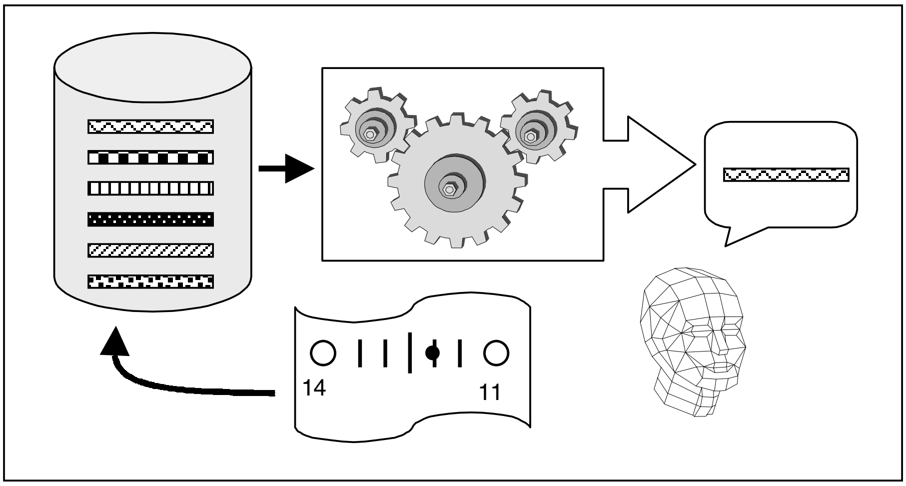
Dans l'implémentation de S.A.G.A.C.E. pour ALESIA, nous avons ajouté certaines méta-règles comme par exemple :

- *Si une règle est déclenchée et qu'elle suggère de jouer plus d'un point de plus que le capital de points de l'adversaire, alors modifier ce choix en jouant le nombre de points de l'adversaire plus un.*
- *Si une règle est déclenchée et qu'elle suggère de jouer tous les points restants alors que le jeton n'est pas dans une position extrême, ne pas la sélectionner.*

Les règles ainsi modifiées ou court-circuitées ne sont pas nécessairement invalides, mais elles suggèrent un choix incohérent dans un contexte précis et c'est à ce titre que les méta-règles permettent de palier certaines lacunes de la base de règles originales.

La découverte de méta-règles efficaces de ce type n'est pas évidente. Nous avons pensé utiliser (encore…) un algorithme génétique pour ce faire mais les résultats ont été d'autant moins été convaincants que définir la forme de ces règles est déjà un épineux problème en soi.

7.5.1.5 Méthodes de sélection des classeurs



S.A.G.A.C.E. intègre différentes méthodes de sélection du ou des classeurs définissant à chaque cycle, l'action retenue. Ces méthodes sont dites « d'arbitrage ».

Pour chaque configuration de jeu, plusieurs classeurs peuvent être déclenchés simultanément parce que leur partie *condition* est compatible avec la situation. Si ces classeurs ont des parties *conclusion* différentes alors il faut un processus d'arbitrage.

Dans certaines applications ou certains jeux, certaines conclusions différentes peuvent ne pas être incompatibles et sont alors appliquées simultanément. Par exemple, certains classeurs peuvent conclure sur les coups à ne pas jouer tandis que d'autres concluent sur les coups à jouer ; si deux classeurs (un parmi chacune de ces catégories) concluent sur deux coups différents, ils peuvent s'appliquer simultanément.

Fonctionnellement, c'est au niveau de l'interface de sortie qu'est déterminé le choix du système. En fonction de la situation du jeu (l'environnement), les classeurs appariés (dont la partie *condition* est compatible avec la situation) sont déclenchés et déposent leur(s) message(s) sur la liste des messages. Quand tous les messages à usage interne (qui permettent l'appariement de classeurs) ont été traités, l'arbitrage entre les messages de sortie (choix du système) s'effectue.

S.A.G.A.C.E. intègre les trois méthodes les plus classiques d'arbitrage :

- La méthode élitiste pure ;
- La méthode de la roue simple ;
- La méthode de la roue exponentielle.

7.5.1.5.1 La méthode élitiste pure

Elle consiste simplement à choisir le message de sortie du classeur déclenchable ayant la plus grande valeur sélective.

S.A.G.A.C.E. intègre cette méthode parce qu'elle est la plus simple et qu'elle est parfois très efficace. Son avantage et son inconvénient majeurs résident dans le fait qu'elle n'intègre aucun processus stochastique.

Dans des circonstances exactement semblables (situation de jeu identique, base de règles rigoureusement identique), les décisions du système seront toujours les mêmes : un comportement optimal est donc persistant, ce qui est un point positif. Par ailleurs, l'apprentissage peut modifier la valeur sélective des classeurs du système et donc modifier le comportement du système si celui-ci s'avère inadapté.

Dans un contexte de jeu répété, cette méthode d'arbitrage est néanmoins dangereuse parce qu'elle est très prévisible dans certaines situations.

Exemple du jeu des trois pierres

Supposons l'existence des deux classeurs suivants dans la base de règles :

Condition		***Conclusion***		***Fitness***
Nb de pierres	*Nb de pierres adversaire*	*MISER*	*SOMME*	
2	*2*	*1*	*3*	*21*
2	*2*	*2*	*2*	*29*

Dans la situation où les deux joueurs ont chacun deux pierres, le joueur considéré va miser 2 pierres et parier que la somme totale sera de 2 pierres conformément au deuxième classeur.

Imaginons qu'il gagne la partie. Dans ce cas, la fitness du deuxième classeur ne peut qu'augmenter par apprentissage. Cela implique que, ultérieurement dans la même situation, ce joueur rejouera exactement la même chose ce qui en fait un joueur très prévisible (un joueur intelligent aurait, au moins, considéré la possibilité d'utiliser le premier classeur).

Ainsi, la méthode élitiste pure présente des inconvénients importants notament dans le cas des jeux à information complète et imparfaite pour lesquels un joueur doit demeurer le moins prédictible possible.

7.5.1.5.2 La méthode de la roue simple

Il s'agit d'une méthode de choix stochastique. Elle consiste à effectuer un tirage aléatoire selon une répartition probabiliste dont chaque coefficient correspond à la valeur sélective d'un classeur déclenché.

Cette méthode est particulièrement adaptée aux jeux à information complète et imparfaite dans la mesure où elle permet d'adopter une stratégie mixte.

Exemple du jeu des trois pierres

On considère à nouveau les deux classeurs précédents :

Condition		**Conclusion**		**Fitness**
Nb de pierres	*Nb de pierres adversaire*	*MISER*	*SOMME*	
2	*2*	*1*	*3*	*21*
2	*2*	*2*	*2*	*29*

Dans la situation où les deux joueurs ont chacun deux pierres, le joueur considéré, s'il utilise la méthode de la roue, va effectuer un tirage aléatoire pour déterminer s'il va miser 1 point et parier que la somme sera 3 ou s'il va miser 2 points et parier que la somme sera 2. La probabilité de chacun de ces choix est proportionnelle aux fitness respectives des deux classeurs (42% et 58%).

7.5.1.5.3 La méthode de la roue exponentielle

Cette méthode, plus rarement utilisée, est une adaptation de la méthode précédente. Elle consiste à déterminer chaque coefficient de la répartition probabiliste à partir d'une fonction exponentielle de la fitness des classeurs correspondants.

Cette méthode permet de paramétrer l'aspect élitiste du tirage aléatoire. Dans l 'exemple précédent, si la fonction utilisée est : exp(3.4*ln(x)), le tirage aléatoire s'effectue proportionnellement à exp(3.4*ln(21)) et exp(3.4*ln(29)) soit (25% et 75%).

En revanche, si la fonction utilisée est : exp(0.6*ln(x)) le tirage s'effectue suivant (45% et 55%).

7.5.1.5.4 La méthode originale d'élection

Nous avons défini une méthode de choix originale pour les jeux dans lesquels les actions des joueurs concernent plus d'une donnée (par exemple, pour le jeu des trois pierres, chaque joueur doit choisir le nombre de pierres qu'il mise et la somme des mises des deux joueurs).

Cette méthode s'apparente quelque peu à la méthode de sélection des classeurs pour la reproduction génétique : les classeurs déclenchés déposent leurs différents messages (un par action) sur la liste des messages, auxquels est attribuée leur fitness (celle des classeurs). Les messages d'action sont alors regroupés par actions identiques et l'action retenue est déterminée par tirage aléatoire pondéré suivant la somme des fitness associées aux messages.

Il faut remarquer que, si certains classeurs concluent sur plusieurs actions de nature différente, c'est parce qu'elles ne sont pas indépendantes. Par exemple, pour le jeu des trois pierres, les classeurs concluent sur la mise et sur la somme parce que la somme n'a, a priori, aucune raison d'être inférieure à la mise[1]. Pour éviter que cette méthode d'élection ne sélectionne deux actions incompatibles, le processus est supervisé par des métaconnaissances chargées de gérer ces éventuelles incompatibilités.

7.5.1.5.5 La méthode originale de répartition dynamique

Nous avons défini une méthode originale de choix stochastique dynamique. Le but de cette méthode est de permettre d'effectuer le choix parmi un sous-ensemble des classeurs déclenchés.

La nécessité du développement d'une telle méthode s'est imposée lorsque certaines applications (Suntzu particulièrement) requerraient la manipulations de grande bases de classeurs (plus d'un millier). Le principe de cette méthode est d'effectuer la sélection sur une sous-partie des classeurs déclenchables dans un temps de calcul fixé par avance ou dynamique. Il s'agit de pouvoir arrêter le processus de sélection à tout moment, en étant capable de sélectionner un classeur avec une probabilité limite équivalente à ce que cette probabilité aurait dû être si tous les classeurs déclenchables avaient été considérés. Le souci était également de ne pas avoir à faire la somme des fitness de tous les classeurs, comme c'est le cas pour la méthode de la roue.

---

[1] En fait, lorsqu'un joueur joue en premier, il est parfois intéressant de proposer une somme inférieure à la mise : il s'agit d'un bluff induisant l'adversaire en erreur dans ses estimations de la mise (ce qui l'amène à donner également une somme inadaptée et lui fait reprendre la main pour le coup suivant).

Dans cette perspective, le principe de la méthode est le suivant :

Les classeurs sont traités deux par deux dans un ordre quelconque (indifférent, par valeur de fitness, par spécificité, etc.). Chaque étape consiste à ne retenir que l'un des deux classeurs. Pour cela, un choix est effectué aléatoirement avec des probabilités proportionnelles à la fitness des deux classeurs. Le classeur élu participe alors à une nouvelle sélection avec le classeur suivant mais sa fitness est augmentée de celle du classeur antagoniste précédent.

Exemple :

Considérons un ensemble de classeurs déclenchables $y_1,\ldots,y_p$ *et leurs fitness associées :*

$y_1 \rightarrow f_1$ $y_2 \rightarrow f_2$ ... ... $y_p \rightarrow f_p$

On effectue un premier choix entre y1 et y2 puis entre le vainqueur et y3 etc.

y_1 *Vs* y_2 *probabilité de choisir* $y_1 = f_1/(f_1+f_2)$ *supposons que* y_2 *soit sélectionné.*

y_2 *Vs* y_3 *probabilité de choisir* $y_2 = (f_1+f_2)/(f_1+f_2+f_3)$*. Supposons que* y_2 *soit de nouveau sélectionné.*

y_2 *Vs* y_4 *probabilité de choisir* $y_2 = (f_1+f_2+f_3)/(f_1+f_2+f_3+f_4)$

etc...

Si le processus est itéré jusqu'au traitement de l'ensemble des classeurs, la probabilité de choisir y2 est : $(1 - f_1/f_1+f_2) * (f_1+f_2)/(f_1+f_2+f_3) * (f_1+f_2+f_3)/(f_1+f_2+f_3+f_4) * \ldots * (f_1+\ldots+f_{p-1})/(S_f) = f_2/S_f$*.* *(*$S_f$ *étant la somme des fitness de tous les classeurs déclenchés).*

La probabilité limite est bien égale à la probabilité qui aurait été obtenue si tous les classeurs avaient été considérés.

Avantages de la méthode

- Ce traitement peut être effectué pendant la phase d'appariement (détermination des classeurs déclenchables). Il peut donc être interrompu avant que tous les classeurs n'aient été traités.
- Il n'est pas indispensable d'effectuer un tri sur les classeurs (en effet, une méthode quasi-équivalente consisterait à effectuer un tri sur les classeurs puis à n'effectuer la sélection que sur une partie d'entre eux, ce qui nécessiterait d'effectuer la phase d'appariement en entier).

Inconvénients de la méthode

- De nombreux traitements sont nécessaires, suivant le nombre de classeurs finalement pris en compte (divisions et tirages aléatoires). Naturellement, la somme partielle des f1…fi (i<p) est incrémentée itérativement et n'est pas recalculée à chaque étape.

Remarque 1 :

Si les classeurs sont triés par fitness décroissantes, on peut arrêter le processus quand on atteint un certain 'pourcentage' de fitness totale. On peut aussi décider d'arrêter quand un candidat a gagné un certain nombre d'affrontements !...

C'est théoriquement satisfaisant dans la mesure où à la limite, cette alternative converge vers la solution classique de choix aléatoire.

Remarque 2 :

La méthode est particulièrement adaptée dans le domaine des jeux pour approximer la solution mixte optimale. En effet, on peut appliquer les règles selon une répartition probabiliste de celles-ci qui dépend de leurs fitness tout en ne considérant qu'un certain nombre de règles comme cela est expliqué en remarque 1.

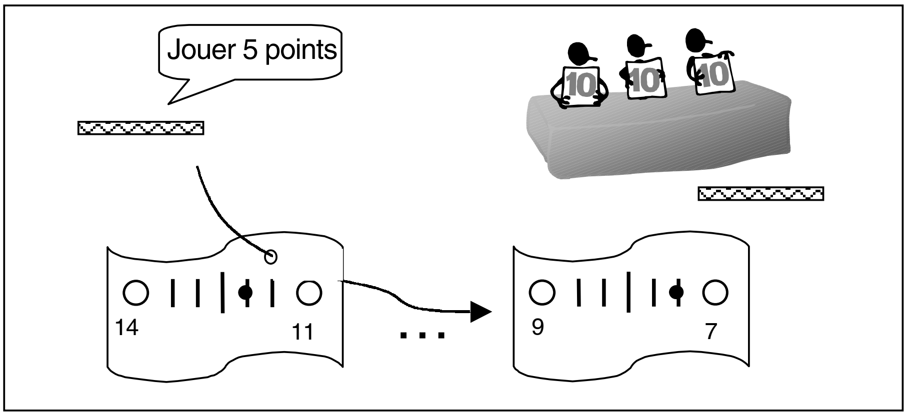
Remarque 3 :

S'il s'avère important que les classeurs soient triés, le tri peut être effectué itérativement. A chaque modification de la fitness d'un classeur, la position de ce dernier peut être actualisée dans une liste générale.

Remarque 4 :

L'idée d'effectuer la sélection aléatoire sur une sous-partie des classeurs est inspirée du théorème de Lipton et Young [Lipton, 1986] concernant les stratégies simples pour les larges jeux à somme nulle. Ce théorème certifie qu'il est possible d'approximer la stratégie optimale au sens de Nash en effectuant une sélection aléatoire *uniforme* restreinte à un nombre déterminé de sous-ensembles de stratégies pures. Dans ce cas, l'espérance de gain correspondante est une approximation de la valeur du jeu complet dont l'erreur désirée définit le nombre de stratégies pures à considérer. La taille de l'ensemble des stratégies pures varie alors selon le logarithme de la taille de l'ensemble des stratégies pures de l'adversaire du jeu original.

7.5.1.6 Mise à jour de la fitness des classeurs



7.5.1.6.1 Les mises à jour des fitness

Différentes méthodes de mise à jour de la fitness des classeurs ont été intégrées à S.A.G.A.C.E.

La plupart sont des méthodes classiques en apprentissage :

- Attribution positive ou négative de crédit en fonction des performances des classeurs ;
- Algorithme du Bucket Brigade ;
- Algorithme du Profit Sharing Plan ;
- Apprentissage par renforcement.

Ces méthodes ont déjà été décrites précédemment et, à quelques ajustements minimes près, n'ont pas été transformées.

La méthode qui s'est révélée être (à la lumière de nos différentes implémentations) la mieux adaptée à l'apprentissage pour les jeux à information complète et imparfaite est une méthode parfois employée en apprentissage par renforcement appelée : « renforcement exponentiel par pondération temporelle » ou « exponential recency-weighted average » [Sutton, 1998].

Il s'agit d'une méthode de renforcement accordant plus d'importance à un renforcement récent qu'à un renforcement perçu précédemment. Classiquement, lorsqu'un classeur reçoit un renforcement (positif ou négatif) en fonction du résultat de son application, sa fitness est mise à jour de la façon suivante :

$$Fitness_t(C) = \frac{r_1 + r_2 + ... + r_{k_C}}{k_C}$$

Où les r_i sont les différents renforcements reçus et K_C le nombre de déclenchements.

Cette méthode est usuellement programmée de façon incrémentale et on calcule la fitness d'un classeur en fonction de sa fitness précédente :

$$Fitness_{k+1}(C) = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = Fitness_k(C) + \frac{1}{k+1}\left[r_{k+1} - Fitness_k\right]$$

Dans le renforcement exponentiel par pondération temporelle, le calcul de la fitness d'un classeur après son $(k+1)^{ème}$ renforcement se calcule de la façon suivante :

$$Fitness_{k+1} = Fitness_k + \alpha\left[r_{k+1} - Fitness_k\right]$$

Où α est appelé 'pas' du renforcement (0 < α □ 1). En règle générale, α est constant.

On vérifie aisément qu'avec cette méthode de mise-à-jour, les renforcements les plus récents sont pondérés de manière plus importante que les renforcements les plus anciens. En fait, le calcul de la fitness est une moyenne pondérée des renforcements et de la fitness initiale ($Fitness_0$) :

$$Fitness_{k+1} = (1-\alpha)^{k+1} Fitness_0 + \sum_{i=1}^{k+1} \alpha(1-\alpha)^{k-i+1} r_i$$

Exemple du jeu des trois pierres

Pour l'utilisation de S.A.G.A.C.E. à ce jeu, nous avons attribué (de manière arbitraire) la $fitness_0$ (initiale) des classeurs à 0,25. Nous avons attribué les renforcements suivants :

- *Gain d'une pierre : 1 (la somme est correcte) ;*
- *Perte d'une pierre : 0 (l'adversaire a gagné le coup) ;*
- *Prise de la main : 0,1 (coup nul mais passage à l'état de premier joueur) ;*
- *Passage de la main : 0,4 (coup nul mais passage à l'état de deuxième joueur) ;*
- *α : 0,01 : pas de renforcement.*

Ainsi, un classeur C sélectionné pour la première fois verra sa fitness évoluer de $Fitness_0 = 0,25$ à (suivant le cas) :

- *0,25 + 0,01*(1 - 0,25) = 0,2575* *(Gain d'une pierre) ;*
- *0,25 + 0,01*(0 - 0,25) = 0,2475* *(Perte d'une pierre) ;*
- *0,25 + 0,01*(0,1 - 0,25) = 0,2485* *(Prise de la main) ;*
- *0,25 + 0,01*(0,4 - 0,25) =0,2515* *(Passage de la main).*

La raison pour laquelle cette méthode s'est avérée la plus efficace contre des joueurs humains est, à notre avis, liée à la mémoire à court terme des humains et à l'importance qu'ils accordent aux coups les plus récents. En ce sens, cette méthode est également la plus logique par rapport à la vocation première de S.A.G.A.C.E. qui est de jouer contre des joueurs humains.

7.5.1.6.2 Les particularités de S.A.G.A.C.E.

Prédictions erronées

S.A.G.A.C.E. fait la distinction entre les classeurs tenant compte des messages du S.C. d'anticipation (partie prédiction renseignée) des classeurs qui n'en tiennent pas compte (partie prédiction laissée libre). Plus particulièrement, pour les classeurs tenant compte de ces messages, une distinction est faite selon que la prédiction était bonne ou erronée. La raison en est simple : un classeur, ayant émis un message d'action en fonction d'une action présumée de l'adversaire s'étant révélée inexacte n'a pas à être renforcé (positivement ou négativement). Ainsi, les classeurs qui utilisent la modélisation ne sont renforcés que si les prédictions sur lesquelles leur déclenchement s'est basé se sont révélées exactes.

Méthode de renforcement par regret

Pour accélérer le processus d'apprentissage, S.A.G.A.C.E. utilise une autre forme de la méthode originale des regrets. Comme nous l'avons signalé, après que les deux joueurs aient annoncé leurs actions, il est possible de déterminer quel aurait été le meilleur coup en fonction de celui de l'adversaire. Ainsi, il est possible de renforcer les éventuels classeurs déclenchables qui auraient (s'ils avaient été sélectionnés) émis les messages optimaux. Nous avons appelé cette méthode « renforcement par regrets ».

7.5.2 Les bases de règles du S.C. d'anticipation

7.5.2.1 Forme des classeurs

Il faut pouvoir décrire sous la forme de classeurs tout comportement observé d'un joueur en fonction de la situation du jeu.

Ainsi, la forme des classeurs du S.C. d'anticipation est la même que celle des classeurs du S.C. stratégique diminuée de la partie prédiction.

Exemple d'Alésia

Ainsi, la forme des classeurs d'anticipation pour ALESIA est la suivante :

Condition			***Action***
Nombre points	*Nombre points de l'adversaire*	*Position jeton*	*Mise*
[1..50]	*[1..50]*	*[-3..3]*	*[1..50]*

Elle correspond à ce que nous avions défini comme la forme minimale des classeurs.

L'enveloppe de ces classeurs contient les mêmes information que celle des classeurs stratégiques, plus deux coefficients particuliers de modélisation UT_0 et UT_1 dont le rôle et l'utilisation sont expliqués dans les sections suivantes (§7.5.2.).

7.5.2.2 Méthodes de création des classeurs et enveloppe

7.5.2.2.1 Création des classeurs

Les classeurs du C.S. d'anticipation sont créés dynamiquement suivant l'observation directe du comportement de l'adversaire. La partie *condition* des classeurs est renseignée par la situation du jeu et la partie *action* par les choix effectués par l'adversaire.

Exemple du jeu des trois pierres :

Si dans la situation où les deux joueurs avaient chacun deux pierres, l'adversaire a misé une pierre et a déterminé que la somme devait faire trois, alors le classeur suivant sera créé dans la base du S.C. d'anticipation :

Condition		***Conclusion***	
Nb de pierres	*Nb de pierres adversaire*	*MISER*	*SOMME*
2	*2*	*1*	*3*

7.5.2.2.2 Enveloppe des classeurs

Une des principales originalités de S.A.G.A.C.E. est la façon dont sont construites et mises à jour les enveloppes de ces classeurs.

Quand un classeur est créé, suite à l'observation du comportement de l'adversaire, un certain nombre de coefficients lui sont adjoints. Ces coefficients renseignent sur l'utilisation de ce classeur virtuel par l'adversaire[1]. Il s'agit de :

- La fitness estimée ;
- UT_0 ;
- UT_1 .

La fitness estimée

La fitness estimée est, comme son nom l'indique, une estimation par l'observateur de la valeur sélective que devrait avoir le classeur correspondant s'il était réellement contenu dans une base de règles de l'adversaire. Elle sera désormais notée : e_fitness. Typiquement, si un coup de l'adversaire est original (il ne l'avait jamais joué dans la situation actuelle) et qu'il s'est avéré bon pour lui, l'e_fitness doit être forte (en supposant que l'adversaire aura tendance à réitérer ce choix ultérieurement dans cette situation). Réciproquement si le coup s'est avéré mauvais, l'e_fitness sera faible.

Si l'adversaire renouvelle un coup dans une situation identique alors, le classeur précédemment créé est identifié et son e_fitness est mise à jour en fonction du résultat de cette nouvelle application virtuelle du classeur (résultat obtenu par l'adversaire en renouvelant cette action dans cette situation).

Les e_fitness sont utilisées pour prédire le comportement de l'adversaire dans une situation déjà rencontrée.

Une version simplifiée de S.A.G.A.C.E. consiste à rechercher, dans une situation donnée, parmi les classeurs déclenchables du S.C. d'anticipation (les classeurs qui représentent le comportement présumé de l'adversaire) celui qui a la plus forte e_fitness et à jouer en fonction de l'action correspondante.

Exemple d'Alésia

Supposons la présence dans le C.S. d'anticipation des classeurs suivants avec leur e_fitness :

---

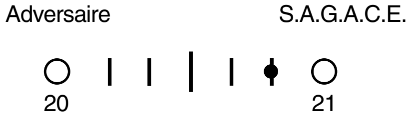
[1] Nous disons virtuel parce que ce type de classeur est déterminé par S.A.G.A.C.E., pas par l'adversaire lui-même. Ces classeurs ne sont, sous cette forme, que des observations comportementales. Il est possible que l'adversaire tienne compte de facteurs particulier non identifiés dans la partie condition de ces classeurs (son humeur, le temps qu'il fait,…).

Condition			*Action*	*E_fitness*
Nb points	*Nb points adv.*	*Position jeton*	*Mise*	
20	*21*	*2*	*20*	*10,05*
20	*21*	*2*	*7*	*5,00*
20	*21*	*2*	*2*	*9,18*

Supposons la présence dans le C.S. stratégique des classeurs suivants :

Condition			*Prédiction*	*Action*	*fitness*
Nb points	*Nb points adv.*	*Position jeton*	*Mise de l'adv.*	*Mise*	
21	*20*	*-2*	*20*	*20*	*14,12*
21	*20*	*-2*	*20*	*21*	*16,40*
21	*20*	*-2*	*7*	*8*	*11,08*
21	*20*	*-2*	*2*	*3*	*17 ,12*

Dans la situation suivante :



S.A.G.A.C.E. va anticiper le fait que son adversaire va miser 20 points car la e_fitness du classeur indiquant de jouer 20 est la plus grande (10,05). Supposant que l'adversaire va jouer 20, S.A.G.A.C.E. va jouer 21 parce que le classeur, proposant cette réponse à une mise de 20 présumée, est celui qui a la plus grande fitness (16,40). Si la prédiction est exacte, le coup est parfait dans la mesure où il garantit le nul à S.A.G.A.C.E.

Imaginons que l'adversaire joue réellement 20 points. Dans ce cas, l'action correspondante sera jugée par S.A.G.A.C.E. plutôt mauvaise (pour l'adversaire) et la e_fitness du classeur va être diminuée. Si elle devient inférieure à 9,18 alors S.A.G.A.C.E. prédira, la prochaine fois que la situation de jeu se reproduira, que l'adversaire jouera 2 points (et il jouera alors 3 points par anticipation).

L'exemple précédent est tiré d'une partie réelle entre S.A.G.A.C.E. et un humain. Cette méthode s'est révélée très satisfaisante. Cependant, la notion d'e_fitness est très subjective. L'e_fitness est en effet calculée par rapport à des critères internes à S.A.G.A.C.E. et rien ne permet d'affirmer que son adversaire utilise les mêmes critères (si tant est qu'il joue avec ce type de formalisme, à base de règles).

Il est ainsi fréquent que les estimations d'e_fitness soient erronées. En fait, l'e_fitness est une valeur trop précise pour estimer une notion si subjective. L'e_fitness doit être considérée comme un mesure d'utilité (cf. §6.7). Dans un jeu comme Alésia, il est très difficile de deviner l'importance qu'un joueur humain accorde aux conséquences des choix de chaque joueur (passage d'une position du jeton à une autre en fonction des points engagés et des points restants). Notons, toutefois, que cette méthode a toujours donné d'excellents résultats contre des programmes utilisant des techniques d'apprentissage classiques.

Nous avons imaginé une méthode de correction d'erreur visant à ajuster la e_fitness en fonction du comportement réellement observé de l'adversaire. Cependant, cette méthode se greffant à la première méthode de mise à jour, il devenait quasiment impossible de déterminer laquelle était efficace ou inefficace. Les résultats ont d'ailleurs été relativement décevants et l'idée a été abandonnée.

La solution finalement retenue pour pallier l'inadéquation entre la précision de la e_fitness et la subjectivité de la méthode repose sur une approche plus simple :

A chaque classeur du S.C. d'anticipation sont adjoints deux paramètres notés UT_0 et UT_1 renseignant réciproquement sur le nombre d'utilisations des classeurs et sur leur nombre d'utilisations avec succès. Il est en effet plus simple de déterminer si une action a été utilisée et si elle l'a été avec un relatif succès que de déterminer les enseignements exacts que l'adversaire en a tirés. En clair, la notion d'e_fitness est abandonnée au profit de ces deux paramètres.

Un succès relatif est, par exemple, le gain du coup (sinon de la partie) ou encore un coup nul (c'est un choix d'implémentation).

Exemple d'Alésia

Pour l'implémentation de S.A.G.A.C.E. pour ALESIA, nous avons considéré qu'un succès relatif correspond au gain de la partie ou à l'avancée du jeton dans la bonne direction pour un sacrifice de points raisonnable (l'estimation du « raisonnable » prend en compte la position d'arrivée du jeton et la différence entre les capitaux des deux joueurs après le coup).

Exemple du jeu des trois pierres

Pour cette implémentation de S.A.G.A.C.E. nous avons considéré qu'un succès relatif correspond au gain du coup (écartement d'une pierre) ou au fait de donner la main à l'adversaire (le second joueur est avantagé à ce jeu).

A l'évidence, certains critères sont encore légèrement subjectifs mais ils font l'unanimité chez les joueurs. Il est en effet rare que les opinions divergent sur le succès d'un coup, particulièrement dans les jeux à information complète et imparfaite ou une anticipation correcte du coup de l'adversaire est, en soi, un succès relatif.

7.5.2.3 Les paramètres UT_0 et UT_1

UT_0 et UT_1 sont des registres de mémoire à court terme. Se basant sur l'idée que les adversaires humains (qui sont la cible privilégiées de S.A.G.A.C.E.) intègrent des processus de mémoire à court terme, ces paramètres permettent d'estimer ce qu'un joueur humain a retenu des derniers coups joués.

Concrètement, UT_0 et UT_1 se présentent sous la forme de registres à décalage. Ils sont codés sur 16 bits (la mémoire à court terme d'un humain est encore plus petite, cf. Chap 6).

$UT_0 = X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7\ X_8\ X_9\ X_{10}\ X_{11}\ X_{12}\ X_{13}\ X_{14}\ X_{15}\ X_{16}$ (représentation binaire) ;

$UT_1 = Y_1\ Y_2\ Y_3\ Y_4\ Y_5\ Y_6\ Y_7\ Y_8\ Y_9\ Y_{10}\ Y_{11}\ Y_{12}\ Y_{13}\ Y_{14}\ Y_{15}\ Y_{16}$ (représentation binaire).

7.5.2.3.1 Mise à jour de UT_0 et UT_1

Chaque fois qu'un classeur d'anticipation est déclenché, UT_0 et UT_1 subissent un décalage binaire à droite. Quand un classeur a été sélectionné (nécessairement parmi les classeurs déclenchables) le bit de poids fort de UT_0 est positionné à 1. Enfin, chaque fois qu'un classeur utilisé l'a été avec un relatif succès, le bit de poids fort de UT_1 est positionné à 1.

Les opérations de mise à jour de UT_0 et UT_1 sont donc de simples opérations binaires.

exemple

Imaginons un classeur C du S.C. d'anticipation et $UT_0(C)$ et $UT_1(C)$ ses paramètres associés. Supposons que $UT_0(C)$ et $UT_1(C)$ soient les suivants avant le déclenchement de C :

UT_0 = `0010110011001110` soit l'entier 11.470 ;

UT_1 = `0010010010000010` soit l'entier 9.346.

• Après déclenchement de C :

UT_0 = `0001011001100111` soit l'entier 5.735 ;

UT_1 = `0001001001000001` soit l'entier 4.673.

Les bits de poids faible ont été éliminés, on peut faire l'analogie avec l'oubli d'une donnée ancienne.

L'opération de déclenchement a uniquement consisté à faire une division par deux des nombre entiers représentants $UT_0(C)$ et $UT_1(C)$.

Imaginons que C ait été sélectionné et que le résultat de son application ait été un échec relatif (perte du coup).

• Après sélection et application de C :

UT_0 = `1001011001100111` soit l'entier 38.503 ;

UT_1 = `0001001001000001` soit l'entier 4.673 (non modifié).

La seule opération nécessaire a été une somme (de l'entier 32768). Les opérations ne peuvent mener à des retenues binaires dans la mesure où les nombres sont toujours préalablement divisés par deux.

Remarque

Les opérations informatiques telles que le décalage à droite (division par deux) ou la modification du bit de poids le plus fort (addition de 32768) sont extrêmement rapides. La mise à jour des paramètres UT_0 et UT_1 est bien plus rapide qu'une modification éventuelle d'e_fitness pour tous les classeurs déclenchables (opérations sur des réels).

7.5.2.4 La fonction S.U.C.R.E.

Afin de pouvoir arbitrer, pour l'anticipation, entre les différents classeurs déclenchables nous avons défini une fonction de sélection. Cette fonction utilise les paramètres UT_0 et UT_1. C'est par comparaison de l'application de cette fonction aux classeurs que sont déterminés les classeurs qui participent à l'anticipation.

S.U.C.R.E. (pour Succès et Utilisation Calculés de Règles Evolutives) est la somme de deux fonctions $Ipt(UT_\alpha)$ où α vaut 0 ou 1 (pour UT_0 et UT_1). Le terme *Ipt* signifie « Importance de la proximité temporelle ». Dans sa forme générique, la fonction *Ipt* se définit ainsi :

$$Ipt(UT_\alpha) = \sum_{i=0}^{\Delta-1} (UT_\alpha \wedge 2^i)^{\Psi_\alpha(i,\Delta)}$$

Avec les notations suivantes :

- Le symbole Λ représente le AND binaire ;
- □ est la longueur de la chaîne binaire associée aux UT_α (fixée à 16 dans les implémentations de S.A.G.A.C.E. que nous avons faites) ;
- Ψ est une fonction de i (position binaire) et de □.

Une version simplifiée de Ipt consiste à fixer $\Psi_\alpha(x,y) = c_\alpha$ (constantes) $\forall$ x, $\forall$ y. *Ipt* s'exprime alors sous la forme :

$$Ipt(UT_\alpha) = \sum_{i=0}^{15} (UT_\alpha \wedge 2^i)^{c_\alpha}$$

Si c_α vaut 1, *Ipt* est la fonction identité : $Ipt(UT_\alpha) = UT_\alpha$.

Remarque :

Dans les différentes implémentations que nous avons faites de S.A.G.A.C.E., nous avons fixé $\Psi(x,y) = c_\alpha$ (constantes). Par ailleurs, nous avons fixé la formule suivante : S.U.C.R.E.(C) = $Ipt(UT_0)$+ $Ipt(UT_1)$. Nous n'avons pas tenté de pondérer la somme pour donner plus d'importance à un terme par rapport à l'autre parce que cela se fait naturellement en agissant sur les fonctions (ou constantes) Ψ_α.

Exemple

Soit un classeur C avec les paramètres suivants :

UT_0 = `0001000000100001` *soit l'entier 5.735 ;*

UT_1 = `0001000000000001` *soit l'entier 4.673.*

On calcule suivant la valeur de Ψ :

- 1^{er} *cas :* $\Psi_\alpha(i, □) = 1$

$Ipt(UT_0) = UT_0 = 5.735$;

$Ipt(UT_1) = UT_1 = 4.673$.

S.U.C.R.E. (C) = 10.408.

- $2^{ème}$ *cas :* $\Psi_\alpha(i, □) = \frac{1}{2}$

$Ipt(UT_0) = 4096^{\frac{1}{2}} + 32^{\frac{1}{2}} + 1^{\frac{1}{2}} = 64 + 5,6568 + 1 = 70,65$

$Ipt(UT_1) = 4096^{\frac{1}{2}} + 1^{\frac{1}{2}} = 64 + 1 = 65.$

S.U.C.R.E. (C) = 135,65

- $3^{ème}$ *cas :* $\Psi_0(i, □) = \frac{1}{2}$ & $\Psi_1(i, □) = \frac{3}{4}$

$Ipt(UT_0) = 4096^{\frac{1}{2}} + 32^{\frac{1}{2}} + 1^{\frac{1}{2}} = 64 + 5,6568 + 1 = 70,65$

$Ipt(UT_1) = 4096^{\frac{3}{4}} + 1^{\frac{3}{4}} = 512.$

S.U.C.R.E. (C) = 582,65

Les différences entre les $2^{ème}$ et $3^{ème}$ cas illustrent l'importance que l'on peut accorder à UT_0 et UT_1 l'un par rapport à l'autre. En fait, les fonctions Ψ_α permettent d'affirmer un *point de vue* sur leur importance réciproque.

7.5.2.5 Anticipation de comportements observés précédemment

Le principe d'anticipation de S.A.G.A.C.E., en vue de déterminer quel comportement déjà observé est le plus susceptible d'être reproduit, repose principalement sur l'utilisation qui est faite de la fonction S.U.C.R.E.

Dans une situation donnée, les classeurs du S.C. d'anticipation sont parcourus pour être appariés (phase de déclenchement des classeurs). Les classeurs déclenchés déposent leur(s) messages d'action sur la liste des messages avec les S.U.C.R.E. correspondants (les actions des classeurs du S.C. d'anticipation, rappelons-le, correspondent à ce que l'adversaire est censé faire).

Les classeurs sont regroupés par actions identiques et on crée une liste ordonnée d'actions (l'ordre décroissant du tri est fixé par la somme s_S.U.C.R.E. des S.U.C.R.E. des classeurs correspondant à chaque action de la liste).

Dans cette liste, on retient les n premières actions telles que la somme des s_S.U.C.R.E. représente une proportion donnée de l'ensemble. La proportion fixée est un paramètre de S.A.G.A.C.E. Cette proportion est appelée « taux de S.U.C.R.E. » et est notée τ. Elle peut être maintenue constante ou peut être modifiée dynamiquement en fonction des performances de l'anticipation (cf. section suivante).

Les actions retenues déterminent la (ou les) action(s) anticipée(s) de l'adversaire.

Quand une seule action est suffisante pour atteindre le taux de S.U.C.R.E. τ, elle définit de façon non ambiguë la prédiction du système. Un message correspondant à cette action est émis de l'interface de sortie du S.C. d'anticipation vers le C.S. stratégique. Quand plusieurs actions sont nécessaires, autant de messages de sortie sont émis par l'interface de sortie du S.C. d'anticipation vers le S.C. stratégique.

Certaines règles du S.C. stratégique peuvent être « tolérantes » vis-à-vis du nombre d'actions anticipées. C'est le cas lorsque la partie *prédiction* des règles est un intervalle (ou un ensemble) et que toutes les actions anticipées y sont inclues (cf. §7.5.3.).

Exemple d'Alésia

Supposons la présence dans le C.S. d'anticipation des classeurs suivants avec les paramètres correspondants (dans cet exemple, $\Psi_0 = 1$, $\Psi_1 = 1$ *et* $\tau = ½$ et la sélection des classeurs stratégiques se fait avec la méthode élitiste pure).

Condition			***Action***	***UT_0***	***UT_1***	***S.U.C.R.E.***
Nb points	*Nb points adv.*	*Position jeton*	*Mise*			
15	*16*	*2*	*15*	*45.516*	*12.416*	*57.932*
15	*16*	*2*	*13*	*17.953*	*16.928*	*34.881*
15	*16*	*2*	*2*	*2066*	*2064*	*4.130*

Supposons la présence dans le C.S. stratégique des classeurs suivants :

Condition			***Prédiction***	***Action***	***fitness***
Nb points	*Nb points adv.*	*Position jeton*	*Mise de l'adv.*	*Mise*	
16	*15*	*-2*	*15*	*15*	*10,12*
16	*15*	*-2*	*14*	*14*	*18,64*
16	*15*	*-2*	*[12..15]*	*16*	*9,45*

Dans la situation suivante :

S.A.G.A.C.E. va anticiper le fait que son adversaire va miser 15 points car le S.U.C.R.E. du premier classeur du S.C. d'anticipation suffit à dépasser τ (½) . Dans ce cas, le coup de S.A.G.A.C.E. va être 15 car la fitness du premier classeur du S.C. stratégique est la plus forte parmi celles des classeurs déclenchables (le premier et le troisième).

Si le choix de τ avait été différent, le choix de S.A.G.A.C.E. aurait pu l'être également. Imaginons τ = ¾, alors les deux premiers classeurs du S.C. d'anticipation auraient été nécessaires pour l'atteindre et deux messages d'anticipation auraient été déposés sur l'interface d'entrée du S.C. stratégique : un indiquant un coup estimé à 15 (1^{er} classeur), l'autre un coup estimé à 13 ($2^{ème}$ classeur). Dans ce cas, seul le $3^{ème}$ classeur du S.C. stratégique aurait été déclenché et S.A.G.A.C.E. aurait joué 16 points.

7.5.2.6 Les bases de métaconnaissances

L'utilisation de UT_0 et UT_1 permet, suivant l'observation du comportement de l'adversaire d'ajuster le modèle qu'on se fait de lui, selon qu'il adapte sa propre stratégie uniquement en fonction des coups qu'il joue ou des coups qu'il joue et de leurs conséquences. Pour cela, on ajuste les fonctions Ψ_α. Les méthodes d'ajustement sont définies par les métaconnaissances du S.C. d'anticipation.

Quand une anticipation s'est révélée erronée (l'action de l'adversaire est différente des actions anticipées) et quand l'action de l'adversaire faisait partie des actions initialement présentes dans la liste des actions anticipées (avant élimination des actions de faible s_S.U.C.R.E. n'ayant pas été nécessaires pour atteindre le taux de S.U.C.R.E.), il est possible d'ajuster dynamiquement les Ψ_α jusqu'à ce que cette action devienne indispensable pour atteindre le taux de S.U.C.R.E..

Nous avons implémenté cette technique pour le jeu ALESIA, mais sans pouvoir affirmer qu'elle a donné des résultats significativement meilleurs que lorsque les fonctions Ψ_α restent fixes.

Le taux de S.U.C.R.E. est un paramètre qu'il est également possible d'ajuster dynamiquement. Comme dans le cas précédent, si une action a été présente dans la liste et si elle n'a pas été retenue pour la détermination des actions anticipées, il est possible d'augmenter le taux de S.U.C.R.E. jusqu'à ce qu'elle le soit.

En implémentant cette méthode nous avons parfois constaté que le taux de S.U.C.R.E. devenait quasiment égal à 1. Dans ce cas, la plupart des prédictions devenaient exactes (l'action future de l'adversaire faisait partie des actions anticipées) mais devenaient en même temps inexploitables (parce que beaucoup trop nombreuses).

Ces observations nous ont conduit à imaginer deux approches :

La première consiste à baisser le taux de S.U.C.R.E. proportionnellement à un certain coefficient infSUC, quand l'action réelle de l'adversaire fait partie des (n-1) actions anticipées (n étant le nombre total d'actions anticipées), et à l'augmenter proportionnellement à un autre coefficient supSUC lorsque l'action de l'adversaire n'en fait pas partie (infSUC et supSUC ont été pris égaux réciproquement à 0.95 et 1.05 dans nos différentes implémentations). En règle générale, le taux de S.U.C.R.E. se stabilise pour les joueurs humains (la valeur dépend de chaque implémentation et de chaque joueur) et pour les stratégies adverses déterministes, mais oscille (autour de la valeur 1) pour des adversaires aléatoires.

L'autre méthode consiste à « débrayer » le S.C. d'anticipation lorsqu'il fournit trop d'actions anticipées. Concrètement, cela consiste simplement à signaler au S.C. stratégique qu'il n'a pas été possible de déterminer le comportement présumé de l'adversaire. Dans ce cas, seuls les classeurs du C.S. stratégique ne prenant pas en compte les messages du C.S. d'anticipation (partie prédiction vide) peuvent être déclenchés.

7.5.2.7 Anticipation de comportements jamais observés précédemment

Quand S.A.G.A.C.E. a observé (et enregistré) un coup de l'adversaire dans une situation originale (jamais rencontrée précédemment) dans laquelle l'adversaire a perdu, deux situations sont envisageables :

1. L'adversaire est déterministe et S.A.G.A.C.E. (qui a cette idée de lui) peut anticiper tout simplement qu'il rejouera le même coup si la situation se représente.

2. L'adversaire est adaptatif et S.A.G.A.C.E. (qui le sait) anticipe qu'il jouera différemment si la situation se représente.

Dans ce dernier cas, le modèle permet simplement d'écarter une possibilité quand la situation se représente.

La généralisation permet parfois de trouver une situation analogue dans laquelle l'adversaire a joué différemment et de se baser dessus mais ce n'est pas systématique. La méthode utilisée par S.A.G.A.C.E. est, quand cela est possible d'utiliser la généralisation ou bien d'utiliser le modèle d'un adversaire similaire (cf. §7.5.3.1.2.).

7.5.3 L'interface entre les deux systèmes de classeurs

7.5.3.1 Arbitrage entre les types de classeurs du S.C. stratégique

Comme nous l'avons signalé à maintes reprises, les classeurs du S.C. stratégique sont de deux types :

- Les classeurs n'utilisant pas l'anticipation du S.C. d'anticipation (partie prédiction vide) ;
- Les classeurs utilisant l'anticipation du S.C. d'anticipation (partie prédiction renseignée).

La plupart du temps, durant l'appariement, des classeurs des deux types sont déclenchés. L'arbitrage pour la sélection se fait en fonction de la fitness des classeurs (méthode élitiste pure, de la roue, etc.).

En règle générale, les classeurs utilisant l'anticipation ne sont renforcés (positivement ou négativement) que lorsque les prédictions correspondantes se sont révélées exactes. Le fait est que ces classeurs, quand ils sont créés par imitation ou introspection, ont rarement une faible fitness (il est souvent aisé de déterminer le choix à effectuer lorsque l'on connaît -ou suppose connaître- le coup de l'adversaire).

Ainsi, tant que le S.C. d'anticipation n'est pas débrayé, S.A.G.A.C.E. utilise principalement les classeurs utilisant l'anticipation.

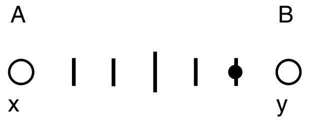
Si le S.C. d'anticipation est « débrayé » (parce que ses conclusions sont insatisfaisantes), seuls les classeurs n'utilisant pas l'anticipation sont déclenchés.

7.5.3.1.1 Piston mathématique

Dans un contexte de jeu à information complète et imparfaite, certains coups sont des coups gagnants. Cela signifie que, quel que puisse être le coup de l'adversaire, un coup particulier peut être optimal.

Exemple d'Alésia

Dans la situation suivante :



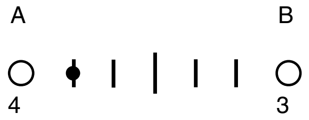
Si x est supérieur à y, il suffit que le joueur A joue (y+1) points pour gagner. Quel que soit le coup de B, ce coup de A est gagnant.

Il est souvent possible de déterminer l'ensemble des coups gagnants dans un jeu à information complète et imparfaite par un algorithme itératif.

Par ailleurs, il est également possible de déterminer les coups non-perdants : il s'agit des coups garantissant au moins le nul quel que puisse être le coup de l'adversaire.

Exemple d'Alésia

Dans la situation suivante :



Il suffit que le joueur A joue 3 points pour éviter de perdre la partie (si B a joué seulement 1 point, le jeton aura avancé d'un cran vers la droite et il restera 1 point à A et 2 points à B, insuffisants pour gagner).

Remarque :

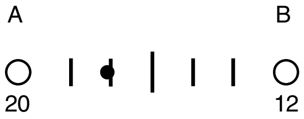
Les coups gagnants (réciproquement non-perdants) sont liés les uns aux autres. Certains coups sont gagnants à long terme si le joueur continue de jouer les coups gagnants subséquents (et réciproquement pour les coups non-perdants).

Exemple d'Alésia

Les classeurs suivants codent des coups gagnants :

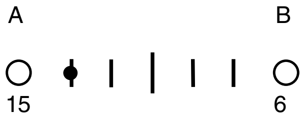
Condition			***Action***
Nombre points	*Nombre points de l'adversaire*	*Position jeton*	*Mise*
8	*6*	*1*	*2*
9	*6*	*0*	*1*
20	*12*	*-1*	*5*

Il convient cependant de noter que ces classeurs ne sont gagnants que si les coups suivants sont également effectués en suivant l'ensemble des coups gagnants. Les deux premiers classeurs donnés en exemple sont facilement « traçables » (on peut simuler à la main tous les coups suivants pour comprendre pourquoi ils sont gagnants). En revanche, le pouvoir gagnant du troisième est plus difficile à comprendre :

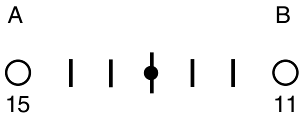


Si A joue 5 points, les deux coups optimaux possibles de B sont 6 (pour avancer d'un cran vers la gauche) ou 1 (pour économiser des points). Ce qui amène aux situations possibles suivantes :

❶



❷



Dans la situation ❶ , le classeur gagnant « suivant » préconise de jouer 6 points. Dans la situation ❷, le classeur gagnant « suivant » préconise de jouer 3 points. En continuant l'investigation, on découvre que, effectivement, la série de coups gagnants mène à la victoire et que B ne peut rien faire.

De la même façon, les classeurs suivants codent des coups non-perdants :

Condition			***Action***
Nombre points	*Nombre points de l'adversaire*	*Position jeton*	*Mise*
15	*12*	*-1*	*5*
22	*26*	*2*	*14*
22	*26*	*2*	*22*

Les deuxième et troisième classeurs sont non-perdants mais ils ne conduisent pas à la même série de coups subséquents.

L'intégration de tels classeurs dans S.A.G.A.C.E. rend son expertise encore plus efficace. Il est évident que certains de ces classeurs pourraient être découverts par le système (Algorithme génétique par exemple), mais leur introduction systématique accélère considérablement l'apprentissage.

Par ailleurs, quand d'autres classeurs sont générés par algorithme génétique ou par généralisation à partir de ces classeurs, ils sont souvent très efficaces. De tels classeurs ne sont ni gagnants ni non-perdants, mais n'en restent pas moins très adaptés au jeu contre des adversaires humains.

Le formalisme des systèmes de classeurs permet facilement l'implémentation de stratégies sur plusieurs coups : il suffit que les conséquences d'un coup soient intégrées dans la partie *condition* du classeur suivant (dans la suite des classeurs d'une stratégie). C'est ainsi que sont découvertes (ou introduites) des stratégies gagnantes faisant intervenir une suite de coups.

7.5.3.1.2 Expertise multi-joueurs et reconnaissance automatique d'adversaire

S.A.G.A.C.E. développe une base de modélisation (base de règles du S.C. d'anticipation) par adversaire identifié. En effet, chaque adversaire est identifié par son nom et les bases de modélisation sont chargées en fonction de chaque adversaire.

Il est ainsi possible de comparer les bases de modélisation pour :

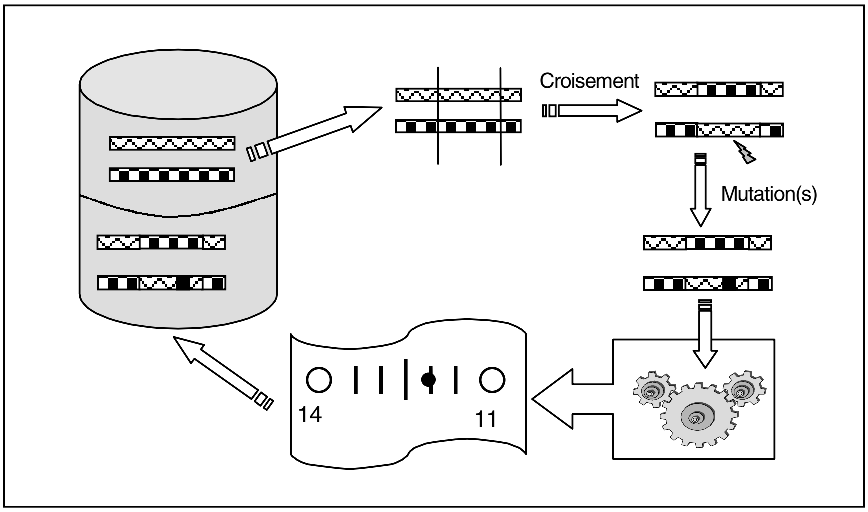
- Etablir des profils de joueurs ;
- Généraliser ces profils.

De la même façon, les bases de règles du S.C. stratégique peuvent être individualisées ce qui personnalise l'apprentissage à chaque adversaire et permet de généraliser des expertises stratégiques contre des profils de joueurs.

Nous avons réalisé quelques expérimentations consistant à créer une base de règle du S.C. stratégique en « piochant » parmi les règles les plus efficaces (fitness élevées) de différentes bases correspondant à différents joueurs. Les résultats ont été convainquants. En revanche, nous avions pensé être en mesure d'identifier un adversaire anonyme en comparant la base de modélisation correspondante avec d'autres bases correspondant à des parties précédentes mais les résultats n'ont été que partiellement satisfaisants (reconnaissance à 65% seulement).

Nous envisageons cependant de poursuivre notre recherche dans cette direction tant les enjeux nous paraissent importants : pouvoir identifier un profil de joueur rapidement permettrait d'utiliser la base de règles du S.C. stratégique existante (parmi des bases génériques de profil) la plus adaptée.

7.5.4 Entraînement du système (génération de situations)



S.A.G.A.C.E. intègre un module de génération automatique de situations de jeu qui fonctionne tel un sparring-partner pour accélérer l'amorçage du système. La fonction de ce module est d'entraîner le S.C. stratégique en le confrontant à des situations susceptibles de d'activer les classeurs nouvellement créés dans le but d'en estimer la valeur sélective (la fitness).

Ce module permet d'exploiter les différentes méthodes de création de classeurs (algorithme génétique, généralisation, imitation, etc.) puis de valider leur utilité rapidement et sans l'intervention d'un expert humain.

Il est clair qu'un système de classeurs classique valide les classeurs créés par expérience mais, dans le cas de jeux contre des adversaires humains, cela nécessiterait un grand nombre de parties propres à décourager les meilleures volontés[1].

Le système d'entraînement consiste à opposer S.A.G.A.C.E. à différents adversaires prédéterminés (adversaires probabilistes, adversaires adaptatifs – apprentissage par renforcement par exemple -, adversaires déterministes, etc.). Les oppositions n'ont pas seulement lieu sur des parties complètes, mais surtout sur des situations de jeu particulières adressant des parties *condition* des classeurs nouvellement créés.

---

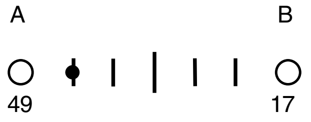
[1] Nous avons toutefois réussi (sous la menace) à persuader quelques victimes d'effectuer plusieurs centaines de parties

Les situations sont générées automatiquement à partir des parties *condition* des classeur. Une procédure simple vérifie simplement que les situations générées sont cohérentes du point de vue du jeu.

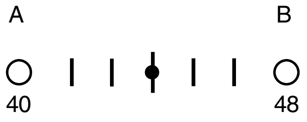
Exemple d'Alésia

Les situations suivantes ne sont pas cohérentes :

❶



❷



❶ ne l'est pas parce que, si A a 49 points c'est qu'un seul coup a été joué et, dans ce cas, le jeton ne peut avoir été déplacé de deux pas.

❷ ne l'est pas parce que, pour que le jeton soit au centre, il faut que les deux coups joués (B ayant 48 points, cela signifie que seulement deux coups ont été joués au maximum) et aient été nuls - impossible car A a dépensé plus de deux points - ou qu'un coup ait conduit le jeton dans un sens et l'autre dans le sens inverse – impossible car B a dû jouer 1 point à chaque coup (il est impossible qu'un seul coup ait été joué puisque cela voudrait dire que les deux joueurs ont joué 2 points).

Normalement, c'est à la création des classeurs qu'il convient de vérifier que leur partie *condition* est cohérente (ce qui rendrait impossible la génération de situations fictives incohérentes) mais les méthodes de création pourraient être défaillantes. Enfin, la présence de classeurs dans le S.C. stratégique avec des parties *condition* incohérentes ne peut avoir de conséquences fâcheuses pour le système, dans la mesure ou, par définition, ils ne sont jamais déclenchés. Cependant, le temps passé à en estimer la valeur sélective est du temps perdu et leur utilisation pour la création de nouveaux classeurs peut être à l'origine de la création d'autres classeurs incohérents.

La méthode de vérification des situations repose sur la méthode précédemment décrite d'élimination de règles (§7.5.1.3.).

8 Expérimentations

Nous avons réalisé plusieurs série d'expérimentations en implémentant S.A.G.A.C.E. pour les différents jeux à information complète et imparfaite que nous avons décrits. Pour chaque jeu, nous avons confronté S.A.G.A.C.E. à différents types de joueurs. La réalisation de programmes capables de jouer à des jeux à information complète et imparfaite a motivé de nombreuses recherches et donné lieux à diverses approches [Meyer, 99]. L'adversaire privilégié de S.A.G.A.C.E. est l'humain, cependant, nous avons comparé les résultats de S.A.G.A.C.E. à ceux de certaines de ces approches.

S.A.G.A.C.E. se compose de différents modules ou méthodes originales et il est souvent difficile de discerner avec exactitude la contribution de chaque module. C'est d'autant plus difficile que certains modules peuvent être « débrayés » par le système en fonction des performances globales. Les différentes méthodes originales intégrées à S.A.G.A.C.E. ne peuvent être évaluées indépendamment les unes des autres. La plupart des résultats présentés dans cette section concernent donc S.A.G.A.C.E. dans son ensemble. Cependant, en ce qui concerne les méthodes de création de règles, nous avons tenté d'illustrer l'intérêt de chacune à travers l'implémentation particulière la plus adaptée : « SunTzu » : en effet, dans cette implémentation le débrayage des modules de création de règle est laissé à l'appréciation de l'utilisateur, alors qu'il est automatique pour les autres applications.

8.1 S.A.G.A.C.E. pour SUNTZU : Méthodes de créations de règles

Pour estimer l'intérêt de chaque méthode de création de règles, S.A.G.A.C.E. a été confrontée à six types de joueurs :

Un joueur simpliste (simplet100) initialisé avec une centaine de règles de comportement déterminées par l'« expertise » d'un joueur humain. Ce joueur n'utilise aucune méthode d'adaptation. Dans chaque situation de jeu, il effectue ses choix aléatoirement (uniformément) parmi les règles applicables ;

Un autre joueur simpliste (simplet300) du même type que le précédent mais possédant 300 règles au lieu de 100 ;

Un joueur (Attrib100) possédant 100 règles de comportement et capable d'en ajuster les valeurs sélectives en fonction des résultats de leurs applications grâce à une méthode simple d'attribution de crédits positifs ou négatifs ;

Un joueur du type précédent mais possédant 300 règles (Attrib300) ;

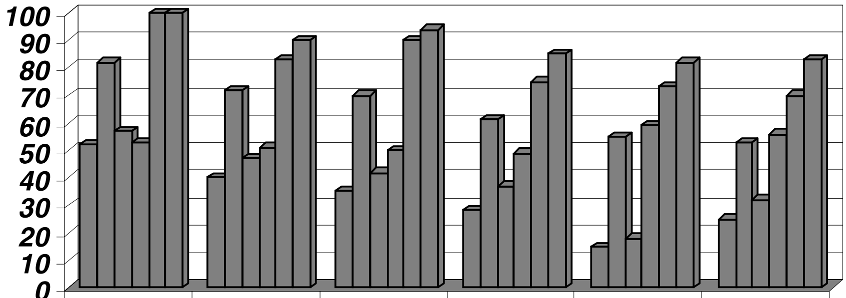
Un joueur basé sur un algorithme particulier de type MinMax (MinMax) adapté à ce type de jeu. Ce joueur détermine son choix en fonction de toutes les actions possibles de son adversaire (fussent-elles stupides) en tentant de minimiser ses pertes. L'ordre de ses trois actions est déterminé aléatoirement ;

Un joueur humain (Humain)[1].

Le tableau suivant précise le pourcentage de parties gagnées par S.A.G.A.C.E. contre ses différents adversaires en fonction des méthodes de création de règles utilisées. L'intitulé « Toutes » correspond à la possibilité offerte à S.A.G.A.C.E. d'utiliser toutes les méthodes simultanément.

Dans ces expérimentations, dix séries de 300 parties ont été disputées contre chaque adversaire informatique et la moyenne des résultats est présentée. Les parties contre des joueurs humains ont concerné deux joueurs seulement et pour 100 parties uniquement. Le système S.A.G.A.C.E. utilisait un système d'apprentissage par attribution positive et négative de crédit pour ajuster la valeur sélective de ses règles (initiales et créées). Le système possédait d'origine les même 100 règles que celle du joueur Simplet100.

Les différentes méthodes de création de règles ont été présentées précédemment mais chaque implémentation nécessite des ajustements particuliers, qu'il serait fastidieux de détailler ici.



	Simplet100	Simplet300	Attrib100	Attrib300	MinMax	Humain
■ *Aucune*	*52*	*40*	*35*	*28*	*15*	*25*
■ *Algo. Gén.*	*82*	*72*	*70*	*61*	*55*	*53*
■ *Généralisation*	*57*	*47*	*42*	*37*	*18*	*32*
■ *Imitation*	*53*	*51*	*50*	*49*	*59*	*56*
■ *Regrêts*	*100*	*83*	*90*	*75*	*73*	*70*
■ *Toutes*	*100*	*90*	*94*	*85*	*82*	*83*

---

[1] Rappelons que S.A.G.A.C.E. a été conçue et développée pour affronter des joueurs humains. Cependant, ses confrontations contre des joueurs informatiques permettent d'illustrer certains de ses aspects particuliers ou originaux.

8.1.1 Algorithme génétique

La méthode de création de règles par algorithme génétique est efficace. Elle permet au système S.A.G.A.C.E. de remporter 82% des parties contre un joueur déterministe possédant les mêmes règles initiales que lui, et surtout, 70% des parties contre un joueur qui non seulement possède les mêmes règles initiales que lui mais encore utilise exactement la même méthode d'apprentissage pour l'ajustement de la valeur sélective de ses règles.

Contre un joueur humain, cette méthode permet au système S.A.G.A.C.E. de remporter 55% des parties, ce qu'il faut comparer aux résultats du système quand il ne crée aucune règle (il remporte 25% des parties contre un joueur humain). L'amélioration est surprenante (plus de 25%), surtout compte tenu du fait que seulement 100 parties ont été disputées contre les joueurs humains, laissant peu de « temps » au système S.A.G.A.C.E. pour ajuster la valeur sélective des règles créées.

8.1.2 Généralisation

La méthode de création de règles par généralisation des règles existantes, si elle est relativement performante, ne s'avère pas aussi efficace que celle de l'algorithme génétique pour cette implémentation. Cela s'explique notamment par le nombre réduit de règles initialement présentes dans le système S.A.G.A.C.E. (100). Quand tous les systèmes de création de règles sont utilisés simultanément, la généralisation permet de créer des règles qui s'avèrent très efficaces (ce qui se vérifie, après les séries de parties, en inspectant la valeur sélective des règles en fonction de leur méthode de création) et cette méthode de création participe pour beaucoup aux performances globales.

8.1.3 Imitation

La méthode de création de règles par imitation des règles observées chez un adversaire n'est performante que contre un adversaire lui-même performant. En première analyse, on pourrait suggérer que l'on apprend de bonnes règles en observant et imitant un bon professeur et, inversement, en observant un piètre adversaire. Cependant, on pourrait penser qu'un bon joueur ne devrait pas se laisser vaincre par un adversaire ne faisant qu'imiter ses propres stratégies.

Le fait est qu'à ce jeu, les joueurs humains qui se sont prêtés à l'expérimentation perdaient (en moyenne) contre un adversaire qui imitait leurs stratégies. En fait, imaginer une bonne stratégie à ce jeu ne signifie pas qu'on soit capable d'en trouver une parade et un joueur peut être désarmé devant ses propres stratégies utilisées par un adversaire.

En ce qui concerne le joueur « MinMax », ce dernier s'est révélé être le meilleur adversaire de S.A.G.A.C.E. après les joueurs humains. Les performances de la méthode de création de règles par imitation contre cet adversaire s'expliquent simplement : le joueur « MinMax » possède des règles de comportement dont le but est de minimiser ses pertes. Un adversaire qui imite son comportement apprend également petit à petit à minimiser ses pertes. Le léger avantage du système S.A.G.A.C.E. quand il utilise uniquement cette méthode réside dans le fait qu'il est également capable par apprentissage (ajustement des valeurs sélectives des règles créées) de définir un ordre préférable d'application des règles, contrairement au joueur « Minmax » qui choisit un ordre aléatoire.

8.1.4 Regrets

La méthode des regrets est sans aucun doute la plus performante des méthodes lorsqu'elles sont utilisées isolément.

Contre des adversaires déterministes (Simplet100 ou Simplet300), l'explication est simple : la méthode des regrets crée après chaque coup une règle codant le meilleur coup qu'il aurait fallu jouer dans ces circonstances. Or, quand une situation semblable se présente à nouveau, par définition, l'adversaire déterministe rejoue le même coup, alors que le système S.A.G.A.C.E. joue la meilleure réponse correspondante.

Contre des adversaires utilisant une méthode d'apprentissage par attribution positive ou négative de crédit (pour l'ajustement de la valeur sélective de leurs règles de comportement), l'efficacité de la méthode des regrets est toujours exceptionnelle. Elle l'est d'autant plus que l'adversaire possède peu de règles applicables simultanément.

En effet, une règle créée par la méthode des regrets reste toujours efficace contre un coup précis de l'adversaire correspondant à une de ses règles de comportement. Une fois que tous ses différents comportements (ses règles) ont été 'observés' par le système S.A.G.A.C.E., ce dernier en connaît toutes les parades qu'il a créées automatiquement. Le problème est uniquement de déterminer quelle règle de regret utiliser quand une situation du jeu correspond à plusieurs comportements précédemment observés chez l'adversaire (plusieurs coups différents de l'adversaire ont entraîné la création d'autant de règles de regret correspondantes par le système).

Rappelons que S.A.G.A.C.E. intègre un système d'attribution positive ou négative de crédits pour ajuster la valeur sélective de ses règles et notamment des règles qu'il crée par la méthode des regrets. Ainsi, si dans une situation particulière un comportement de l'adversaire a entraîné la création d'une règle de regret spécifique et si, dans cette même situation, l'adversaire n'adopte plus ce comportement, la valeur sélective de la règle de regret correspondante va diminuer parce qu'elle ne sera plus adaptée.

8.1.5 Combinaison des méthodes

La méthode de création de règles la plus efficace pour le système S.A.G.A.C.E. dans cette application s'est avérée être une combinaison de chacune des méthodes précédemment décrites.

Concrètement, après chaque coup, le système crée une règle avec chacune des méthodes de création implémentées (quand cela est possible[1]).

Remarque : La création est un terme parfois impropre dans la mesure où la règle éventuellement créée peut être déjà présente dans la base de règles. Dans ce cas, la règle n'est pas créée à nouveau, mais sa valeur sélective est mise à jour (cf. description des méthodes).

Quand toutes les méthodes sont utilisées simultanément, certaines d'entre elles s'avèrent bien plus efficaces que lorsqu'elles sont utilisées seules. C'est le cas de la méthode des algorithmes génétiques et de celle de la Généralisation.

---

[1] la méthode de création par imitation est utilisée seulement si le coup de l'adversaire a été efficace (pour lui). Il est en effet inutile d'imiter un comportement inefficace.

La méthode des algorithmes génétiques permet de transformer de bonnes règles créées par regret notamment en d'autres règles souvent très efficaces. La méthode de généralisation permet d'élargir le contexte d'application des règles créées par regret particulièrement. Il se trouve que les règles ainsi créées s'avèrent généralement très efficaces. En fait, la méthode des regrets permet de créer des règles par définition efficaces mais très spécifiques, tandis que les méthodes des algorithmes génétiques ou de la généralisation permettent d'adapter les règles ainsi créées.

Evidemment, il est malheureusement fréquent que les règles créées par algorithmes génétiques ou par généralisation soient inadaptées (inefficaces) et que leur utilisation diminue l'efficacité générale du système. C'est cette constatation qui a entraîné la réalisation d'un système de génération de situation pour validation des règles créées (cf. § 7.5.4).

8.2 S.A.G.A.C.E. pour ALESIA

8.2.1 Jeux contre un adversaire artificiel simple

Considérons deux joueurs identiques et très simples à l'origine (leurs bases de règles de comportement sont constituées uniquement de quelques règles simples). Si on laisse ces deux joueurs jouer ensemble un millier de parties, leurs performances seront très proches. En revanche, si un des deux joueurs est adaptatif par apprentissage (s'il peut modifier la fitness de ses règles en fonction des résultats de leurs applications), alors celui-ci prend un certain avantage (420 parties gagnées contre 310 - en moyenne -). D'un autre coté, tant qu'on n'autorise pas ce joueur à créer de nouvelles règles, ses performances restent modestes. Cela est dû au fait qu'il n'a pas assez de bonnes règles. Il peut arriver aussi que, parmi les règles créées génétiquement, s'en trouve une qui puisse être très adaptée à un adversaire simple particulier, auquel cas les performances sont excellentes. Toutefois, cette très bonne règle peut être tout à fait inadaptée à un autre adversaire (fut-il aussi simple que le précédent mais différent). D'autre part, la création de cette bonne règle est purement aléatoire, rien ne garantit sa création par le système ni que le nombre de règles à créer avant d'obtenir cette règle particulière restera faible.

La création de règles par les algorithmes génétiques n'est donc pas une solution satisfaisante (en tout cas pas suffisante).

L'utilisation de S.A.G.A.C.E. contre un adversaire simple et non-évolutif conduit à des résultats excellents. Très vite, le joueur S.A.G.A.C.E. modélise les règles de son adversaire. Bien évidemment, plus cet adversaire possède de règles plus la modélisation est longue, mais le résultat est toujours excellent. Le joueur SUN-TZU (qui possède de bonnes règles d'utilisation de la modélisation) qui utilise la méthode S.A.G.A.C.E. a ainsi gagné 997 parties à 1 contre le joueur IVAN, qui possède seulement une trentaine de règles et qui n'a pas la possibilité d'évoluer.

8.2.2 Adversaires probabilistes

Le principal problème rencontré lors de la programmation de ALESIA a été de trouver des stratégies efficaces pour lutter contre des joueurs jouant de façon purement aléatoire. Le joueur ALEA ne possède qu'une règle. Celle-ci lui fait miser, à chaque tour, un nombre compris entre 1 et 10. Un humain a tôt fait de remarquer que ALEA a un comportement particulier et qu'il suffit de jouer 11 à chaque coup pour le vaincre systématiquement. Si on fait jouer un joueur évolutif (mais n'utilisant pas S.A.G.A.C.E.), tel que ABEL contre ALEA, le nombre de parties gagnées par ABEL est, en moyenne, de 220 contre 200, résultat très décevant. Si on laisse ABEL créer des règles génétiquement, il arrive qu'il crée une bonne règle qui lui fait miser, par exemple, 13 points par tours. Cette règle se révélera être très efficace contre ALEA et lui permettra de gagner toutes les parties, mais seul le hasard gouverne la découverte de cette règle comme nous l'avons déjà souligné précédemment. Un long et fastidieux calcul (que nous ne reproduisons pas ici) permet d'évaluer à un milliard le nombre de règles simples et valides possibles pour ALESIA ; on a donc environ une chance sur 1 milliard de créer par hasard la règle qui fait miser 11 à chaque tour : c'est peu...

Si on autorise ABEL à utiliser S.A.G.A.C.E. alors, pendant les cinquante premières parties, il va en perdre environ 5, puis il va gagner les 450 suivantes pour un total de 490 à 5 !

La méthode S.A.G.A.C.E. permet de lutter très efficacement contre un adversaire ne s'en remettant qu'au hasard pour miser.

Nous avons fait jouer notre système contre des joueurs simulés ayant des stratégies probabilistes. Ces stratégies faisant intervenir le hasard viennent naturellement à l'esprit d'un humain cherchant à contrecarrer la modélisation dont il sait qu'il pourrait faire l'objet. Le premier de ces adversaires est un joueur probabiliste jouant aléatoirement un nombre compris entre un et le nombre de points qui lui reste.

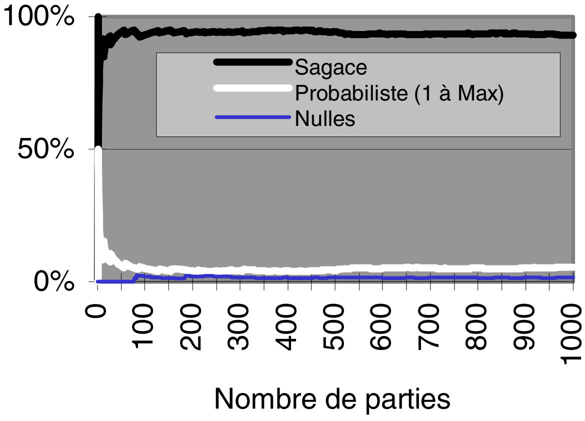


Figure 8.1. *Adversaire probabiliste (1 à Max).*

Très rapidement, le joueur Sagace parvient à générer une modélisation très fiable de cet adversaire et s'y adapte parfaitement (cf. figure 8.1).

Le second adversaire simulé est également probabiliste, mais il choisit une mise toujours comprise entre 1 et 50. Si la mise calculée est supérieure au nombre de points qui lui reste, il joue tous ses points.

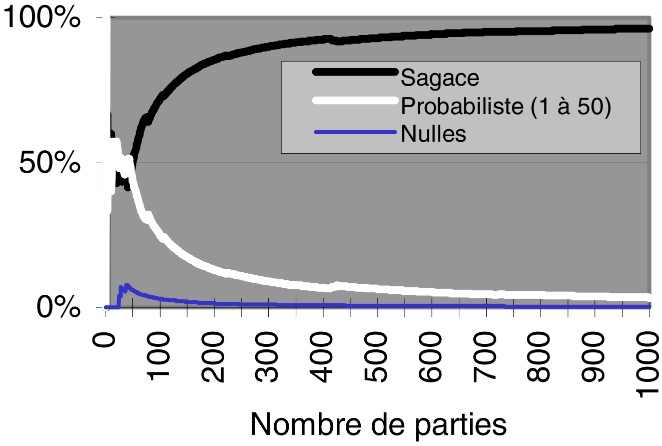


Figure 8.2. *Adversaire probabiliste (1 à 50).*

Ce joueur demande un temps plus long que le joueur précédent pour être modélisé, mais obtient finalement de moins bons résultats (cf. Figure 8.2). La modélisation est plus délicate (la convergence est moins rapide) parce qu'il n'y a pas équiprobabilité dans les choix d'action, les coups les plus hauts étant naturellement favorisés. Le modèle élaboré par le joueur Sagace est néanmoins très fiable après une cinquantaine de parties.

8.2.3 Adversaires théoricien

Le troisième adversaire simule un théoricien expert. Il est en effet capable de repérer mathématiquement quand une situation permet une stratégie qui assure la victoire et la joue dans ce cas. Sinon, son comportement est déterminé par un ensemble de règles probabilistes introduites après une expertise humaine du jeu (par exemple 'ne jamais jouer plus de un point de plus que ce qui reste à l'adversaire').

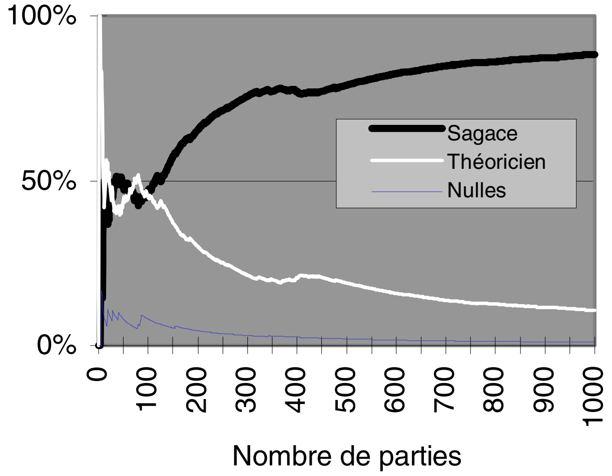


Figure 8.3. *Adversaire théoricien.*

Au moins 150 parties sont nécessaires au joueur Sagace avant de générer un modèle fiable de ce joueur (cf. figure 8.3). Avant, il y a une période pendant laquelle les joueurs se valent. Ensuite, pendant les 150 parties suivantes, le joueur Sagace se révèle très performant, puis le pourcentage de parties gagnées finit par se stabiliser. En continuant la série de parties, ce pourcentage atteint la valeur limite de 85%.

Le joueur utilisant S.A.G.A.C.E. adapte sa stratégie de telle façon qu'il évite de se retrouver dans des situations pour lesquelles il existe une stratégie gagnante pour son adversaire.

8.2.4 Adversaires adaptatifs

Dans un troisième type d'expérimentations, nous avons fait jouer notre système contre des joueurs simulés ayant des stratégies adaptatives. Un joueur humain dit adopter de telles stratégies.

L'adversaire simulé est un joueur qui fait évoluer ses stratégies en fonction de leurs performances. Il a été implémenté sous forme d'un système de classeurs. Ce joueur

adaptatif apprend de façon classique par rétribution positive et négative de ses règles de choix d'action en fonction de leurs performances [HOL 86].

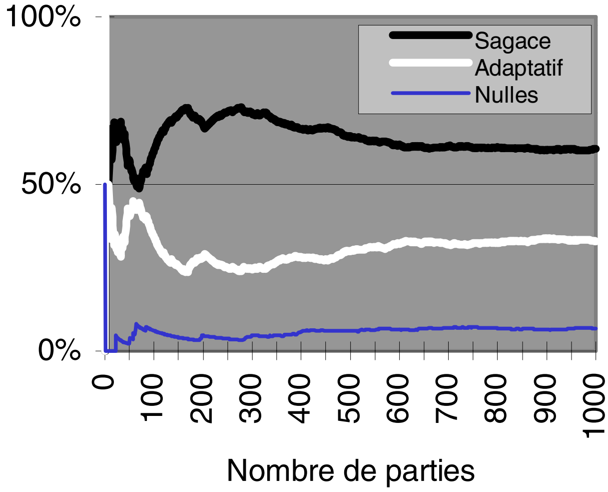


Figure 8.4. *Adversaire Adaptatif.*

Durant les cinquante premières parties, le joueur Sagace parvient relativement bien à modéliser le joueur adaptatif et son pourcentage de gain augmente (cf. figure 8.4). Pendant les cinquante parties suivantes, le joueur adaptatif s'adapte (!), puis le joueur Sagace parvient à déterminer un nouveau modèle cohérent et son pourcentage de gain augmente à nouveau. Après un certain nombre de telles oscillations 'adaptatives', le pourcentage de gain des deux joueurs se stabilise (60% pour le joueur Sagace, 33% pour le joueur adaptatif). On remarquera que ce type de joueur se révèle moins performant que les bons joueurs humains.

Quelle que puisse être la complexité d'un joueur artificiel programmé de cette façon, la méthode S.A.G.A.C.E. est toujours parvenue à créer une base de modélisation cohérente de ce joueur. En revanche, la durée de modélisation peut devenir très importante pour un joueur utilisant une stratégie mixte efficace et adaptative.

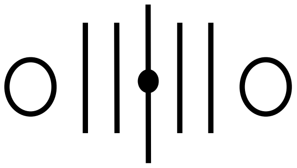
Il faut noter qu'un joueur qui a la possibilité de créer souvent de nouvelles règles grâce à la génétique peut devenir imprédictible mais, dans le même temps, ses performances chutent parce que les règles créées ne sont pas toutes efficaces, bien au contraire. Il est donc particulièrement difficile d'être à la fois performant et imprédictible et c'est pourquoi S.A.G.A.C.E. se révèle être si efficace.

8.2.5 Adversaires théoriciens adaptatifs

Nous avons effectué une étude théorique du jeu Alésia afin d'évaluer notre approche en la confrontant à une telle approche, nous appuyant sur les résultats et les méthodes de la Théorie des jeux. Nous avons créé un joueur artificiel capable d'utiliser les solutions du joueur théoricien tout en étant également capable d'apprentissage et d'adaptation.

Nous avons, dans un premier temps, élaboré un algorithme MinMax tout à fait classique. Il est évident qu'un tel algorithme ne peut prétendre rivaliser avec un joueur tel que S.A.G.A.C.E. parce que ce dernier est adaptatif tandis qu'un algorithme MinMax est câblé. Cependant, cet algorithme MinMax était un passage obligé vers un algorithme théorique adaptatif.

Comme l'impose ce type d'algorithme, nous avons conçu une fonction d'évaluation permettant d'établir, après parcours de l'arbre de décision, la stratégie optimale. Nous avons décomposé cette fonction d'évaluation en cinq sous-fonctions correspondant chacune à une des positions possibles du jeton.



Les cinq sous-fonctions prennent en compte la dualité position / capitaux restants. La forme générale des sous-fonctions est la suivante : $2/\pi \, \mathrm{ArcTan}(\mathrm{sgn}(x-y)(x-y)^2/100)$, x et y étant les capitaux des deux joueurs. Chacune des fonctions est bornée entre –1 et 1 et est ajustée par des coefficients particuliers en fonction de la position dont elles dépendent. A titre d'exemple, voici quelques courbes représentatives des ces fonctions.

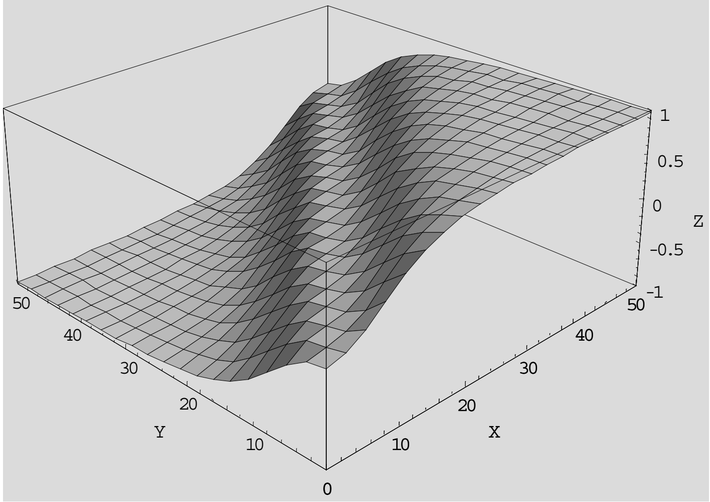


Figure 8.5. Courbe correspondant à la position centrale

Comme le montre la courbe de la figure 8.5, si les deux joueurs ont le même capital dans la position centrale, la fonction est nulle. Plus la différence entre les joueurs est importante, plus la fonction d'évaluation estime la position favorable.

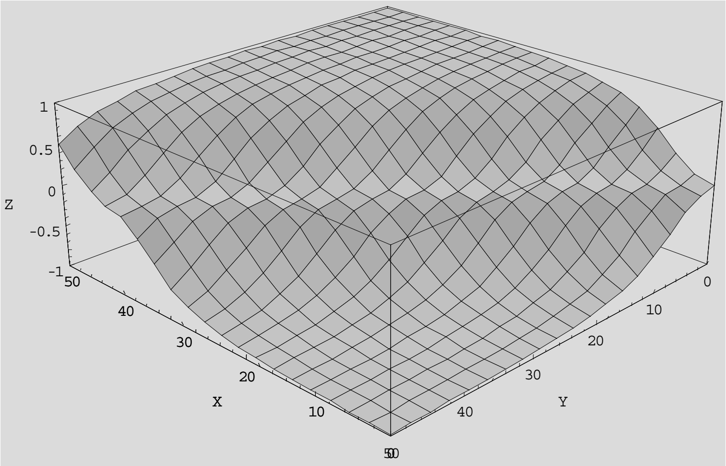


Figure 8.6. Courbe correspondant à un pas vers la citadelle adverse

Cette courbe (figure 8.6) n'est plus symétrique. Elle tient compte de l'avantage de la position par rapport à un éventuel déficit de capital.

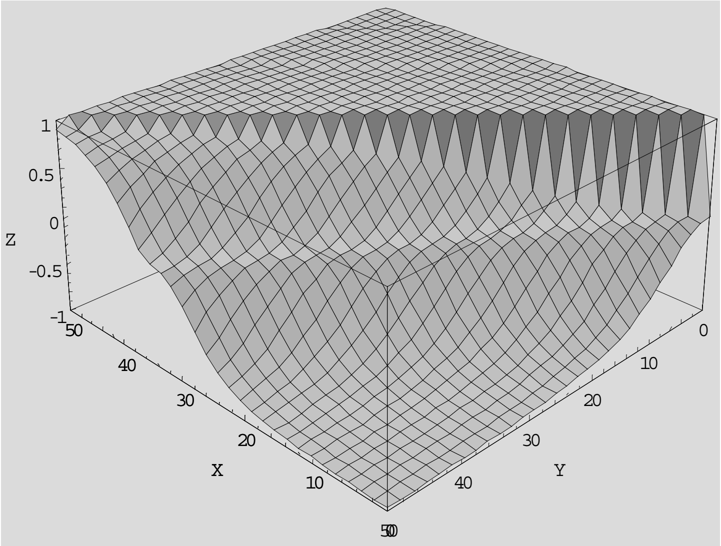


Figure 8.7. Courbe correspondant à un pas de citadelle adverse

Cette courbe (figure 8.7) tient compte du fait que, devant la citadelle adverse, si on possède plus de point que l'adversaire, on a gagné (il suffit de jouer un point de plus que le capital de ce dernier).

Le programme MinMax obtenu joue très correctement contre des adversaires peu adaptatifs, mais il s'est avéré très mauvais contre les autres (et plus particulièrement contre S.A.G.A.C.E. qui gagne rapidement toutes les parties). Son caractère déterministe suffit à expliquer les résultats.

Par la suite, nous sommes avons élaboré un programme capable de calculer systématiquement toutes les évaluations de toutes les positions possibles, en parcourant l'arbre jusqu'aux feuilles. Un sérieux problème est alors apparu rapidement dans la programmation. En effet, plus de 80% des positions sont susceptibles de conduire à une défaite ce qui implique que la valeur correspondante de MinMax est de –1. Dès lors, il n'est pas possible de choisir entre différentes stratégies possibles. Les algorithmes MinMax ne sont pas du tout adaptés à ce type de jeu. Nous avons alors développé un nouvel algorithme plus subtil qui prenait en compte à la fois les calculs des coefficients de MinMax mais aussi la moyenne de l'espérance de gain des coups possibles. Cet algorithme s'est avéré bien meilleur que le MinMax seul mais, étant toujours déterministe, il a été très décevant devant S.A.G.A.C.E.

Le programme que nous avons développé ensuite s 'appuie sur une adaptation adaptative de la Théorie des jeux, plus particulièrement sur une extension adaptative des équilibres de Nash.

Pour certains jeux, Il est parfois impossible de déterminer mathématiquement les répartitions probabilistes des stratégies mixtes correspondant à l'équilibre de Nash à cause de la puissance de calcul ou de mémoire nécessaires. Nous avons donc tenté de créer un programme adaptatif qui devrait apprendre à trouver ces coefficients pour le jeu ALESIA[1]. L'idée consiste à créer une structure capable de gérer les répartitions probabilistes de toutes les situations possibles du jeu et de permettre au programme d'ajuster ses coefficients en fonction de ses performances.

Nous avons longuement réfléchi à la façon d'initialiser un tel programme. Il paraît en effet illusoire d'espérer trouver, dans un temps raisonnable, tous les coefficients en partant de répartitions d'origine uniformes ; la convergence devant alors être beaucoup trop longue. Nous sommes parvenus à réutiliser les résultats des deux programmes précédents pour initialiser ce programme théorique adaptatif. Partant de répartitions probabilistes inspirées des évolutions des programmes précédents, le programme théorique adaptatif ajuste en permanence les répartitions en fonction de ses propres performances.

Ce programme s'est avéré redoutable contre tous les programmes que nous avions imaginés précédemment, c'est-à-dire contre tous les précédents adversaires de S.A.G.A.C.E. Ses résultats sont d'ailleurs meilleurs que ceux de S.A.G.A.C.E. dans certains cas (La convergence est plus rapide).

Nous avons alors tenté l'expérience qui s'imposait : l'affrontement entre le théoricien adaptatif et S.A.G.A.C.E.

Nous avons lancé deux séries de tests. La première a consisté à opposer le théoricien adaptatif à un joueur codé comme S.A.G.A.C.E. sous le formalisme d'un système de classeurs et pouvant apprendre, c'est à dire pouvant ajuster sa stratégie (la force de ses règles) mais n'utilisant pas l'Anticipation. Avant la série de parties, nous avons effectué une copie des bases de règles de ce joueur afin d'utiliser exactement les mêmes pour le joueur S.A.G.A.C.E. dans le but d'évaluer de façon indiscutable les conséquences des anticipations que réalise S.A.G.A.C.E. contre un tel adversaire.

---

[1] Pour le jeu ALESIA, nous avons calculé la stratégie mixte optimale au sens de Nash et le joueur correspondant fait l'objet de section suivante.

Adversaire : Nombre de parties :	Adaptatif / classeurs	Adaptatif / S.A.G.A.C.E.
100 premières parties	14 %	9 %
1000 suivantes	53 %	20 %
2000 suivantes	82 %	30 %

Pourcentages de parties gagnées par le théoricien adaptatif

Sur 1000 parties : le joueur théoricien adaptatif en a gagné 521, le joueur adaptatif à système de classeurs en a gagné 460 (d'où le score de 53 %).

Sur 1000 autres parties, cette fois contre S.A.G.A.C.E., le joueur théoricien adaptatif en a gagné 197 et S.A.G.A.C.E. 772 (d'où le score de 20 %).

Ces résultats sont fondamentaux, parce qu'ils valident notre approche de la résolution des jeux à information complète mais imparfaite quand il n'est pas possible de déterminer de façon mathématique les répartitions probabilistes correspondant à l'équilibre de Nash.

Nous ne prétendons pas pour autant que notre joueur théoricien adaptatif est le plus efficace qu'il soit possible de concevoir, mais ce joueur avait les moyens théoriques de trouver la solution optimale au sens de Nash, c'est à dire de trouver la stratégie garantissant, en moyenne, au moins le match nul. C'est d'ailleurs ainsi que ce joueur s'est révélé bien meilleur qu'un joueur adaptatif classique (système de classeurs et algorithmes génétiques).

En fait, le principe de fonctionnement de S.A.G.A.C.E. le rend très adaptatif, étant donné que son anticipation du joueur théoricien adaptatif modifie le propre comportement de celui-ci en temps réel. Ainsi, le joueur théoricien adaptatif ne parvient pas à converger vers la stratégie optimale.

Nous pensons que, sous réserve qu'il soit confirmé dans d'autres circonstances et sur d'autres jeux, ce résultat est fondamental. Il pourrait amener à changer la façon usuelle d'appréhender les applications de la Théorie des jeux.

Remarques :

Ce théoricien adaptatif adopte une forme évoluée de la stratégie du "fictitious play" de Brown et Robinson [Brown, 1951], [Robinson, 1951] notament décrit par Owen [Owen, 1995]. Si sa conception doit le mener naturellement à une stratégie optimale (répartition probabiliste en fonction des choix qu'elle a affectués) contre tout adversaire particulier, on constate que S.A.G.A.C.E. ralentit tellement sa convergence qu'elle n'a plus aucun intérêt pratique (contre S.A.G.A.C.E. du moins).

Nous avons également expérimenté un théoricien adaptatif particulier qui utilise différemment la répartition probabiliste qu'il fait évoluer en temps réel. En effet, il joue systématiquement la mise correspondant au coefficient 'probabiliste' le plus important. Ce joueur a de meilleures performances contre S.A.G.A.C.E. que le précédent durant les 1000 premières parties (25 % contre 20 %). En revanche, les performances s'équivalent après les 2000 parties suivantes (30%). Enfin, si on poursuit l'expérience avec 5000 nouvelles parties, ce nouveau joueur obtient 20 % de gain contre S.A.G.A.C.E (les performances baissent), alors que le théoricien adaptatif original parvient à 40 % (les performances augmentent et on s'approche du résultat optimal théorique correspondant à l'espérance de gain, à savoir 50 %, mais très lentement). Cette expérience corrobore notre intuition concernant la Théorie des Jeux : un joueur dont la stratégie consiste à découvrir les coefficients des répartitions probabilistes des équilibres au sens de Nash a des chances de rivaliser avec S.A.G.A.C.E. même si la convergence peut être très longue. En revanche, tous les joueurs adaptatifs classiques que nous avons implémentés n'ont jamais été très efficaces contre S.A.G.A.C.E. sans doute parce qu'ils n'ont aucun processus d'Anticipation consciente alors qu'une telle capacité est sans aucun doute décisive dans ce type de jeux. Nous ne prétendons cependant pas avoir étudié tous les adversaires possibles contre S.A.G.A.C.E. mais nous avons utilisé la plupart des techniques classiques.

8.2.6 Jeux contre NASH

Une approche encore plus classique de la programmation de jeux consiste à utiliser la stratégie optimale au sens de Nash quand il est possible de la calculer. L'approche décrite au paragraphe précédente étant une approche empirique de détermination de cette stratégie quand on est incapable de la déterminer par calculs.

Nous avons développé un algorithme permettant de jouer à ALESIA suivant la stratégie optimale au sens de Nash. Cet algorithme est basé sur la méthode du simplex (méthode mathématique d'optimisation linéaire) [Vajda, 1961], [Press, 1986], [Owen, 1995].

Par définition, cet algorithme garantit l'espérance de gain à ALESIA. Ce jeu étant à somme nulle, l'algorithme correspondant garantit que, en moyenne, le joueur qui l'utilise gagne au moins autant de partie qu'il en perd.

Cependant, contre un adversaire peu performant, cet algorithme obtient des résultats bien meilleurs. Contre les adversaires précédemment décrits, ce dernier s'est toujours révélé meilleur. Par contre, ses performances comparées à celles de S.A.G.A.C.E. contre ces mêmes stratégies ont presque toujours été moins bonnes :

- Contre le joueur probabiliste (1àMax) : 98% de parties gagnées, 1% de parties perdues ;
- Contre le joueur probabiliste (1à50) : 95% de parties gagnées, 3% de parties perdues ;
- Contre le théoricien : 60% de parties gagnées, 35% de parties perdues ;
- Contre le joueur adaptatif : 55% de parties gagnées, 45% de parties perdues ;
- Contre le théoricien adaptatif : 50% de parties gagnées, 45% de parties perdues ;
- Contre un joueur humain[1] : 45% de parties gagnées, 40% de parties perdues.

Les résultats des affrontements entre le joueur « Nash » et S.A.G.A.C.E. illustrent les facultés d'adaptation de ce dernier et amènent à s'interroger sur la convergence de la méthode S.A.G.A.C.E. (figure 8.8.) vers la stratégie optimale quand il joue contre un joueur qui l'utilise.

On peut considérer que le joueur « Nash » est un joueur de référence. Comme il est impossible de le vaincre en moyenne (gagner plus de partie que lui sur un nombre infini théorique de parties), on peut évaluer par comparaison les performances de ses adversaires.

Les 5% de gain d'écart entre le joueur « Nash » et un bon joueur humain se justifient par les inattentions ou les faibles capacités de calcul humaines.

Le joueur « Nash » ne tire aucun profit des parties précédentes, il n'a aucune représentation de ses adversaires et n'a aucune faculté d'adaptation. Son unique intérêt est la possibilité qu'il offre d'entraîner un joueur adaptatif. En effet, les méthodes de création de règles par imitation, par regrets et finalement par généralisation permettent à S.A.G.A.C.E. d'apprendre beaucoup en jouant contre un joueur tel que « Nash ». C'est en partie l'adaptation des règles ainsi créées par généralisation qui garantit un amorçage performant de S.A.G.A.C.E.

---

[1] Les résultats sont une moyenne concernant 5 joueurs humains pour 100 parties.

Si le S.C. d'anticipation est le meilleur atout de S.A.G.A.C.E., il n'est pleinement efficace qu'après un certain nombre de parties nécessaires à établir un modèle cohérent de ses adversaires. Pour les premières parties contre un nouvel adversaire, il est important que S.A.G.A.C.E. ait un niveau de jeu correct afin qu'il offre un minimum de résistance, forçant ses adversaires à jouer au mieux de leurs capacités. Ce niveau de jeu initial s'acquiert principalement dans des parties préliminaires contre des adversaires tel que « Nash ».

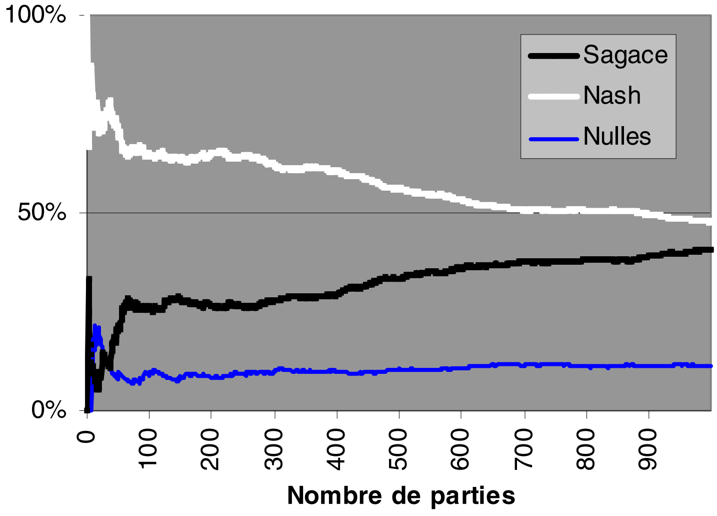


Figure 8.8. *Stratégie optimale (Nash).*

La figure 8.8 est représentative d'une série d'expérimentations ayant opposé S.A.G.A.C.E. et le joueur « Nash ».

Comme nous l'avons mentionné, la stratégie du joueur « Nash » n'est pas évolutive. Ainsi, les variations des courbes représentatives des performances comparées des deux joueurs ne sont dues qu'à l'évolution de la stratégie de S.A.G.A.C.E.

S.A.G.A.C.E. s'adapte de façon continue à « Nash ». Sa courbe de performance est linéaire et varie de 5% à 25% durant les cent premières parties. L'adaptation durant cette période est très rapide. Entre la 100ème et la 1000ème partie, l'amélioration des performances de S.A.G.A.C.E. est toujours linéaire mais beaucoup plus lente.

Dans plusieurs séries d'expériences, S.A.G.A.C.E. a convergé vers une stratégie satisfaisante offrant presque autant de victoires que « Nash ». Ainsi, pour la série correspondant à la figure 8.8., après environ 150 parties, S.A.G.A.C.E. a construit un modèle relativement cohérent de « Nash » dans les débuts de parties. Cela lui a permis d'utiliser une stratégie agressive (il jouait souvent entre 14 et 18 points au premier coup puis entre 10 et 15 points au second coup) payante. La plupart du temps, S.A.G.A.C.E. réussissait avec cette stratégie à se trouver à un pas de la victoire avec environ 25 points restants qu'il misait entièrement. Cela lui assurait approximativement autant de victoires qu'à « Nash ». Dans d'autres séries S.A.G.A.C.E. a convergé vers un ensemble de stratégies utilisées en alternance.

Dans aucune série d'expériences S.A.G.A.C.E. n'est parvenu à égaler « Nash ». Il est cependant toujours parvenu à réduire l'écart à moins de 10% en 1000 parties.

8.2.7 Jeux contre un adversaire humain

Comme cela s'imagine aisément, il est délicat de demander à un expérimentateur humain de jouer un millier de parties contre S.A.G.A.C.E. comme ce fut le cas pour les séries d'expériences entre joueurs informatiques. Pour cette raison, les séries comportaient 'seulement' cent parties. Par ailleurs, l'expérimentation n'a, à ce jour, porté que sur une dizaine de sujets.

Nous sommes convaincu qu'un humain est incapable de jouer totalement au hasard. Il y a toujours des schémas de pensée cycliques, ou des automatismes irraisonnés, qui orientent chaque joueur humain dans un style de stratégies qui lui est propre. S.A.G.A.C.E. parvient toujours à créer une base de modélisation d'un joueur humain, particulièrement lorsque celui-ci ne sait pas ce qu'est S.A.G.A.C.E. Quand le joueur humain est averti, celui-ci tente de jouer plus ou moins aléatoirement et sa modélisation est beaucoup plus difficile mais, comme pour un joueur artificiel évolutif, ses performances baissent au cours des parties.

Il semble très difficile pour un humain de créer une stratégie à la fois performante et difficilement prédictible. Le seul joueur ayant toujours eu des résultats acceptables (moins de deux tiers de parties perdues) est un joueur qui possède deux ou trois stratégies plus ou moins fixes et qui bascule de l'une à l'autre sans arrêt. Dès que ce joueur ne se concentre plus, il redevient prédictible ou inefficace.

S.A.G.A.C.E. s'avère être le meilleur joueur à ce jour, joueurs artificiels et humains confondus[1]. Nous avons conduit de nombreuses études concernant d'autres joueurs informatiques utilisant S.A.G.A.C.E. mais ne possédant que très peu de règles initiales (dans sa base comportementale). Contre un joueur humain, de tels adversaires se révèlent moins performants que le joueur SUNTZU qui possède de nombreuses règles initiales. En fait, il apparaît qu'un humain qui connaît le fonctionnement interne de S.A.G.A.C.E. parvient relativement bien à anticiper sur les actions de joueurs possédant peu de règles comportementales. Cela nous conforte dans l'idée qu'il est nécessaire de doter les joueurs informatiques de moyens d'anticiper sur leur propre comportement afin de modifier leur stratégie lorsqu'ils deviennent trop prévisibles. Il paraît donc indispensable de doter les joueurs artificiels de suffisamment de règles comportementales (ou de bons moyens d'en créer de nouvelles) afin qu'ils aient la possibilité de modifier leur stratégie en temps réel en utilisant différentes règles et non toujours les mêmes, aussi judicieuses soient-elles.

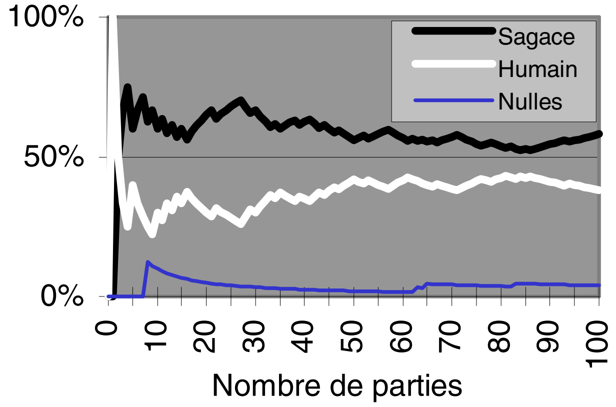


Figure 8.9. *Adversaires humains.*

Comme le montre la figure 8.9, les progressions du taux de parties gagnées par les joueurs ne sont pas monotones. Les deux joueurs semblent s'adapter au jeu l'un de l'autre sans arrêt. Après environ 85 parties, le joueur humain ne semble plus capable d'imaginer de nouvelles stratégies adaptées au joueur S.A.G.A.C.E. et le taux de parties gagnées par celui-ci augmente légèrement.

---

[1] Le classement des joueurs est effectué suivant le nombre de parties gagnées et suivant les scores obtenus. Si « Nash » est un joueur imbattable en théorie, les scores qu'il obtient contre ses adversaires sont toujours moins bons que ceux de S.A.G.A.C.E.

Des séries de parties contre différents humains et plus longues montrent que les courbes finissent par se stabiliser. En général autour des mêmes valeurs que celles de la Figure 8.9 (55% pour le joueur S.A.G.A.C.E. et 45% pour le joueur humain). Ce type de courbes est difficile à interpréter mais il souligne cependant la grande adaptabilité du jeu humain face à celui de la machine et réciproquement. Il montre également que l'intensité des variations diminue avec le nombre de parties.

8.2.8 Une série d'étude

Nous avons conduit une série particulière de 150 parties contre un très bon joueur humain pendant laquelle nous avons enregistré les performances de S.A.G.A.C.E. par rapport à l'utilisation faite du S.C. d'anticipation.

Les enregistrements correspondant font l'objet du tableau suivant :

	Gain du joueur humain	Gain de S.A.G.A.C.E.	Coups gagnés sans anticipation	Coups gagnés avec anticipation	Utilisation de l'anticipation	Prédictions exactes
Parties 1à50	26	19	32.40%	70.14%	14.24%	66.51%
Parties 51à100	15	30	35.47%	69.12%	26.20%	70.40%
Parties 101à150	14	34	37.10%	74.65%	38.65%	74.08%

Dans la première série de 50 parties, S.A.G.A.C.E. établit le modèle de son adversaire. Il effectue environ 67% de prédictions exactes. Quand S.A.G.A.C.E. utilise son modèle, il remporte 70% des coups joués mais le S.C. d'anticipation n'est engagé pour déterminer le coup à jouer que 14% du temps. Cela signifie que la plupart du temps (les 86% restant) le système considère que l'utilisation du modèle n'est pas pertinente sans doute parce que les prédictions sont trop imprécises[1].

---

[1] Les 67% de prédictions exactes du S.C. d'anticipation ne signifient pas, à eux seuls, que le modèle est fiable et que l'utilisation des prédictions pour déterminer le coup à jouer est pertinente. Ils signifient seulement que lorsque S.A.G.A.C.E. considère que le modèle est fiable, et qu'il est utilisé pour déterminer le coup à jouer, le S.C. d'anticipation fournit effectivement 67% de bonnes prédictions. Comme cela a été expliqué précédemment, la pertinence de l'utilisation du modèle dépend de la qualité des prédictions, c'est-à-dire de leur précision (prédire que l'adversaire va jouer entre 1 et 30 peut être une prédiction juste mais inutilisable).

Dans la seconde série, les prédictions sont meilleures et le S.C. d'anticipation est utilisé plus souvent pour déterminer le coup à jouer. Cela a pour conséquence d'augmenter nettement le pourcentage de coups gagnés (et par conséquent le nombre de parties gagnées). Par ailleurs, le pourcentage de coups gagnés sans utilisation de l'anticipation augmente également (mais légèrement). Cette augmentation est due en partie à l'apprentissage (adaptation de la valeur sélective des règles du S.C. stratégique en fonction de leur performance).

On constate que les résultats de cette série d'expériences au bout de 100 parties sont légèrement inférieurs à ceux obtenus en moyenne (49% pour S.A.G.A.C.E. contre 41% pour le joueur humain au lieu de 55% pour S.A.G.A.C.E. et 45% pour le joueur humain comme l'indique la figure 8.9). Il s'agit vraisemblablement d'un « caprice » des statistiques dû au nombre de parties nulles … Cependant l'important est la différence de gain entre S.A.G.A.C.E. et le joueur humain qui reste environ égale à 10%.

Dans la troisième série, les prédictions s'améliorent encore (¾ de prédictions exactes). Les prédictions sont également plus précises (cela n'est pas représenté dans le tableau) ce qui justifie l'utilisation accrue du S.C. d'anticipation (plus de 38%) pour déterminer le coup à jouer[1].

L'amélioration continue (depuis la 1ère partie jusqu'à la 150ème) du taux de coups gagnés lors de l'utilisation de l'anticipation est intimement liée à celui de prédictions exactes. En effet, les règles du S.C. stratégique qui utilisent les prédictions du S.C. d'anticipation sont, la plupart du temps, des règles qui font gagner le coup sous réserve que la prédiction soit juste (elles sont schématiquement du type « si je pense que l'adversaire va jouer ceci alors, jouer cela »).

Le fait que les deux taux ne soient pas exactement les mêmes s'explique par deux possibilités : Il est possible qu'une prédiction fausse n'empêche pas de gagner un coup (par exemple, si la prédiction est « l'adversaire va jouer 10 », que le système joue un de plus soit 11 et qu'en réalité l'adversaire a joué 5, le coup est tout de même gagné) ou, plus rarement, qu'une prédiction juste n'empêche pas d'en perdre un (par exemple si la prédiction est « l'adversaire va jouer 10 », que cela déclenche une règle stupide -créée par un mécanisme quelconque- faisant jouer 9, et si la prédiction est juste, le coup est tout de même perdu).

---

[1] Les prédictions du S.C. d'anticipation sont utilisées pour le déclenchement de règle d'anticipation du S.C. stratégique.

Le joueur S.A.G.A.C.E. s'avère redoutable contre un joueur humain. Ses adversaires se déclarent systématiquement surpris par ses capacités d'anticipation[1].

8.2.9 Création de règles

L'utilisation de S.A.G.A.C.E. a permis au joueur SUNTZU d'élaborer certaines stratégies très efficaces en modélisant ses différents adversaires humains. Il a ainsi créé les règles suivantes :

« Quand l'adversaire a 9 points, qu'il me reste entre 5 et 50 points et que le jeton est situé à 1 pas de sa citadelle, jouer au hasard entre 1 et 5 points ».

Cette règle, garantit au joueur de ne pas perdre la partie (un simple calcul permet de le vérifier).

« Quand l'adversaire a 9 points, que j'en ai au moins 10 et que le jeton est situé à une case de sa citadelle, jouer 10 points »

Cette règle, garantit la victoire. Quand elle est applicable, elle doit être appliquée de préférence (notamment par rapport à la précédente). En fait, une telle règle (qui garantit la victoire) a toujours une grande valeur sélective qui la fera appliquer en priorité.

« Quand un adversaire a entre 1 et 5 points, qu'il me reste au moins 5 points de plus que lui, et que, enfin, le jeton est situé à un ou deux pas de sa citadelle, jouer 5 points ».

Cette règle est assez simple et elle garantit la victoire de SUNTZU si ses conditions d'applications sont vérifiées.

« Quand l'adversaire a 37 points, que j'en ai au moins 25 et que le jeton est situé au centre (à trois pas de la citadelle adverse), jouer 7 points ».

Cette règle sibylline (avant une série de calculs) garantit la victoire si les règles utilisées après elle sont du même type (optimales). Cela signifie qu'il existe après le coup correspondant (jouer 7 points) une stratégie gagnante quels que soient les choix de l'adversaire.

S.A.G.A.C.E. permet ainsi de « découvrir » des règles très efficaces ainsi que des stratégies complexes (sur plusieurs coups) implicitement.

---

[1] D'après ces résultats, il semblerait que S.A.G.A.C.E. ait repris à son compte, en la modifiant légèrement, la célèbre phrase de Terence (Carthage -190/-159 Av J.-C.) : « non Homo sum, humani nihil a me alienum puto ».

8.3 S.A.G.A.C.E. pour « Pierre / Ciseaux / Papier »

Dans cette application, S.A.G.A.C.E. a été opposée à deux types de joueurs particuliers :

- Un joueur utilisant l'algorithme de Minasi [Minasi, 1991] ;
- Des joueurs humains.

La confrontation contre le joueur utilisant la stratégie optimale au sens de Nash n'a aucun intérêt dans cette application. En effet, cette stratégie consiste à jouer aléatoirement de façon uniforme (1/3 ; 1/3 ; 1/3) et garantit de ne pas perdre, en moyenne, plus de parties que de parties gagnées quelle que soit la stratégie de l'adversaire. Dans « ALESIA », le joueur « optimal au sens de Nash » pouvait remporter, contre un adversaire peu « doué », bien plus de parties en moyenne que contre un adversaire « doué » (cela est du au fait qu'une partie comporte plusieurs coups). Dans « Pierre / Ciseaux / Papiers », la stratégie (1/3 ; 1/3 ; 1/3) donne toujours les mêmes résultats en moyenne contre toute stratégie adverse.

8.3.1 L'algorithme de Minasi

Cet algorithme est extrêmement simple. Il repose sur une analyse des patterns composés par les coups successifs des deux joueurs. Dans cet algorithme, un pattern est une séquence de paires de coups (un coup par joueur). Pendant une partie, l'ensemble des paires de coups est enregistré en continu :

Ex : R s P r P r S r P s P r R r.

→ Avec R pour Rock (Pierre) ; S pour Scissors (Ciseaux) et P pour Paper (Papier).

Les coups en majuscules correspondent aux coups du joueur « Minasi », les coups en minuscules à son adversaire.

Quand le système doit effectuer son choix pour le coup en cours, il recherche dans l'enregistrement la plus longue série de coups (en partant des derniers coups) répétée au moins une fois qui correspond à la situation actuelle. Cette série est utilisée pour déterminer le coup à jouer.

Exemple :

Si l'enregistrement est le suivant :

P r P r R p R s P r P p S r R r P r S r R p S s R s R r P r

En partant de la fin de la séquence (les coups les plus récents) l'algorithme effectue la recherche de la façon suivante :

Coups suivant Séquence	**r**	**p**	**s**	
r	5	3	0	❶
P r	2	2	0	❷
r P r	1	1	0	❸
R r P r	1	0	0	❹

La ligne ❶ correspond à la recherche du pattern 'r' : Il a été localisé 8 fois dans l'enregistrement (la dernière, n'ayant pas de successeur, ne compte pas). 5 fois il a été suivi d'un 'r' et 3 fois d'un 'p'.

La ligne ❷ correspond à la recherche du pattern 'P r' : Il a été localisé 4 fois dans l'enregistrement. 2 fois il a été suivi d'un 'r' et 2 fois d'un 'p'.

La ligne ❸ correspond à la recherche du pattern 'r P r' : Il a été localisé 2 fois dans l'enregistrement. 1 fois il a été suivi d'un 'r' et 1 fois d'un 'p'.

La ligne ❹ correspond à la recherche du pattern 'R r P r' : Il a été localisé 1 seule fois dans l'enregistrement et a été suivi d'un 'r'.

Comme le pattern 'R r P r' n'a été localisé qu'une fois, il est impossible de trouver une séquence répétée la plus grande qui soit suivie éventuellement d'un autre coup. La recherche s'interrompt et statue sur le fait que l'adversaire va jouer 'r'. L'algorithme de « Minasi » va donc choisir de jouer 'P'. Si deux séries répétées statuent sur deux coups différents, l'algorithme indique de jouer aléatoirement.

L'algorithme de « Minasi » suppose que ses adversaires réitèrent un choix quand les circonstances sont approximativement les mêmes.

« Minasi » ne tient aucun compte des conséquences des choix effectués par les deux joueurs. Il n'adapte pas sa stratégie en fonction des résultats quelle obtient et ne considère pas que l'adversaire puisse le faire pour lui-même.

« Minasi » possède, en quelque sorte, un modèle de son adversaire, consistant uniquement en l'enregistrement des coups. Contre un adversaire déterministe (objet de plusieurs séries d'expérimentations que nous avons faites) ce modèle est idéal et « Minasi » offre d'excellents résultats. Etrangement, ce modèle est suffisant pour vaincre (en moyenne) presque systématiquement un humain sur une série de 100 parties.

8.3.2 « Minasi » contre un adversaire humain

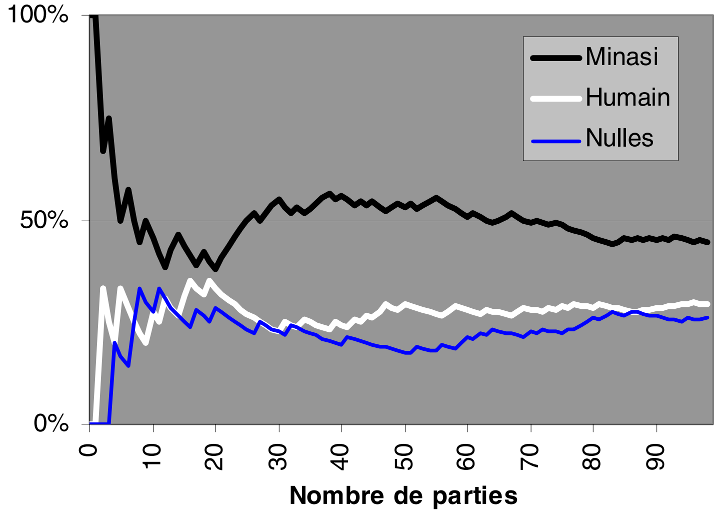


Figure 8.10. « Minasi » contre des adversaires humains.

La figure 8.10 illustre de façon représentative les séries d 'expériences de jeux opposant « Minasi » à des joueurs humains. Pendant une vingtaine de parties, les joueurs humains parviennent à élaborer une stratégie efficace (consistant à modifier sans arrêt leurs séries de coups). Ensuite, la plupart des joueurs humains se révèlent incapables d'exhiber des comportements qui n'entrent pas dans des patterns exploitables efficacement par « Minasi ».

Cependant, il est fréquent qu'un joueur humain se décide (après une série de longueur très variable) à jouer au hasard. Souvent, ses performances augmentent alors pendant une courte période (une vingtaine de parties), comme cela est le cas sur la figure 1. entre les parties 30 et 50. Mais, systématiquement, « Minasi » finit par gagner plus souvent.

Si un joueur humain était capable de jouer uniformément aléatoirement, il jouerait la stratégie optimale au sens de Nash et gagnerait, par définition, autant qu'il perdrait (en moyenne). Ces séries d'expériences corroborent notre intuition qu'un humain est inapte à générer du hasard consciemment.

Quoi qu'il en soit, l'algorithme de Minasi, aussi simple soit-il, s'avère meilleur que la plupart des adversaires que nous lui avons fait rencontrer (déterministes et humains).

8.3.3 S.A.G.A.C.E. contre un adversaire humain

Nous avons implémenté une version simplifiée de S.A.G.A.C.E. pour jouer au jeu de « Pierre / Ciseaux / Papier ». Cette version ne crée pas les règles de comportement de ses adversaires par observation, elle utilise simplement l'enregistrement des parties.

De la même façon, elle ne possède pas de base de règles censée mettre à profit les prédictions pour déterminer le choix à faire dans la mesure où ce dernier est évident : il ne nécessite que trois règles simples qu'il est inutile de chercher à faire découvrir par le système et auxquelles il est également inutile de chercher à attribuer une valeur sélective forte quand elles sont appliquées alors que les prédictions sont bonnes[1].

Ces trois règles sont les suivantes :
Si l'adversaire va jouer 'r' (pierre) alors jouer 'P' (papier) ;
Si l'adversaire va jouer 's' (ciseaux) alors jouer 'R' (pierre) ;
Si l'adversaire va jouer 'p' (papier) alors jouer 'S' (ciseaux).

La différence fondamentale entre « Minasi » et S.A.G.A.C.E. tient dans le fait que S.A.G.A.C.E. prend explicitement en considération les résultats des coups précédents. S.A.G.A.C.E. utilise différemment (de « Minasi ») l'enregistrement des coups précédents.

---

[1] Cf. § 7.5.1.6.2. sur les particularités de S.A.G.A.C.E. en ce qui concerne l'évaluation de la fitness des classeurs en fonction de la justesse des prédictions.

Les paramètres UT_0 et UT_1 de la méthode S.A.G.A.C.E. sont utilisés conjointement à l'enregistrement pour déterminer le prochain coup supposé de l'adversaire. Le coefficient UT_0 correspond en pratique aux calculs effectués par « Minasi ». Le coefficient UT_1 concerne le succès des coups de l'adversaire. Il permet de tenir compte du fait qu'un humain est adaptatif et qu'il ne va sans doute pas réitérer une série de coups qui s'est précédemment soldée par une défaite.

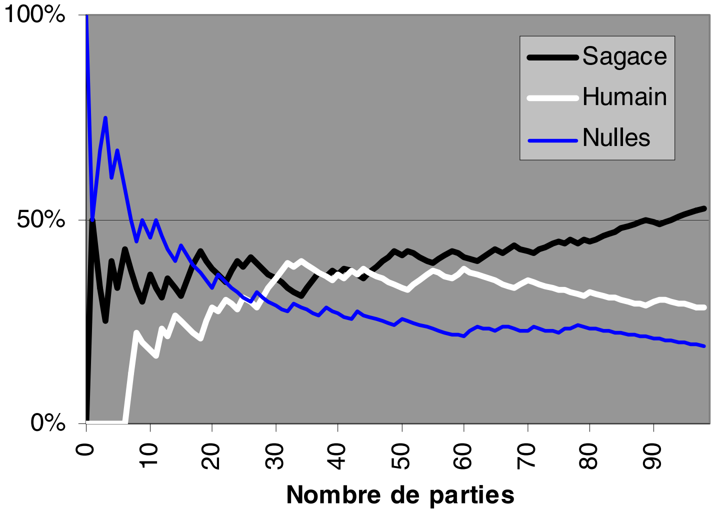


Figure 8.11. S.A.G.A.C.E. contre des adversaires humains.

Comme le montre la figure 8.11, S.A.G.A.C.E. est souvent moins efficace contre un humain que ne l'est « Minasi » pendant les premières parties (une vingtaine). Nous supposons que la raison principale de cette différence est liée au fait que les joueurs humains, en début de partie, jouent de façon presque automatique, sans se soucier des résultats qu'ils obtiennent. Généralement, un joueur humain réfléchit de plus en plus au fur et à mesure des coups et finit ainsi par prendre en compte le succès ou l'échec de ses stratégies. Il devient alors plus facile à modéliser par S.A.G.A.C.E.

La plupart du temps, au bout de 50 parties, les performances de S.A.G.A.C.E. augmentent de façon très significative. A partir de ce moment, S.A.G.A.C.E. devient beaucoup plus performant que « Minasi » contre un joueur humain.

L'intégration d'un modèle basé sur le type des coups précédents, mais également sur leur éventuel succès, semble ainsi justifiée pour un jeu aussi simple que « Pierre / Ciseaux / Papier ».

Il faut noter que, si les parties continuent (au-delà de 100), les gains des deux joueurs se stabilisent (autour de 50% pour S.A.G.A.C.E. et 30% pour l'humain).

Par ailleurs, certains joueurs humains parviennent mieux à « tromper » S.A.G.A.C.E. en jouant de façon plus ou moins aléatoire[1]. Dans ce cas, ils parviennent à stabiliser les gains à des valeurs moins importantes pour S.A.G.A.C.E. (4 à 8% de moins).

8.3.4 S.A.G.A.C.E. contre « Minasi »

Nous avons réalisé certaines expériences afin de comparer les méthodes S.A.G.A.C.E. et « Minasi » utilisées l'une contre l'autre.

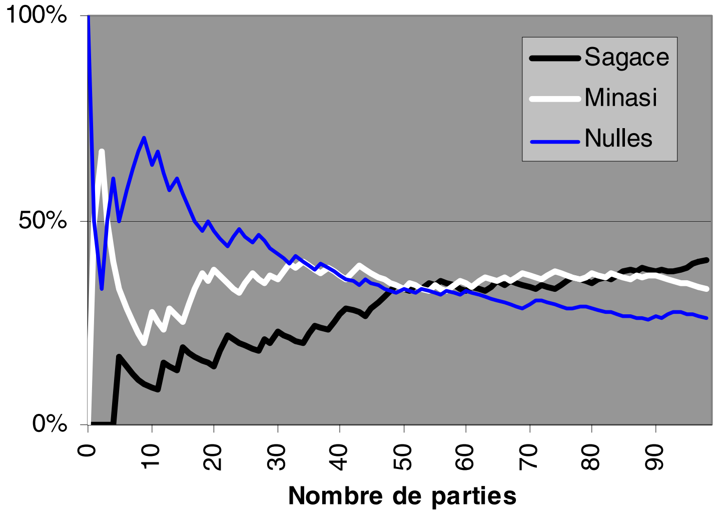


Figure 8.12. S.A.G.A.C.E. contre « Minasi ».

Les premières parties sont rarement significatives. Parfois S.A.G.A.C.E. est largement meilleure, parfois « Minasi » l'est et, parfois, ce sont les parties nulles qui sont majoritaires (comme c'est le cas sur la figure 8.12).

Par contre, comme l'illustre la figure 8.12, après une quarantaine de parties, S.A.G.A.C.E. et « Minasi » sont à peu près aussi performantes (la figure 8.12 est représentative de l'ensemble des expériences menées).

---

[1] Le programme autorise le joueur à effectuer son choix en tapant sur une des trois touches du clavier ('R','S' ou 'P'). Le traitement est suffisamment rapide pour que si le joueur tape très vite sur le clavier, chaque pression de touche déclenche un coup. En fait, le joueur ne réfléchit absolument pas, il se contente de frapper au hasard les trois touches correspondantes du clavier. Ainsi entrevue, la question de la faculté de « générer du hasard » pour un humain se pose différemment selon qu'il s'agisse de générer du hasard « physique » (en manipulant des objets) ou du hasard « intellectuel » (en réfléchissant). Des expériences que nous avons menées en utilisant notre programme, il semblerait qu'un humain réussisse à générer plus facilement du hasard « physique » que du hasard « intellectuel ».

Il se trouve que S.A.G.A.C.E. est presque systématiquement (plus de 90% des séries de parties) légèrement meilleur « Minasi ». Ce résultat est particulièrement intéressant parce qu'il signifie que S.A.G.A.C.E. est capable (au moins en partie) d'anticiper les coups de « Minasi » en lui prêtant un niveau de « conscience » qu'il n'a pas. En effet, S.A.G.A.C.E. effectue ses prédictions sur la base du succès éventuel des coups de l'adversaire alors que ce dernier (« Minasi ») n'en tient aucun compte.

8.4 S.A.G.A.C.E. pour « Pair / impair » (ou « Matching Pennies »)

Dans les années 50, Hagelbarger[1] inventa une machine capable avec un réel succès de deviner le comportement d'un joueur au jeu « Pair / impair ». La machine en question remporta environ 53% des 10.000 parties engagées contre des joueurs humains. Cet écart de 6% entre les performances de la machine et celles des joueurs humains est remarquable sur une série aussi longue.

Peu de temps après, Shannon présenta sa propre machine qui s'avéra encore meilleure que celle de Hagelbarger [Shannon, 1953]. Il faut noter que ces machines étaient des machines physiques et mécaniques conçues avec les moyens électroniques de l'époque.

Nous avons implémenté S.A.G.A.C.E. pour la réalisation d'un programme jouant au jeu « Pair / impair » afin d'évaluer ses performances vis-à-vis de celle de Shannon. Nous avons également adapté et implémenté l'algorithme de Minasi pour ce jeu. Enfin, nous avons réalisé quelques programmes correspondant à des joueurs utilisant des stratégies aléatoires mixtes et à des joueurs utilisant des automates à états finis.

8.4.1 L'algorithme de Shannon

L'algorithme de Shannon (désormais noté « Shannon »), à l'instar de celui de Minasi, est basé sur l'identification de patterns dans le comportement de son adversaire. De la même façon que celui de Minasi, cet algorithme suppose qu'un adversaire va réitérer un comportement effectué dans des circonstances semblables. Cependant, d'une part « Shannon » n'enregistre pas tous les coups (il a une mémoire de 8 patterns uniquement) et, d'autre part, il tient compte (comme S.A.G.A.C.E.) du succès ou de l'échec des coups de son adversaire.

Le type de patterns considérés par « Shannon » concerne le succès (ou l'échec) de deux coups successifs et l'attitude de l'adversaire en fonction de ces deux coups (a-t-il persévéré ou changé de stratégie).

---

[1] D. H. Hagelbarger, Bell Laboratories.

Il existe donc huit situations possibles :

- Le joueur gagne, rejoue la même chose et gagne à nouveau. Il peut alors jouer la même chose ou changer.
- Le joueur gagne, rejoue la même chose et perd. Il peut alors jouer la même chose ou changer.
- Le joueur gagne, joue différemment et gagne à nouveau. Il peut alors jouer la même chose ou changer.
- Le joueur gagne, joue différemment et perd. Il peut alors jouer la même chose ou changer.
- Le joueur perd, rejoue la même chose et gagne. Il peut alors jouer la même chose ou changer.
- Le joueur perd, rejoue la même chose et perd à nouveau. Il peut alors jouer la même chose ou changer.
- Le joueur perd, joue différemment et gagne. Il peut alors jouer la même chose ou changer.
- Le joueur perd, joue différemment et perd à nouveau. Il peut alors jouer la même chose ou changer.

Ces huit situations correspondent aux huit mémoires de « Shannon » qui contiennent uniquement deux informations :

La première indiquant si le joueur a joué la même chose ou changé sa stratégie la dernière fois que chacune de ces situations s'est présentée.

La seconde indiquant si cette attitude a été répétée à l'issue de la dernière situation antérieure identique.

« Shannon » joue en fonction du contenu des huit mémoires. Si la seconde information indique que l'attitude de l'adversaire a été répétée, « Shannon » considère qu'il va de nouveau répéter son attitude et joue le coup correspondant. Si l'information indique que l'attitude de l'adversaire n'a pas été répétée, « Shannon » joue au hasard (uniformément « pair » ou « impair »).

Pour ces séries d'expérimentations, nous sommes parvenus à persuader des joueurs humains d'effectuer des séries de 500 parties.

8.4.2 « Shannon » contre des joueurs humains

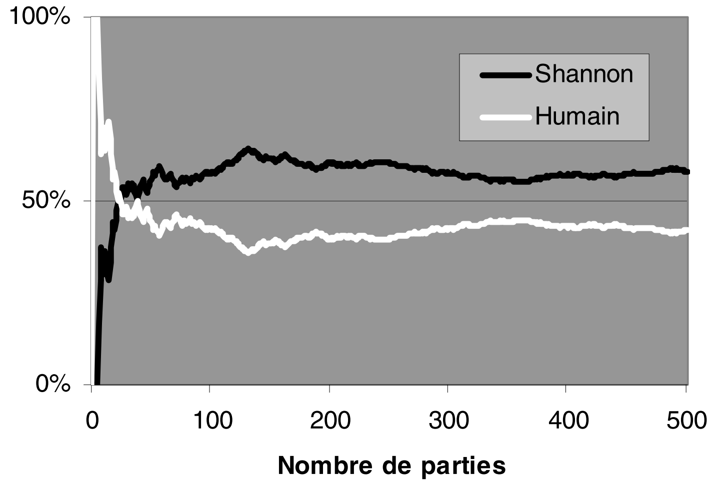


Figure 8.13. « Shannon » contre des joueurs humains

Il faut environ 40 parties à « Shannon » pour que le modèle (ses mémoires) qu'il tient à jour de son adversaire soit valide et exploitable (Figure 8.13). En moyenne, le taux de parties gagnées par « Shannon » atteint 58%. Il faut toutefois noter que « Shannon » a, en moyenne, recours au hasard 55% du temps.

Lorsque « Shannon » n'utilise pas le hasard, il obtient 65% de prédictions justes.

Les résultats sont meilleurs que les 53% évoqués plus haut. Il est possible que, si les séries d'expériences avaient permis d'engager 10.000 parties, nous aurions retrouvé ce résultat. En effet, après 500 parties, le taux de parties gagnées par « Shannon » baisse très légèrement de façon presque linéaire.

8.4.3 « Minasi » contre un adversaire humain

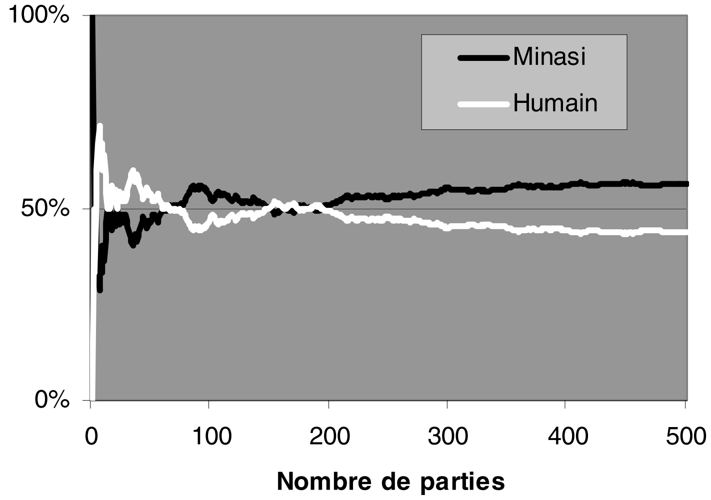


Figure 8.14. « Minasi » contre des adversaires humains.

La modélisation de « Minasi » est plus longue (figure 8.14). Le modèle devient efficacement exploitable au bout de 200 parties ! Ensuite, le taux de parties gagnées par « Minasi » atteint, en moyenne, 55%.

La « lenteur » de la modélisation est sans doute due au fait que les patterns ne sont constitués que de séries de deux valeurs possibles (« pair » ou « impair ») ce qui rend d'éventuelles périodes détectables moins caractéristiques.

Par ailleurs, contre « Minasi », les joueurs humains finissent presque systématiquement par jouer au hasard dès les premières parties. Il se trouve que les humains semblent éprouver moins de difficultés à « générer du hasard uniforme » sur deux valeurs (« Pair » et « impair ») que sur trois (comme dans le jeu « Pierre / Ciseaux / Papier ») ou plus. Cela rend le modèle plus délicat à « construire ».

En moyenne, contre des adversaires humains, « Minasi » recourt au hasard 10% du temps. Lorsque ce n'est pas le cas (qu'il n'utilise pas le hasard), il obtient 56% de prédictions justes. C'est inférieur aux résultats des prédictions de « Shannon ».

8.4.4 **S.A.G.A.C.E. contre un adversaire humain**

Pour cette application, S.A.G.A.C.E. a été implémentée de la même façon que pour le jeu « Pierre / Ciseaux / Papier ».

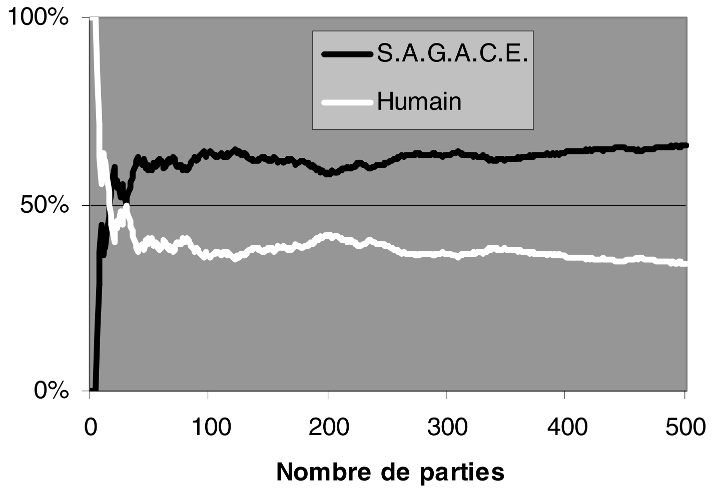


Figure 8.15. S.A.G.A.C.E. contre des adversaires humains.

S.A.G.A.C.E. s'adapte très rapidement contre un joueur humain (figure 8.15). Il prend un ascendant conséquent sur son adversaire après seulement 30 parties. En moyenne, S.A.G.A.C.E. remporte 66% des parties. C'est beaucoup plus que « Shannon » ou que « Minasi ». S.A.G.A.C.E. utilise le hasard 20% du temps. Quand ce n'est pas le cas, il obtient environ 70% de prédiction exactes.

L'éventuel recours au hasard pour un joueur humain est moins efficace contre S.A.G.A.C.E. que contre « Minasi ». Cela paraît surprenant mais le modèle qu'effectue S.A.G.A.C.E. est différent de celui de « Minasi ». Le fait que S.A.G.A.C.E. prenne en considération le succès ou l'échec des coups de ses adversaires fait que les patterns extraits de l'enregistrement par S.A.G.A.C.E. sont différents de ceux de « Minasi ». En particulier, ils sont plus courts et concernent des coups plus récents.

Nos convictions sur la mémoire à court terme des humains nous laissent à penser que, si les processus cognitifs mis en jeu par un humain pour « générer du hasard » utilisent cette mémoire à court terme, cela constitue une explication plausible aux meilleures performances de S.A.G.A.C.E. que « Minasi » contre des joueurs humains tentant de jouer au hasard simplement parce que S.A.G.A.C.E. considère des événements plus récents que ceux que considère « Minasi ».

« Shannon » s'avère bien moins efficace que S.A.G.A.C.E. mais possède une mémoire bien inférieure. Le fait que cette mémoire soit sans doute aussi grande que la mémoire à court terme des humains est la seule similarité qu'on puisse trouver entre elles. La mémoire à court terme des humains concerne, a priori, les événements les plus récents. Les mémoires de Shannon peuvent concerner des événements très anciens. En effet, imaginons qu'un joueur humain se comporte ainsi : il joue « pair » et perd, rejoue « pair » et perd à nouveau et renouvelle cette série peu de temps après. Dans ce cas, la mémoire correspondante deviendra exploitable. Il est néanmoins possible que le joueur ne se comporte à nouveau suivant ce schéma que très longtemps après et la mémoire correspondante ne sera exploitée qu'à ce moment (alors que l'humain n'aura, sans doute, aucun souvenir de ces agissements et ne les intégrera donc pas dans son processus de décision).

8.4.5 Prédiction et hasard

Des trois méthodes comparées (« Shannon », « Minasi », S.A.G.A.C.E.) S.A.G.A.C.E. est celle qui fournit les prédictions les plus fiables (70%). « Shannon » fournit des prédictions plus fiables que « Minasi » (65% contre 55%) mais utilise plus souvent le hasard.

Dans le jeu « Matching pennies », le fait de recourir uniformément au hasard correspond à la stratégie optimale (50% d'espérance de gain). Ainsi, quand « Shannon » recourt au hasard 55% du temps, il gagne, en moyenne, autant de parties qu'il en perd. Le fait d'avoir tant recours au hasard n'est donc pas pénalisant par rapport à un autre joueur (humain notamment).

Pour un autre jeu, pour lequel la stratégie optimale n'est pas connue (et pour lequel le recours au hasard pourrait être inadéquat s'il est fait de façon uniforme), l'utilisation de l'algorithme de Shannon nous parait plus hasardeuse. En effet, pour le jeu « Matching pennies », quand Shannon joue au hasard, il le fait uniformément (il choisit « pair » ou « impair » en jetant une pièce de monnaie) or cette stratégie est optimale au sens de Nash. Dans un jeu ou la stratégie au sens de Nash est inconnue, Shannon ne pourrait pas recourir judicieusement au hasard puisqu'il ne saurait pas suivant quelle répartition probabiliste il faudrait le faire.

Parfois il est plus intéressant de prendre le risque d'exploiter une prédiction hasardeuse que de s'en remettre au hasard. C'est pour cela que certains paramètres de la méthode S.A.G.A.C.E. permettent d'ajuster les conditions d'utilisation de la modélisation en fonction de la confiance que le système lui accorde.

8.4.6 Martingale

Après une analyse mathématique de son algorithme, Shannon a démontré qu'il existait une martingale permettant de le battre suivant un ratio de 3:1. Pour cela, il faut connaître l'état des huit mémoires de l'algorithme et effectuer certains calculs.

Quand on fait jouer « Minasi » et « Shannon » l'un contre l'autre, ils sont, en moyenne, ex æquo. Quand S.A.G.A.C.E. affronte « Shannon », S.A.G.A.C.E. gagne en moyenne 57% des parties. C'est beaucoup moins que la martingale de Shannon.

On en conclura que S.A.G.A.C.E. n'est pas une méthode mathématique et qu'elle peut se révéler moins efficace que d'autres méthodes quand elles sont confrontées l'une à l'autre. Cependant, rappelons que l'adversaire privilégié de S.A.G.A.C.E. est le joueur humain.

8.4.7 Autres séries d'expériences

Nous avons comparé les trois méthodes précédemment décrites quand elles sont opposées deux à deux et opposées à d'autres stratégies originales. Nous voulions vérifier si S.A.G.A.C.E. , bien que développée pour jouer contre des joueurs humains était capable de rivaliser avec d'autres méthodes.

Nous avons réalisé quatre nouveaux programmes pour le jeu « Matching Pennies » :

Le premier joueur (« Mixte ») joue suivant une répartition probabiliste non uniforme ;

Le deuxième joueur (« Cyclique ») joue de façon cyclique une séquence de coups programmés. Ces coups peuvent être 'P', 'F' ou 'R' (pour Random uniforme) ;

Le troisième joueur (« Automate ») est capable de jouer suivant un automate à états finis ;

Le quatrième joueur (« Fictitious player ») joue selon la méthode du « joueur fictif » [Brown, 1951], [Robinson, 1951], [Owen, 1995]. Rappelons que cette méthode permet de découvrir, de façon itérative, la stratégie optimale au sens de Nash.

Les trois premiers joueurs que nous avons utilisés pour illustrer nos résultats sont les suivants :

« Mixte » joue 30% de 'P' et 70% de 'F'.

« Cyclique » joue suivant la séquence 'P P F R P P P R F F P R F F F'

« Automate » est représenté sur la figure suivante (figure 8.16).

Remarque : Une transition 'PF' signifie « Si j'ai joué 'P' et lui 'F' », une transition '#P' signifie « s'il a joué 'P' », etc..

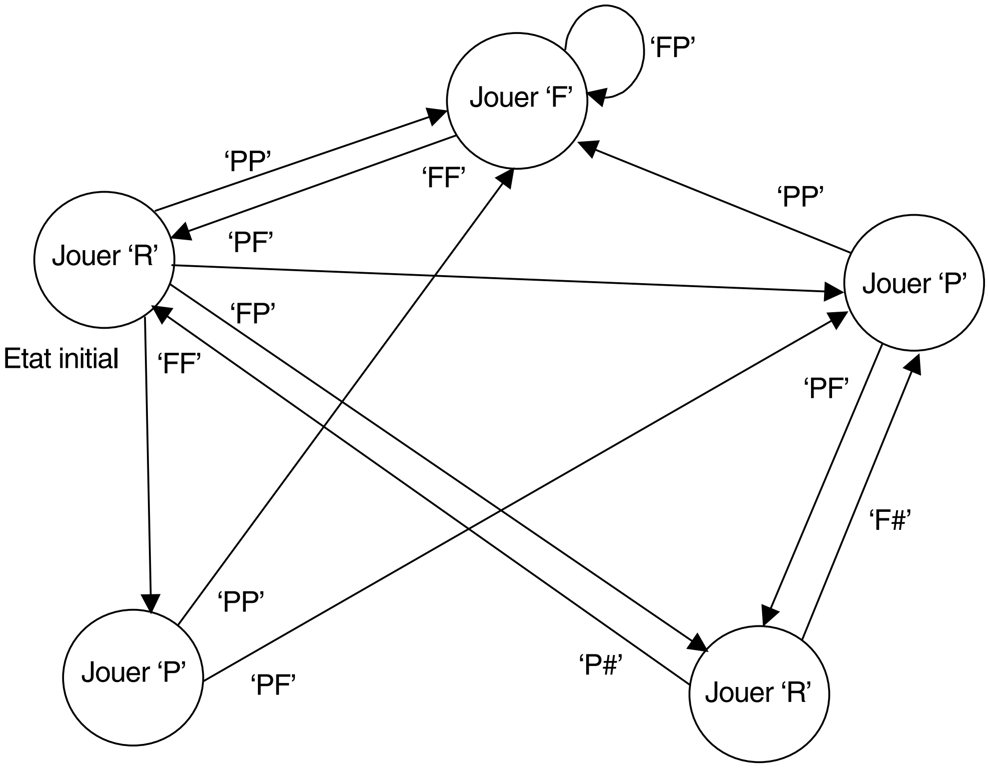


Figure 8.16. « *Automate* », Automate pour le jeu « Matching pennies ».

Les résultats moyens des séries de parties sont reportés dans le tableau suivant :

	S.A.G.A.C.E.	« Minasi »	« Shannon »	« Mixte »	« Cyclique »	« Automate »	« Fictitious »
S.A.G.A.C.E.	-	54% - 46%	54% - 46%	60% - 40%	70% - 30%	75% - 25%	71% - 29%
« Minasi »	46% - 54%	-	51% - 49%	56% - 44%	68% - 32%	71% - 29%	71% - 29%
« Shannon »	46% - 54%	49% - 51%	-	54% - 46%	50% - 50%	67% - 33%	64% - 36%

* Les pourcentages de gauche représentent les scores des joueurs 'ligne' (S.A.G.A.C.E., « Minasi » et « Shannon »).

Dans ces conditions, S.A.G.A.C.E. est le meilleur joueur. « Shannon » qui est meilleur que « Minasi » contre les joueurs humains est moins performant que ce dernier contre les trois joueurs « Mixte », « Cyclique » et « Automate ». Contre S.A.G.A.C.E., « Minasi » et « Shannon » ont des résultats équivalents.

Contre certains automates, S.A.G.A.C.E. se révèle être moins performant que « Minasi ». En majeure partie, ce sont des automates avec de très nombreux états qui ont des cycles très longs. Rappelons que S.A.G.A.C.E. accorde moins d'importance aux événements anciens qu'aux événements récents. De ce fait, il oublie délibérément des informations qui peuvent, dans certains cas, être pertinentes.

Contre ces mêmes automates, « Shannon » s'avère également peu performant, cela étant du à la faible quantité d'information qu'il conserve (ses huit mémoires).

Comme cela est tout à fait prévisible, S.A.G.A.C.E. ne se comporte pas mieux que d'autres méthodes contre certains adversaires qu'il est théoriquement impossible d'appréhender de façon optimale. Par exemple, contre l'automate probabiliste de Fortnow et Whang [Fortnow, 1994], S.A.G.A.C.E. ne se révèle, en moyenne, pas meilleure que Minasi ou Shannon.

L'automate de Fortnow et Whang se présente sous la forme suivante :

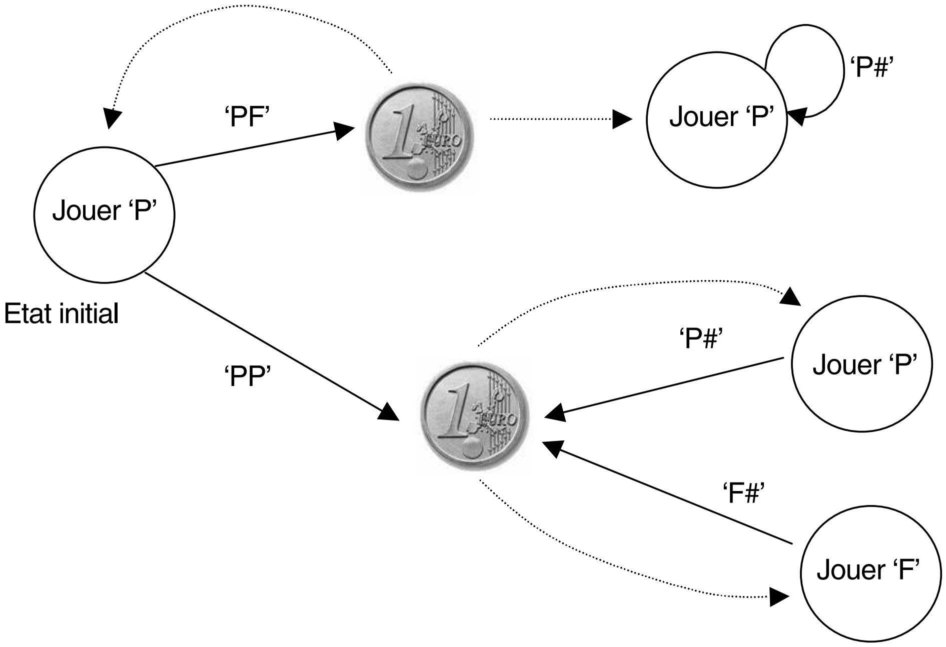


Figure 8.17. Automate de Fortnow et Whang pour Pair/Impair

Les transitions en pointillés signifient que l'automate effectue un choix aléatoire uniforme entre les différentes transitions. La partie « basse » de l'automate serait équivalente à un état noté 'R' qui bouclerait sur lui-même, mais nous avons cherché à conserver le formalisme original.

L'automate de Fortnow et Whang, à l'opposé de celui de Shannon, ne cherche pas à deviner le comportement de son adversaire mais tente de ne pas être facilement prévisible par lui (il a un rôle inversé).

Malgré sa simplicité apparente, cet automate pose un problème théorique : comment optimiser une stratégie à lui opposer. Pour tout k, il existe une stratégie régulière pour un joueur qui lui offre une espérance de gain de $1\text{-}2^{-k}$: il lui suffit de jouer 'F' k fois puis systématiquement 'P'. Pour autant, une espérance de gain de 1 est impossible : une stratégie consistant à jouer systématiquement 'P' mène à un gain de –1 et, s'il joue 'F', il y a toujours une petite probabilité que l'adversaire soit dans la partie « basse » de l'automate (ce qui donne une espérance de gain de 0).

S'il n'est théoriquement pas possible de définir la stratégie optimale contre cet adversaire, une stratégie 'F'(k)'P'(∞) (jouer 'F' k fois puis systématiquement 'P') semble très satisfaisante. Minasi, Shannon ou S.A.G.A.C.E. ne sont pas capables d'identifier explicitement une stratégie comme celle de cet automate parce qu'elles ne distinguent pas le premier coup des autres dans les jeux répétés (ce qui leur permettraient d'identifier le fait qu'il est inadapté contre cet automate de jouer 'P' au début parce que cela l'entraîne systématiquement dans un état d'équilibre de Nash - choix purement aléatoire -).

Seule S.A.G.A.C.E. (parmi les trois méthodes précédemment citées) est capable d'adopter une stratégie explicitement de la forme 'F'(k)'P'(∞)[1]. Mais rien ne garantit que S.A.G.A.C.E. converge vers une telle stratégie.

Il faut noter que l'automate de Fortnow et Whang est, quoi qu'il en soit, très peu performant. Minasi, Shannon et S.A.G.A.C.E., si elles sont incapables (par définition) de trouver une stratégie optimale contre cet automate, n'en sont pas moins efficaces (elles gagnent, en moyenne, bien plus de la moitié des affrontements).

La stratégie du joueur fictif se révèle inefficace dans la pratique. Bien qu'elle assure la convergence vers de la stratégie optimale au sens de Nash lorsqu'elle est confrontée à elle-même, elle est inadaptée contre des adversaires tels que « Minasi », « Shannon » ou S.A.G.A.C.E. parce qu'elle est totalement déterministe. Pour cette même raison, elle est inadaptée contre un joueur humain (un joueur humain attentif gagne plus de 65% de parties contre ce « joueur fictif »).

---

[1] Il faut qu'elle ait choisi de débrayer le module de modélisation, qu'elle n'utilise donc que le S.C. stratégique et qu'enfin elle possède dans sa base de stratégie les éléments d'une telle règle composite (par exemple : « jouer 'F'» ; « si j'ai joué 'F' précédemment, jouer 'F' » ; « si j'ai joué deux fois 'F', jouer 'P' » ; « si j'ai joué 'P' jouer 'P' »).

Nous avons imaginé une méthode moins déterministe (déjà présentée lors de la présentation des expérimentations pour le jeu Alésia) censée converger vers la solution mixte optimale contre un joueur particulier (pas nécessairement au sens de Nash). Il nous a semblé que, pour jouer contre un humain, la méthode du « joueur fictif » devait être aménagée. Ainsi, nous avons implémenté une version non déterministe de cette méthode que nous avons appelée « Mixed Fictitious Player ». Ce dernier effectue les mêmes enregistrements que le « joueur fictif » (cf. § 5.2.5.3.) mais joue suivant la répartition probabiliste des gains espérés (en fonction des enregistrements) plutôt que de jouer la stratégie pure donnant le meilleur résultat en fonction des enregistrement. Ce joueur se révèle (contre un joueur humain, « Minasi », « Shannon » ou S.A.G.A.C.E.) bien meilleur que le « joueur fictif » original.

Enfin, nous avons implémenté une version de ce joueur qui enregistre différemment les coups de son adversaire. Il différentie les parties gagnées des parties perdues par ses adversaires (suivant les considérations qui nous ont amené à définir les paramètre UT_0 et UT_1 de S.A.G.A.C.E.).

Ce joueur que nous avons choisi d'appeler « Sagacious Fictitious Player » se révèle encore meilleur que « Mixed Fictitious Player » contre un joueur humain, « Minasi », « Shannon » ou S.A.G.A.C.E. Son niveau est tout a fait comparable à celui de Mlinasi ou Shannon contre des joueurs humains et nous soupçonnons qu'il devrait se comporter bien mieux qu'eux pour des jeux plus complexes.

Remarque :

L'étude complète théorique et pratique de ces deux types de joueurs (« Mixed Fictitious Player » et « Sagacious Fictitious Player ») fait partie des perspectives de ce travail. Il est néanmoins immédiat que si on fait jouer le « Sagacious Fictitious Player » contre lui-même et s'il joue selon la stratégie optimale pure dictée par ses enregistrements alors il converge (comme le « fictitious player ») vers la solution optimale au sens de Nash. Par contre, une étude théorique doit être menée pour déterminer la convergence s'il joue en fonction d'une stratégie mixte définie à l'aide de ces mêmes enregistrements.

8.5 S.A.G.A.C.E. pour le jeu des trois pierres

Ce jeu se singularise des jeux précédents par le fait qu'il autorise le bluff.

A ce jeu, celui qui parle en premier (pour donner son estimation de la somme des deux mises) fournit une information exploitable à son adversaire. A contrario, celui qui parle en premier retire un choix potentiel à son adversaire. Les rôles des deux joueurs ne sont pas symétriques.

Le fait que la main (le joueur qui parle en premier) change à chaque fois qu'aucun des deux joueurs n'a gagné donne un intérêt important au bluff. Par exemple, le joueur jouant en premier peut cacher 3 pierres et annoncer qu'il pense que la somme des deux mises (ses 3 pierres et celles cachées par l'adversaire) est 2.

Il est particulièrement difficile de réaliser un programme basé sur la Théorie des jeux capable de bluffer. Le bluff est un aspect que l'on trouve plutôt dans les jeux à information incomplète (comme le poker).

L'utilisation de la Théorie requiert la donnée de la matrice des gains du jeu. Il est donc nécessaire de donner une estimation d'un gain « relatif » lié à un bluff réussi.

8.5.1 Théorie contre un adversaire humain

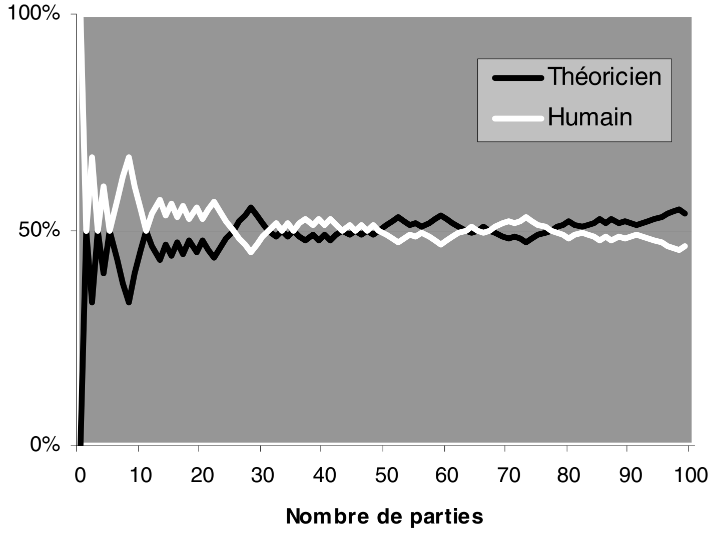


Figure 8.18. La Théorie contre des joueurs humains

Nous avons développé deux programmes « théoriciens ». L'un est doté de facultés de bluff, l'autre pas.

Contre un joueur humain, la forme des courbes de résultats de ces deux programmes est très semblable. Le bluffeur étant toutefois un peu meilleur (5 à 10% de mieux).

Les joueurs théoriciens gagnent souvent plus de 50% des parties contre un humain (le taux de 50% représente la valeur théorique, le jeu étant, dans sa globalité, équitable). Ce jeu demande un minimum de réflexion et un joueur humain se fait battre, en moyenne, s'il tente de jouer aléatoirement ou s'il ne réfléchit pas suffisamment.

8.5.2 Apprentissage par renforcement contre un adversaire humain

Nous avons développé deux programmes capables de faire évoluer leurs stratégies grâce à un algorithme d'apprentissage par renforcement.

Le premier programme utilise un algorithme de renforcement classique :

$$Fitness_{k_c}(C) = \frac{r_1 + r_2 + \ldots + r_{k_C}}{k_C}$$

Pour l'implémentation nous avons fixé la fitness initiale $fitness_0$ des classeurs à 0,25. Nous avons attribué les renforcements suivants :

Gain d'une pierre : 1 (la somme est correcte) ;

Perte d'une pierre : 0 (l'adversaire a gagné le coup) ;

Prise de la main : 0,1 (coup nul mais passage à l'état de premier joueur) ;

Passage de la main : 0,4 (coup nul mais passage à l'état de deuxième joueur).

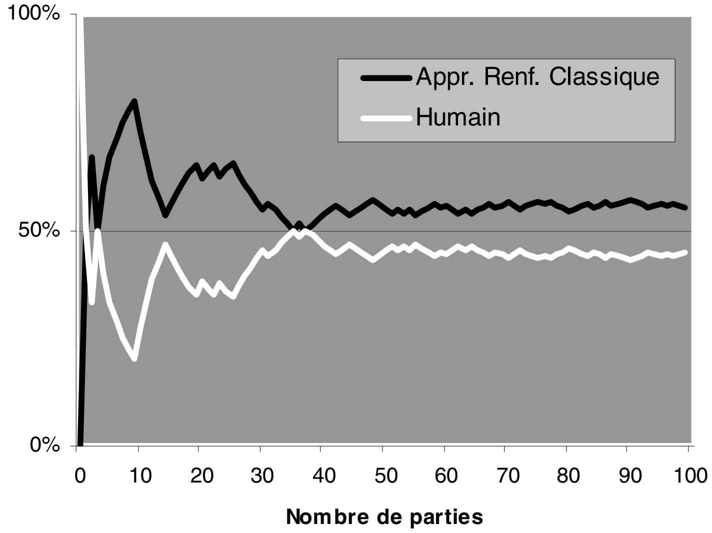


Figure 8.19. Apprentissage par renforcement « classique » contre des joueurs humains

Ce joueur se révèle meilleur que le joueur théoricien (figure 8.19).

Le principe même de ce type d'apprentissage fait que ce joueur utilise le bluff plus fréquemment contre les bons joueurs humains (ceux contre lesquels les stratégies calculatoires ne sont pas payantes). Les coups avec bluff sont souvent efficaces (et donc renforcés), sans doute parce qu'ils déstabilisent les joueurs humains.

En modifiant le renforcement attribué au bluff, on peut rendre un joueur particulièrement bluffeur. Les joueurs humains ont beaucoup plus de mal à jouer contre ce joueur que contre le joueur théoricien parce que ce dernier est sans surprise. S'il annonce une somme de 0 c'est qu'il a nécessairement misé 0 pierre. Ainsi, si son adversaire a joué autre chose que 0, il gagne en annonçant une mise égale à la somme.

L'autre joueur utilisant un algorithme d'apprentissage par renforcement utilise un renforcement exponentiel par pondération temporelle :

$$Fitness_{k+1} = (1-\alpha)^{k+1} Fitness_0 + \sum_{i=1}^{k+1} \alpha(1-\alpha)^{k-i+1} r_i$$

Les renforcements étant les mêmes que ceux du joueur précédent. Nous avons fixé le pas de renforcement α à 0,01.

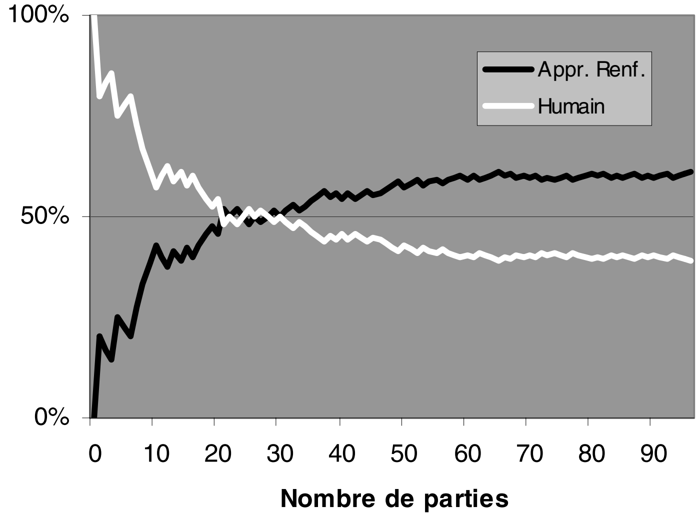


Figure 8.20. Apprentissage par renforcement temporel contre des joueurs humains

Ce joueur est plus performant que le précédent (figure 8.20). Le fait qu'il accorde plus d'importance aux coups récents qu'aux coups plus anciens semble l'avantager.

Cela corrobore notre intuition sur le développement des stratégies des joueurs humains qui semblent plus concernés par les coups récents.

La logique de ce joueur semble échapper complètement à la compréhension de ses adversaires humains. Certains joueurs se sont révélés incapables de le distinguer du joueur S.A.G.A.C.E.

8.5.3 S.A.G.A.C.E. contre un adversaire humain

S.A.G.A.C.E. utilise exactement les mêmes paramètres que le joueur précédent pour l'apprentissage des classeurs du S.C. stratégique. La différence avec le joueur précédent se situe principalement dans la présence du S.C. d'anticipation de S.A.G.A.C.E.

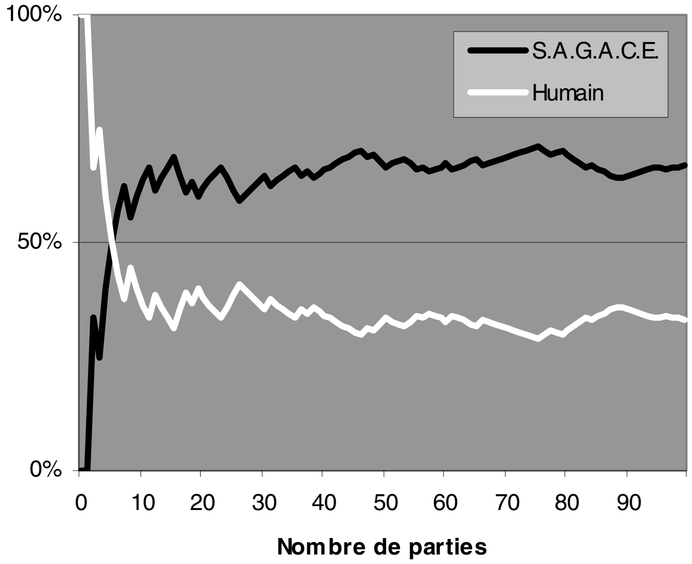


Figure 8.21. S.A.G.A.C.E. contre des joueurs humains

S.A.G.A.C.E. est un très bon joueur à ce jeu. Il surpasse tous les autres programmes que nous avons développés.

La courbe des résultats de S.A.G.A.C.E. contre les humains (figure 8.21) est très semblable à celle du programme basé sur un apprentissage par renforcement.

Deux points importants les différencient toutefois :

- Il faut en général une dizaine de parties seulement à S.A.G.A.C.E. pour dominer ses adversaires humains (contre 30 pour le joueur précédent) ;
- S.A.G.A.C.E. gagne, en moyenne, 66% des parties (contre 55% pour le joueur précédent).

Ces différences illustrent parfaitement l'intérêt de l'approche S.A.G.A.C.E. L'ajout du S.C. d'anticipation (dans S.A.G.A.C.E.) est la seule différence entre ces deux joueurs.

9 Conclusions et perspectives

9.1 Résumé

S.A.G.A.C.E. a été développée pour adresser des problèmes pour lesquels il est souhaitable, sinon nécessaire, d'anticiper les choix d'un intervenant humain.

Le domaine des jeux en général, et celui des jeux à information complète et imparfaite, en particulier, nous a paru être un champ d'expérimentations particulièrement adéquat. Nous avons ainsi cherché à développer plusieurs programmes utilisant S.A.G.A.C.E. et permettant d'affronter des joueurs humains.

Classiquement, les programmes censés être capables d'affronter des joueurs humains dans des jeux de réflexion sont basés sur la Théorie des jeux. Si la théorie permet, quand elle est applicable, de fournir des solutions optimales pour les jeux à information complète et parfaite, elle est inadaptée aux jeux à information complète et imparfaite contre des humains parce qu'elle suppose à tort une parfaite rationalité des joueurs. De ce fait, elle aboutit souvent à des stratégies inefficaces.

Les meilleurs systèmes capables de jouer à des jeux à information complète et imparfaite ont recourt à une modélisation de leurs adversaires. Cette modélisation se base sur une série d'observations du comportement de l'adversaire et donne souvent de bons résultats. Cependant, très peu de systèmes ont été développés pour affronter explicitement des joueurs humains et n'intègrent donc que très rarement des considérations sur les stratégies humaines.

S.A.G.A.C.E. s'avère très efficace contre les joueurs humains parce qu'elle prend explicitement en compte les facteurs qui caractérisent les stratégies humaines : l'irrationalité, la faiblesse de calculs, la mémoire limitée etc.

9.2 Qualités de S.A.G.A.C.E.

S.A.G.A.C.E. est particulièrement adaptative, ce qui semble indispensable lorsqu'on s'attache à la modélisation de comportements eux-mêmes hautement adaptatifs (les comportements humains).

S.A.G.A.C.E. s'adapte en identifiant chacun de ses adversaires et en ajustant sa stratégie en fonction de chacun d'eux. Elle intègre des processus d'apprentissage de nouveaux comportements (nouvelles règles) par observation, par imitation, par recombinaison de comportements déjà assimilés, par « regrets ».

S.A.G.A.C.E. possède des facultés de modélisation de ses différents adversaires. Ces facultés lui permettent de prédire le comportement de ces adversaires et d'anticiper leurs changements de stratégies. De plus, elle identifie chacun de ses adversaires individuellement, ce qui offre des possibilités de généralisation et lui offre la possibilité de prédire le comportement d'un joueur inconnu mais qui exhibe un comportement proche d'un adversaire identifié.

S.A.G.A.C.E. est également réactive dans la mesure où elle est susceptible de « débrayer » le module d'anticipation si ses performances sont insuffisantes. Cela est rendu possible par l'aspect modulaire de la méthode : l'anticipation est une composante pouvant être prise en compte ou non par le module de stratégie générale.

Enfin, S.A.G.A.C.E. est capable, dans des circonstances particulières, d'anticiper un coup jamais observé de l'adversaire (S.A.G.A.C.E. peut créer des règles par la méthode des regrets dans la base de modélisation d'un adversaire et ces règles peuvent correspondre à un coup que l'adversaire aurait dû jouer et qu'il est donc susceptible de jouer ultérieurement dans la même situation). Cela est remarquable dans la mesure où tous les systèmes connus basent leur modélisation uniquement sur des comportements observés[1].

9.3 Limitations de S.A.G.A.C.E

9.3.1 les Métaconnaissances

Un certain nombre de paramètres de la méthode S.A.G.A.C.E. sont réglés par des métaconnaissances :

Paramètres de l'algorithme génétique (critères de sélection, taux de mutation, etc.) ;

Paramètres de l'apprentissage ('pas' du renforcement) ;

Paramètres de l'anticipation (critère de débrayage de la modélisation pour l'anticipation, taille de la mémoire, coefficient de UT_0 et UT_1, etc.) ;

Paramètres de la généralisation ;

Paramètres d'expertise du jeu (méta-règles du jeu, coups incohérents, coups stupides, etc.) ;

Paramètres des méthodes d'élection (choix de la règle à appliquer) ;

---

[1] Ce n'est toutefois pas le cas des systèmes utilisant des algorithme comme M* qui supposent que l'adversaire fonctionne selon un algorithme MinMax uniquement caractérisé par une fonction d'évaluation (de forme connue) et une profondeur de recherche qu'il s'agit de définir. Cependant, ces approches ne sont, à notre sens, pas réalistes en ce qui concerne des joueurs humains et n'ont d'ailleurs pas apporté de preuves (expérimentations) du contraire.

Dans sa version actuelle, la méthode exige un certain nombre de choix heuristiques quant au contenu des bases de méta-connaissances. Par ailleurs, certains de ces choix sont, a priori, très liés aux applications de la méthode (c'est particulièrement le cas des paramètres d'expertise).

9.3.2 le bluff

Parmi les considérations concernant les stratégies humaines, il en est une que nous n'avons pas exploitée totalement (nous l'avons seulement exploitée pour l'implémentation du jeu des trois pierres) : la faculté de bluffer. Cette forme d'intelligence machiavélique est pourtant une composante essentielle des interactions entre les êtres vivants et les humains en particulier.

Dans la mesure où S.A.G.A.C.E. construit un modèle du comportement de ses adversaires, on peut supposer que la dimension du bluff serait implicitement prise en compte par ledit modèle. Cependant, cette approche ne nous semble pas satisfaisante, comme ne nous avait pas paru satisfaisante l'approche consistant à considérer qu'une méthode quelconque d'apprentissage (pour les jeux) intègre implicitement un modèle de l'adversaire puisqu'elle s'adapte en fonction de lui.

La différence entre un jeu avec une composante de bluff et un jeu n'en contenant pas nous semble aussi importante que celle qui distingue un jeu à information parfaite d'un jeu à information imparfaite.

Nous comptons poursuivre nos travaux dans ce sens et l'intégration de faculté de bluff ainsi que la capacité d'identifier le bluff des adversaires constitue donc notre première perspective.

9.4 1ère Perpective. Ajout d'une nouvelle dimension: le bluff

S.A.G.A.C.E. construit un modèle explicite de ses adversaires. Il nous semble que pour être efficace, un programme jouant contre des humains à un jeu intégrant une part de bluff doit construire un modèle dédié pour chacun de ses adversaires et doit apprendre à bluffer par expérience.

Nous avons choisi de développer une extension à S.A.G.A.C.E. capable d'appréhender de tels jeux.

Un modèle spécifique sera développé pour l'estimation du degré de bluff d'un joueur particulier dans chaque situation. Ce modèle prendra en compte les situations précédentes dans lesquelles l'adversaire aura bluffé, à quel point, et si cela a été bénéfique ou pas pour lui.

Il est parfois impossible de détecter un bluff. Par exemple au Poker, si un joueur est laissé seul dans une partie, il remporte le pot sans montrer son jeu. Rien ne permet alors de savoir s'il a bluffé (et gagne avec un jeu faible) ou s'il possédait un jeu fort. Pour cette raison, dans un premier temps, S.A.G.A.C.E. prendra uniquement en compte les bluffs avérés des adversaires[1].

Dans un second temps, nous tenterons de donner à S.A.G.A.C.E. des moyens d' « acheter de l'information » en « payant pour voir » même avec un jeu faible. Cette pratique est parfois indispensable pour constituer un modèle cohérent du bluff de ses adversaires.

Enfin, S.A.G.A.C.E. intégrera des méthodes pour apprendre à bluffer elle-même :

- Une méthode par imitation ;
- Une méthode par « regrets » ;
- Une méthode par recombinaison génétique de critères.

9.4.1 Bluff par imitation

Tout comme S.A.G.A.C.E. est capable d'apprendre des règles par imitation, elle pourrait apprendre, sous forme de règles, des stratégies de bluff.

En particulier, S.A.G.A.C.E. pourrait, sous réserve qu'un bluff avéré de l'adversaire s'est révélé efficace, générer des règles de la forme :

SI situation=situation-observée-de-l'adversaire ALORS bluffer-tel-qu'observé

La partie « *bluffer-tel-qu'observé* » pouvant sans doute être avantageusement remplacée par une stratégie générale et indépendante de bluff[2]. Certaines règles pourraient suggérer à S.A.G.A.C.E. de bluffer et d'autres pourraient lui permettre de déterminer comment.

De plus, S.A.G.A.C.E. pourrait, après avoir observé un bluff avéré, déterminer les critères de choix de l'adversaire qui l'ont conduit à tenter ce bluff (quel jeu il avait, quel était le comportement des autres joueurs, etc.).

---

[1] Ces bluffs se soldent le plus souvent par des échecs parce que, s'ils ont été avérés, cela signifie qu'un joueur a « payé pour voir ». La plupart du temps ceci se produit parce que le joueur pensait à un bluff et avait lui-même un bon jeu. Il arrive cependant qu'un joueur « paie pour voir », simplement pour identifier un éventuel bluff de l'adversaire (on appelle ce comportement « achat d'information »).

[2] Plutôt que de reproduire exactement un bluff observé, il semble en effet plus judicieux de distinguer la volonté de bluffer de la façon de bluffer.

9.4.2 Bluff par « regrets »

La méthode originale des regrets intégrée à S.A.G.A.C.E. pourrait être ajustée pour adresser le problème du bluff. Il ne suffit pas de reprendre la méthode telle quelle. En effet, rappelons-le, la méthode des regrets consiste à déterminer par calcul le meilleur coup qu'il aurait fallu jouer connaissant, en fin de manche, le coup effectué par l'adversaire.

Il n'est bien sûr pas possible de connaître le comportement qu'aurait eu l'adversaire devant un bluff, mais la révélation éventuelle de son jeu en fin de manche peut permettre de déterminer s'il aurait pu être profitable de bluffer ou pas. Effectivement, le modèle de l'adversaire doit contenir des indications précisant avec quel jeu et dans quelles circonstances un adversaire est susceptible de bluffer ou de subir un bluff. Enfin, la façon de jouer d'un joueur révèle parfois, en partie, son jeu.

Ainsi, connaissant l'attitude d'un joueur devant un bluff en fonction de son jeu propre et modélisant son jeu, une méthode adaptée des regrets permettrait de définir des situations dans lesquelles il est judicieux de bluffer contre ce joueur.

9.4.3 Bluff par recombinaison de critères

Un bluff s'effectue sur la base d'un certain nombre de critères :

- Le jeu du joueur ;
- Les jeux présumés des adversaires ;
- Leur attitude durant la manche ;
- Etc.

Sous réserve de déterminer une forme satisfaisante pour les règles de bluff, il sera possible de découvrir de nouvelles règles par recombinaison génétique de règles existantes (pouvant être créées par introspection, par imitation ou par la méthode des regrets).

9.4.4 Choix d'un jeu approprié

A l'instar du groupe de recherche « GAMES » de l'Université d'Alberta dirigé par Schaeffer, nous avons choisi d'appliquer les techniques originales correspondantes au jeu du Hold'em Poker. Ce jeu est sans doute l'un des jeux les plus intéressants de notre point de vue parce qu'il intégre une forte proportion de réflexion, de bluff, d'incertitude tout en minimisant la part de hasard.

Un sérieux travail statistique étant nécessaire avant d'aborder les autres aspects de ce jeu, nous avons d'abord de centrer notre étude sur un jeu à la combinatoire moins importante : une variante originale du Poker de dés qui utilise un système de mises inspiré du Hold'em Poker.

9.5 2ème Perpective : généralisation de l'approche

Nous avons, par ailleurs, l'ambition de généraliser une partie de la méthode S.A.G.A.C.E., lui adjoignant des procédures d'auto-paramétrage (notamment pour les bases de métaconnaissances).

Comme cela a été décrit précédemment, l'ajustement de l'ensemble des paramètres de l'algorithme génétique utilisé pour la création de nouvelles règles est effectué par un second algorithme génétique. Dans la version actuelle, ce second A.G. est particulièrement simplifié et nous envisageons de le compléter.

Plus généralement, nous souhaitons gérer tous les paramètres de S.A.G.A.C.E. (actuellement sous forme de méta-règles) par un algorithme génétique susceptible de les ajuster automatiquement en fonction des performances globales de la méthode.

Il s'agit d'un problème épineux d'optimisation multi-critères. Cependant, les différents paramètres ne sont pas tous interdépendants. La qualité de l'anticipation, par exemple, est indépendante de l'usage qu'on en fait et il est donc possible d'optimiser indépendamment l'un et l'autre.

Enfin, nous réfléchissons à une façon d'intégrer explicitement une hiérarchie dans le système de classeurs stratégique (voire également dans celui d'anticipation) [Donnart, 1998].

9.6 3ème Perpective : application à d'autres domaines

9.6.1 Méthode des regrets

La plupart des méthodes ou techniques originales incorporées dans S.A.G.A.C.E. devraient trouver des applications dans de nombreux autres domaines que les jeux :

Par exemple, la méthode des regrets intégrée à S.A.G.A.C.E., sous réserve d'être ajustée, devrait s'appliquer à l'implémentation d'agents d'interface dont le but serait d'aider un utilisateur d'un système complexe (navigation sur le WEB, gestion du courrier électronique, etc.).

Cet ajustement de la méthode des regrets pourrait être fondé sur le feedback de l'utilisateur vers ses agents d'interface [Meyer, 1997c]. Sachant quelles actions des agents ont été inefficaces, et sachant ce que le l'utilisateur aurait souhaité, il serait possible de créer par regret les règles de comportement adéquates.

9.6.2 Amorçage

Pour initialiser des systèmes à base de règles, il est souvent utile de faire appel à un expert pour qu'il formalise ses propres connaissances sous une forme adéquate afin de les introduire dans les systèmes. Il est cependant souvent impossible de faire faire cette transcription à l'expert lui-même qui, bien qu'il sache indiquer les choix judicieux, ne sait souvent pas expliciter ses critères de décision.

La méthode de modélisation de S.A.G.A.C.E. par anticipation devrait permettre une telle forme d'amorçage en affranchissant l'expert de la nécessité d'expliciter ses critères, dans la mesure où ces derniers seraient découverts par la méthode [Meyer, 1996b].

9.6.3 Interfaces collaboratrices hommes-machines

Le travail en équipe n'est efficace que lorsque les réactions des partenaires sont prévisibles. Dans le cadre de règles de collaboration spécifiées, le travail s'effectue correctement. C'est ainsi que l'homme et la machine sont capables, dans des circonstances ordinaires, de coopérer parce que les machines sont programmées pour ne pas surprendre.

Dans les situations de crise, quand les circonstances échappent aux schémas établis, des hommes opérant de concert sont souvent capables de bien réagir parce qu'ils savent se mettre à la place les uns des autres et parviennent par introspection à anticiper réciproquement leurs actions ou attitudes. C'est d'ailleurs une composante majeure des entraînements des spécialistes dans de nombreux domaines.

Permettre, à terme, aux machines d'anticiper les comportements humains dans des circonstances non explicitement programmées c'est, sans doute, rendre plus robustes, moins hasardeuses et plus efficaces les collaborations hommes-machines.

A ce titre, l'approche S.A.G.A.C.E. trouvera naturellement de nombreuses applications dans des interfaces de collaboration hommes-machines.

ANNEXE A Captures d'écran

A.1. SunTzu

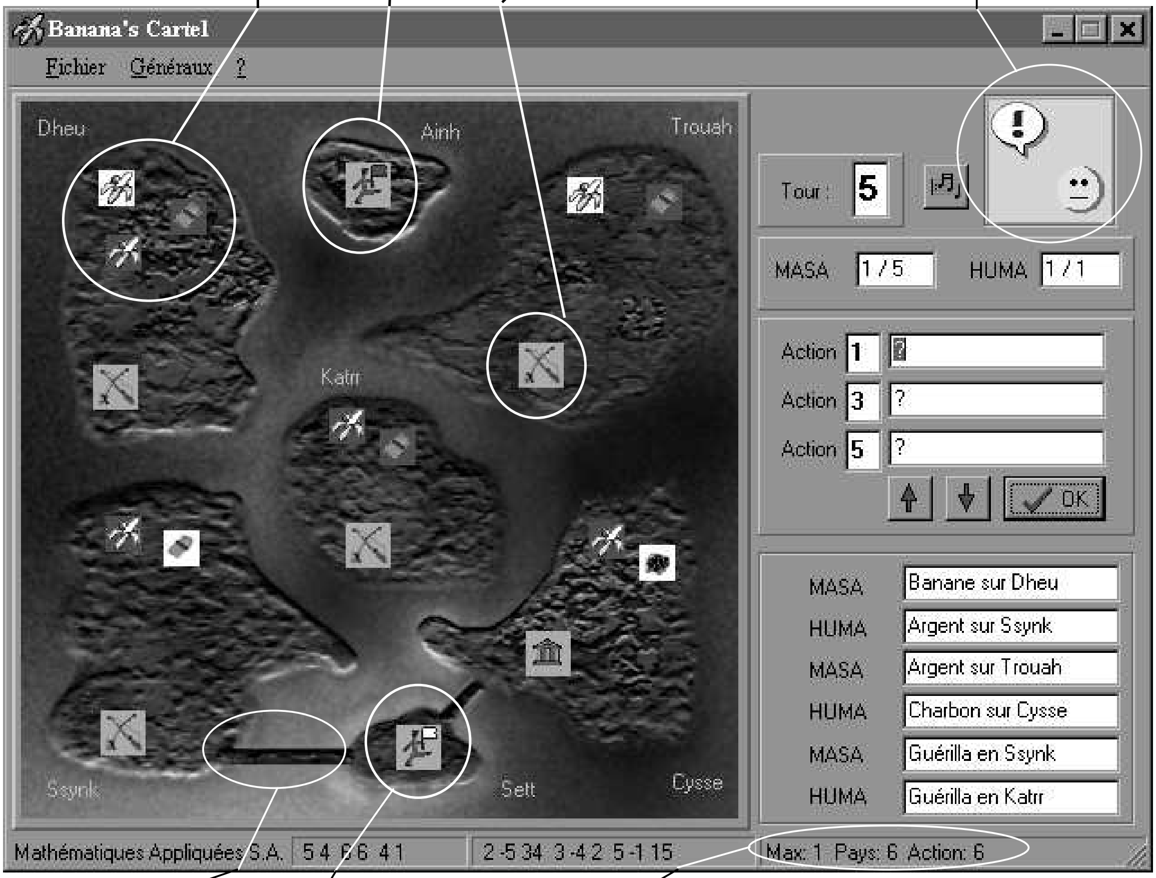
La reproduction suivante montre la version la plus aboutie[1] du programme SunTzu. Le « blé » a été remplacé par des « Bananes ».

Ressources posées par les joueurs (Banane pour le joueur humain ; Banane et Argent pour le joueur informatique)

Pays du joueur informatique

Régime politique (ici : « Guérilla »)

Etat émotionnel du joueur informatique. (ici : Normal car les deux joueurs ont marqué un point. Le point d'exclamation signifie qu'il a découvert, après examen des résultats de ce tour, une nouvelle stratégie par la méthode des regrets –détaillée au bas de l'écran-)



Tour : 5

MASA 1 / 5 HUMA 1 / 1

Action 1 ?
Action 3 ?
Action 5 ?

MASA Banane sur Dheu
HUMA Argent sur Ssynk
MASA Argent sur Trouah
HUMA Charbon sur Cysse
MASA Guérilla en Ssynk
HUMA Guérilla en Katr

Mathématiques Appliquées S.A. | 5 4 6 6 4 1 | 2 -5 34 3 -4 2 5 -1 15 | Max: 1 Pays: 6 Action: 6

Pont entre les pays 6 et 7

Pays du joueur humain

La dernière stratégie découverte par le système. (ici : la stratégie est « Poser du charbon sur le pays 6 ». Cette stratégie est judicieuse parce que si la joueur informatique l'avait utilisée en action N°1, le joueur humain n'aurait pas pu poser de charbon ; gain relatif +2).

---

[1] Le programme qui est reproduit ici est une version développée au sein de la société Mathématiques Appliquées S.A..

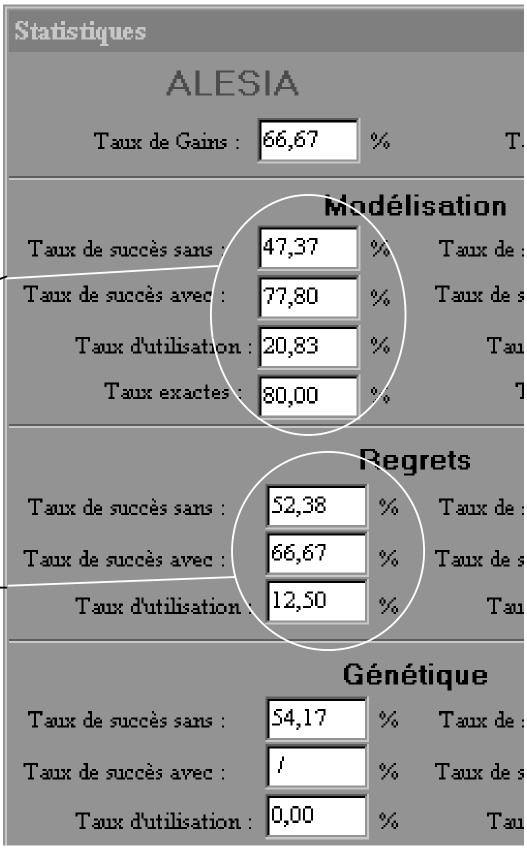
A.2. Alésia

Position du jeton

Camp du joueur humain

Dernière mise du joueur humain

Capital restant du joueur humain



Taux de Gains : 66,67 %

Modélisation

Taux de succès sans : 47,37 %

Taux de succès avec : 77,80 %

Taux d'utilisation : 20,83 %

Taux exactes : 80,00 %

Regrets

Taux de succès sans : 52,38 %

Taux de succès avec : 66,67 %

Taux d'utilisation : 12,50 %

Génétique

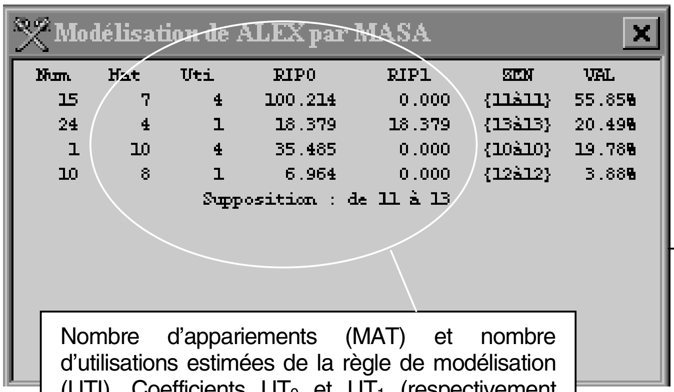
Taux de succès sans : 54,17 %

Taux de succès avec : / %

Taux d'utilisation : 0,00 %

Statistiques concernant la modélisation (le S.C. d'anticipation).
(ici :
- Le joueur informatique gagne 78% des coups quand il utilise la modélisation et seulement 47% sinon.
- La modélisation est utilisée pour 21% des coups
- Les prédictions sont exactes à 80%).

Statistiques concernant l'utilisation de la méthode des regrets.
(ici :
- Le joueur informatique gagne 67% des coups quand il utilise cette méthode et seulement 52% sinon.
- La méthode est utilisée pour 13% des coups)



```
 Num   Mat   Uti    RIP0      RIP1      SEN      VAL
  15     7     4  100.214     0.000  {11à11}  55.85%
  24     4     1   18.379    18.379  {13à13}  20.49%
   1    10     4   35.485     0.000  {10à10}  19.78%
  10     8     1    6.964     0.000  {12à12}   3.88%
              Supposition : de 11 à 13
```

Nombre d'appariements (MAT) et nombre d'utilisations estimées de la règle de modélisation (UTI). Coefficients UT_0 et UT_1 (respectivement notés RIP0 et RIP1).

Fenêtre d'information sur la modélisation.
(ici :
S.A.G.A.C.E. a identifié 4 comportements possibles de son adversaire humain « il va jouer 11, 13, 10 ou 12 ». La valeur de S.U.C.R.E. -VAL- calculée à partir des coefficients UT_0 et UT_1 des choix correspondants font abandonner l'idée qu'il va jouer 10 ou 12. → conclusion, il va jouer 11 ou 13 ce qui est traduit en « de 11 à 13 ». Remarque : la valeur 12 se trouve être intégrée dans cet intervalle).

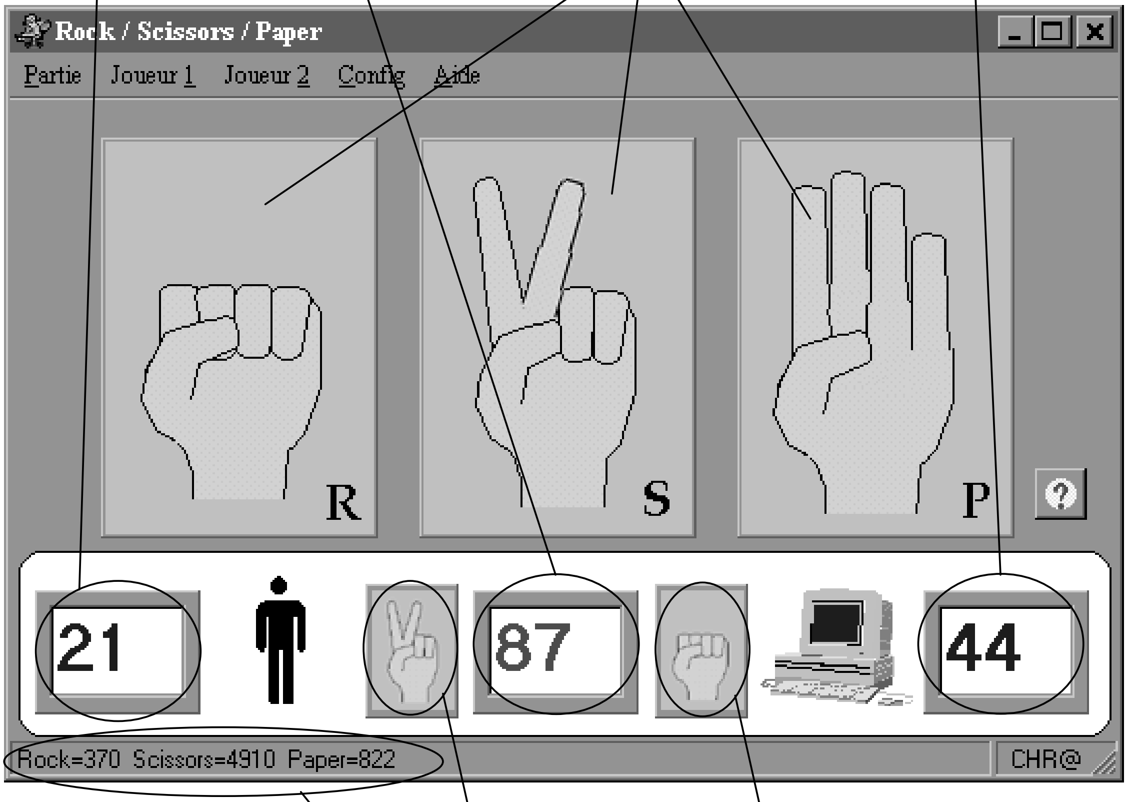
A.3. « Pierre / Ciseaux / Papier »

Score du joueur humain.

Nombre de parties jouées

Choix du joueur humain.
Il clique avec la souris sur ces images ou appuie sur les touches indiquées.

Score du joueur informatique.
(ici : S.A.G.A.C.E.)



```
Rock=370 Scissors=4910 Paper=822
```

Les trois valeurs de S.U.C.R.E. pour la prédiction de S.A.G.A.C.E. (cet affichage est optionnel…) (ici : les valeurs indiquent que le coup le plus probable de l'adversaire humain était « Scissors » et c'est pour cela que le programme a joué « Rock »).

Choix du joueur humain à ce tour. (ici : il a joué « Scissors » comme le pensait S.A.G.A.C.E.).

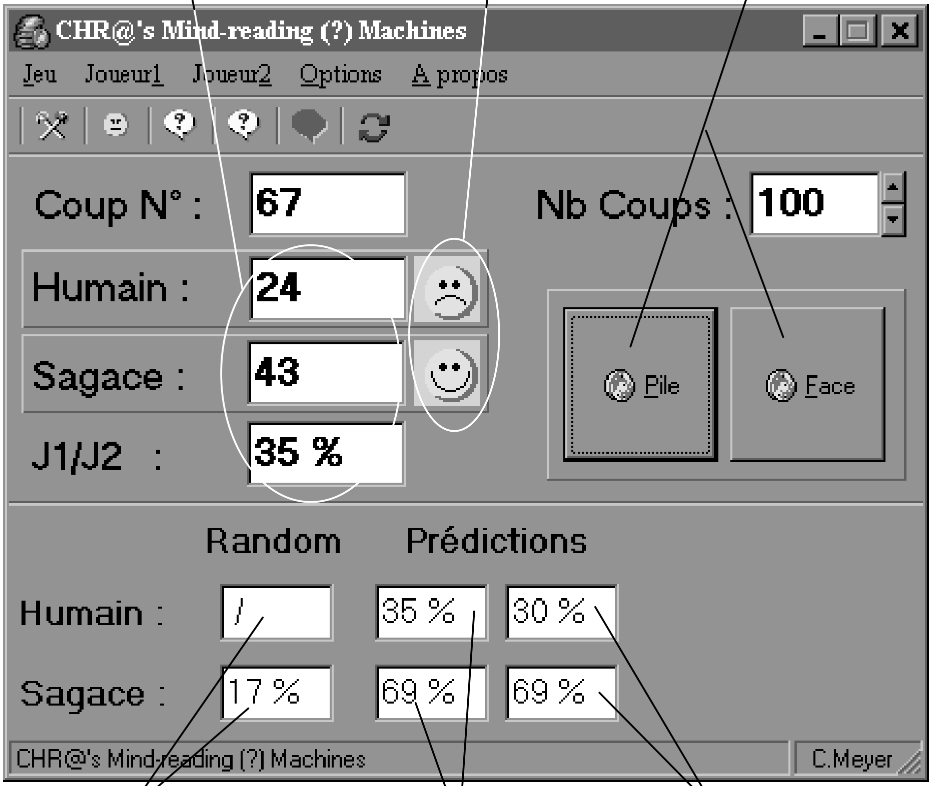
Choix du joueur informatique à ce tour. (ici : il a joué « Rock » parce qu'il pensait - à raison - que son adversaire allait jouer « Scissors »).

A.4. « Pair / Impair » (ou « Matching pennies »).

Scores des deux joueurs et pourcentage relatif

Emotions des deux joueurs.
(estimée pour le joueur humain).

Commandes du joueur humain.
Il peut cliquer sur les images ou appuyer sur les touches ‘P’ et ‘F’.



Coup N° : 67

Nb Coups : 100

Humain : 24

Sagace : 43

J1/J2 : 35 %

	Random	Prédictions	
Humain :	/	35 %	30 %
Sagace :	17 %	69 %	69 %

Taux d’utilisation du hasard.
(ici : 17% de recours au hasard pour S.A.G.A.C.E. Le recours au hasard du joueur humain est toujours inconnu et considéré comme nul).

Taux de prédictions exactes des deux joueurs.
(ici : S.A.G.A.C.E. a fait, en moyenne, 69% de prédictions exactes et le joueur humain 35%).

Taux de prédictions exactes des deux joueurs sachant que l’autre n’a pas joué au hasard.
(ici : le joueur humain a fait 30% de prédictions exactes alors que S.A.G.A.C.E. ne jouait pas aléatoirement.
Pour S.A.G.A.C.E. le taux est le même que le précédent parce qu’on considère que l’humain ne joue jamais au hasard).).

A.5. « Le jeu des trois pierres ».

L’implémentation particulière[1] de ce jeu qui est présentée ici s’appelle « le jeu des trois pièces » (au lieu de : « le jeu des trois pierres »).

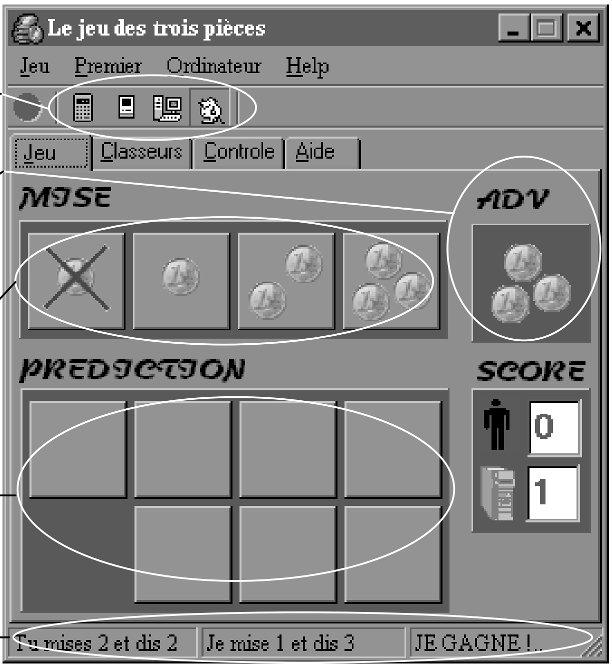
Définition de l’adversaire informatique :
Calculatrice = joueur mathématique faible ;
Macintosh = joueur mathématique moyen ;
Comp. PC = joueur mathématique optimal ;
Le rat Psycharpax = joueur S.A.G.A.C.E.

Nombre de ~~pierres~~… pièces restantes de l’adversaire.

Nombre de pièces restantes et commandes de choix de la mise du joueur humain.

Commandes de choix de la prédiction du joueur humain. S’il joue en second, la case correspondant à la prédiction du joueur informatique est grisée.

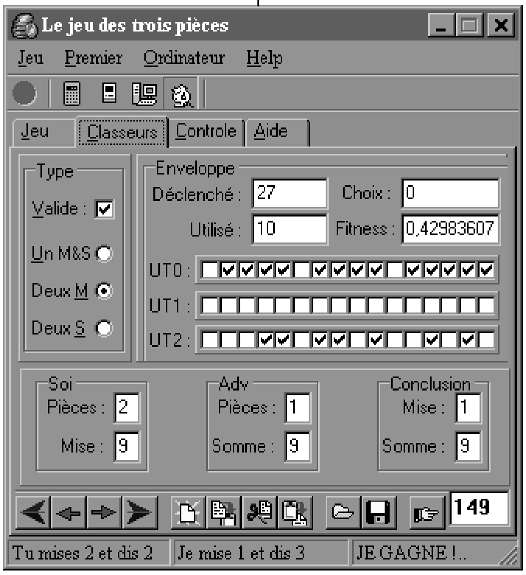
Commentaires du joueur informatique.



0
1

Tu mises 2 et dis 2 | Je mise 1 et dis 3 | JE GAGNE !..

Outil de visualisation et d’édition des classeurs du S.C. stratégique.



Type
Valide : ☑
Un M&S ○
Deux M ●
Deux S ○

Enveloppe
Déclenché : 27
Choix : 0
Utilisé : 10
Fitness : 0,42983607
UT0 : ☐☑☑☑☑☐☑☑☑☑☐☑☑☑☑☑
UT1 : ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐
UT2 : ☐☐☐☑☑☐☑☑☐☑☐☐☑☐☑☐

Soi
Pièces : 2
Mise : 9

Adv
Pièces : 1
Somme : 9

Conclusion
Mise : 1
Somme : 9

149

Tu mises 2 et dis 2 | Je mise 1 et dis 3 | JE GAGNE !..

Paramétrage de S.A.G.A.C.E.
Coefficients de la fonction S.U.C.R.E. (Phi0 et Phi1). Coefficients pour l’apprentissage par renforcement pour le S.C. stratégique

Apprentissage
Gain pièce : 1
Perte pièce : 0
Donne main : 0,4
Prend main : 0,1
Alpha : 0,05

Anticipation
MAJ des UT : ☑
Anticipation : ☐

S.U.C.R.E.
Phi0 : 0,5
Phi1 : 0,5

---

[1] Le programme qui est reproduit ici est une version développée au sein de la société Mathématiques Appliquées S.A..

ANNEXE B SUNTZU - exemples -

B.1. Forme des règles

L'implémentation de "SUNTZU" sous forme de systèmes de classeurs a nécessité une étude précise pour déterminer la forme des règles susceptibles de permettre de bien jouer à ce jeu.

Nous avons choisi une méthode introspective : nous nous sommes demandé comment nous raisonnons nous-même en jouant à ce jeu. C'est en nous observant réfléchir que nous avons pu déterminer les paramètres principaux de nos prises de décisions.

Il se trouve que nos réflexions étaient avant tout pays-centrées ; c'est-à-dire que nous regardions rarement l'ensemble du jeu mais plutôt chaque pays individuellement. Bien sûr certaines décisions (comme le fait de relier deux pays) avaient des conséquences globales mais les décisions étaient prises localement.

B.1.1. La partie conclusion

La conclusion des règles informatiques est directement inspirée du genre de décision prise par un joueur humain. Il y a huit décisions atomiques possibles qui sont :

Mise en guérilla:	codée 1
Mise en république:	codée 2
Mise en dictature:	codée 3
Introduction d'argent:	codée 4
Introduction de blé:	codée 5
Introduction de charbon:	codée 6
Liaison indirecte du pays avec celui du joueur:	codée 7
Liaison directe entre deux pays:	codée1K
Coupure de liaison:	codée 8
Ne rien faire sur ce pays	codée 9

La liaison directe d'un pays (codée 7) est une décision locale mais son application pourra être traitée de façon générale pour faire le meilleur choix stratégique de liaison. Par exemple, si une règle concluant sur la liaison indirecte du pays 4 est appliquée, elle ne décrit pas comment faire cette liaison, le système a d'autres outils pour cela.

Une liaison directe concerne seulement deux pays : une voie de chemin de fer est établie entre eux.

Le fait de ne rien faire sur un pays (action 9) n'est pas à proprement parler une action : c'est une décision possible qui ne prend pas la place d'une des trois actions d'un joueur mais qui l'empêche de choisir une action concernant ce pays.

B.1.2. La partie condition

La partie condition prend en compte tous les critères que nous avons pu juger déterminants pour une prise de décision :

La position exacte du pays auquel on s'intéresse,

La distance de ce pays au sien,

(Ces deux paramètres peuvent être simplifiés en un seul en utilisant les numéros des pays).

Le régime politique actuel dans le pays concerne,

La dernière ressource posée sur ce pays,

Les quantités d'argent, de blé, de charbon apportées par chacun des joueurs sur ce pays.

Certaines combinaisons linéaires de quantités d'argent, blé et charbon (ex : somme du charbon et du blé de l'adversaire),

La liaison avec le pays du joueur ou de l'adversaire (et le type de cette liaisons),

Les instants auxquels la règle peut s'appliquer (en début de tour, en fin de tour, etc.),

Les tours auxquels peut s'appliquer la régle (dernier tour uniquement par exemple),

B.1.3. La partie anticipation

Cette partie ne contient qu'une information : l'intention que l'on prête à l'adversaire.

Il s'agit du coup que l'on pense que l'adversaire va jouer sur le pays considéré.

B.1.4. La forme retenue

Pays		Régime				Ressource			A1		B1		C1		A7		B7		C7		R1		R7		Position						Tours		P	Z
		D1	D7	G	R	A	B	C													μ	r	μ	r	1	2	3	4	5	6				
□	□	b	b	b	b	b	b	b	□	□	□	□	□	□	□	□	□	□	□	□	i	j	k	l	b	b	b	b	b	b	□	□	z	z

La prémisse d'une règle indique dans quelles situations elle est susceptible d'être appliquée.

Pays : indique les pays concernés par la règle *(ex : 25 pour les pays 2 à 5 inclus).*

Régime : indique sous quels régimes la règle s'applique *(Pour signifier un pays vide de tout régime, on invalide tous les cas).*

- DI : Sous une dictature instaurée par le joueur 1 (B est un booléen).

- D7 : Sous une dictature instaurée par le joueur 2 (Le pays noté 7).

- G : En guérilla.

- R : En république.

Ressource : indique pour quelle(s) ressource(s) dernièrement placée(s) la règle s'applique.

- A : la dernière ressource placée sur le pays est de l'argent.

- B : la dernière ressource placée sur le pays est du blé.

- C : la dernière ressource placée sur le pays est du charbon.

A1,B1,C1 : les quantités d'argent, de blé, de charbon du joueur I sur le pays (ex : *B1 =23 signifie "S 'il y a 2 ou 3 blés du joueur 1">.*

A7,B7,C7: idem pour le joueur 2 (l'adversaire).

R1 : indique de quelle façon le pays doit être relié au pays du joueur 1.

μ : liaison réelle (0: non relié; 1: un chemin; 2: deux chemins distincts ; 3: un ou deux chemins; 4: aucune irnportance).

r : liaison vue uniquement d'un point de vue "Chemin de fer" c'est-à-dire sans considérations politiques.

R7 : idem pour le pays du joueur 2.

Position : indique à quelle(s) position(s) peut être jouée la règle.

- 1 a 6 booléen sur chacune des six positions possibles

Tours : indique à quel(s) tour(s) peut être jouée la règle *(ex: 44 signifie « seulement au $4^{ème}$ tour).*

P(réd) : prédiction de ce que va jouer l'adversaire. Le codage est le même que celui de la conclusion des règles. (cf. coefficient 'ZEN') *(ex: 3 signifie « si on pense que l'adversaire va instaurer la dictature sur ce pays»).*

Z(EN) : conclusion de la règle.

- 1 : guérilla 2 : république 3: dictature
- 4 : argent 5 : blé 6 : charbon
- 7: essayer de relier au pays du joueur (chemin de fer ou politique)

(Dans ce cas, il y a recherche d'une solution de liaison en utilisant la base stratégique du joueur).

-11 : lier ce pays avec le pays 1.

-1K : lier ce pays avec le pays k.

Exemple d'une règle

Pays		Régime				Ressource			A1		B1		C1		A7		B7		C7		R1		R7		Position						Tours		P	Z
		D1	D7	G	R	A	B	C													μ	r	μ	r	1	2	3	4	5	6				
5	7	N	N	O	N	O	O	O	1	6	0	6	0	6	0	6	0	6	0	6	4	4	1	2	N	N	O	O	O	O	3	6	2	3

Cette règle doit s'interpréter de la manière suivante :

- Pour les pays Nos 5, 6 et 7,
- Si le régime politique est une guérilla,
- S'il y a au moins un "argent" du joueur 1(6 est le maximum qu’il puisse y avoir)
- Quelle que soit la façon dont le pays est relié ou non au pays,
- Si le pays est relié géographiquement par deux chemins distincts au pays 7 mais réellement de façon unique *(un des deux chemins passe par un pays en dictature)*
- En position 3, 4, 5 ou 6,
- Au 3ème, 4ème, 5ème ou 6ème tour,
- Si on pense que l'adversaire va mettre le pays en république,

ALORS, le mettre en dictature !

L’enveloppe des règles contient d'autres coefficients qui sont utilisés par les bases de stratégie. Il s’agit de la fitness, du nombre d'utilisations, etc.

B.2. Exemple de la méthode de généralisation de S.A.G.A.C.E.

Pays		Régime				Ressource			A1		B1		C1		A7		B7		C7		R1		R7		Position						Tours		P	Z	Fitness
		D1	D7	G	R	A	B	C													μ	r	μ	r	1	2	3	4	5	6					
2	3	N	N	N	N	N	N	O	1	6	0	0	0	6	2	5	0	6	0	6	4	4	4	4	N	N	O	O	O	O	2	8	0	3	1.10
2	5																																		1.20
2	6																																		1.41
		O	N	N	N																														0.90
		N	O	N	N																														0.81
		N	N	N	O																														1.10
		N	N	O	O																														0.91
		N	N	N	O																														1.50
						O	O	O			0	1																							0.60
											0	0			1	5									N	O	O	O	O	O					1.70
															0	6									O	O	O	O	O	O					0.81
															1	6																			1.82
									0	6																									4.11
2	6	N	N	N	O	O	O	O	0	6	0	0	0	6	1	6	0	6	0	6	4	4	4	4	O	O	O	O	O	O	2	8	0	3	

Ce tableau montre les différentes étapes de la généralisation d'une règle. Ne sont indiquées, pour chaque étape, que les données qui changent. Au départ, la règle a une fitness moyenne (indiquée à droite du tableau) de 1,10. Cette fitness augmente avec les différents essais pendant trois étapes puis elle devient inférieure à la fitness de la règle initiale. La généralisation va alors porter sur un autre coefficient (un autre paramètre) de la règle en repartant de l'avant dernière règle créée. On peut suivre facilement les modifications subies par cette règle.

Toutes les étapes suivantes ont été retirées de ce tableau parce que les tentatives correspondantes n'ont pas amélioré la dernière règle affichée.

Le résultat final est tout de même impressionnant. A ce jeu, une fitness de 4,11 est en effet relativement rare.

Malheureusement, la généralisation n'est pas toujours aussi efficace et les nombreux calculs qu'elle exige peuvent avoir été réalisés en vain.

ANNEXE C ALESIA - aspects théoriques -

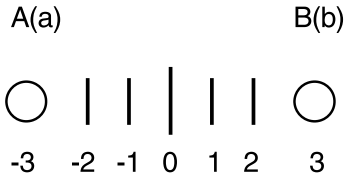
C.1. Stratégies pures à ALESIA

Nous allons démontrer que :

1) Il n'existe pas de stratégie pure garantissant la victoire.
2) Il n'existe pas de stratégie pure garantissant le nul.

La notion de garantie signifie qu'une telle stratégie devrait être applicable quelle que soit la stratégie adverse et, évidemment, sans la connaître au préalable.

C.1.1. Notations



Notons :

A et B les deux joueurs et a et b leur capital de points respectif,

p la position du jeton avec $p \in [-3,3]$

(-3 signifiant une victoire pour B et 3 une victoire pour A),

(a,b,p) la configuration du jeu,

G(a,b,p) la proposition « il existe une stratégie gagnante pour B dans (a,b,p) »,

N(a,b,p) la proposition « il existe une stratégie non perdante pour B dans (a,b,p) »,

a' et b' les capitaux respectifs de A et B après le coup envisagé,

Δ la mise de B au coup envisagé ($\Delta = b - b'$).

Avec ces notations, nous démontrerons $\neg G(50,50,0)$ et $\neg N(50,50,0)$

(Rappelons que les règles du jeu Alésia stipulent que les deux joueurs ont des capitaux initiaux égaux à 50, que le jeton est initialement situé au centre des 7 positions possibles et enfin que sont considérées comme nulles les parties pour lesquelles au final -quand les deux joueurs ont épuisé leur capital- $p \neq -3$ et $p \neq 3$).

C.1.2. démontrons que $\neg G(50,50,0)$

Quelle que puisse être la stratégie de B, il existe, parmi toutes les stratégies possibles de A une stratégie qui lui fasse jouer exactement le même nombre de points que B pour chaque coup. Avec cette stratégie (que nous appellerons miroir) le jeton ne bouge jamais et la partie est nulle.

Conclusion : $\neg G(50,50,0)$.

Remarque : la stratégie correspondante de A ne peut être explicitée. Il en existe une *particulière* pour chaque stratégie de B.

On ne peut considérer que la stratégie miroir réponde à N(a,b,p), dans la mesure où on ne peut l'expliciter avant que le jeu n'ait commencé. En effet, le premier coup lui-même est un pari sur le choix de l'adversaire. S'il est vrai que parmi, les 50 coups possibles, l'un est le bon, on ne peut dire lequel et la stratégie correspondante n'est donc pas identifiée.

C.1.3. démontrons que $\neg N(50,50,0)$

Lemme 1 :

$$\boxed{a \geq b - p + 3 \Rightarrow \neg N(a, b, p)}$$

démonstration :

$a \geq b - p + 3 \Leftrightarrow a - b \geq 3 - p$

La différence (a - b) représente l'avantage en points de A sur B. Si cette différence est supérieure à 3 - p, où que soit situé le jeton (quel que soit p), il existe, parmi toutes les stratégies possibles de A une stratégie consistant à jouer systématiquement un point de plus que B jusqu'à ce que p = 3 (victoire de A). Donc,

$a \geq b - p + 3 \Rightarrow \neg N(a, b, p)$.

Remarques :

Pour p = 2, il est immédiat que si $a \geq b + 1$, il suffit que A joue b + 1 points pour gagner. Cette stratégie, contrairement à la plupart des autres cas ($b \neq 2$, $a < (b+1)(3-p)$), est identifiée.

Pour $a \geq (b+1)(3-p)$, il suffit que A joue toujours b+1, il est sûr de gagner…

Il existe beaucoup d'autres cas de stratégie gagnante identifiée…

Lemme 2 :

$$\boxed{a \geq ½\,(b + 3) \;\Rightarrow\; \neg N(a, b, 2)}$$

démonstration :

Supposons que la situation soit (a, b, 2).

Si $b \geq a$ alors B <u>doit</u> jouer au moins a points pour ne pas perdre et A a tout intérêt à jouer 1 (on suppose $a \geq 1$)

La situation évolue donc ainsi :

$(a, b, 2) \rightarrow (a - 1, b - a, 1)$

$$\begin{aligned} \text{Or,}\quad a \geq ½\,(b + 3) &\Leftrightarrow 2a - 3 \geq b \\ &\Leftrightarrow a - 3 \geq b - a \\ &\Leftrightarrow a - 1 \geq b - a + 2 \\ &\Leftrightarrow (a - 1) - (b - a) \geq 2 \end{aligned}$$

d'après le lemme 1, $(a - 1) - (b - a) \geq 2 \;\Rightarrow\; \neg N(a - 1, b - a, 1)$

donc, $a \geq ½\,(b + 3) \;\Rightarrow\; \neg N(a, b, 2)$.

Lemme 3 :

$$\boxed{a \geq 1/3\,(2b + 8) \;\Rightarrow\; \neg N(a, b, 1)}$$

démonstration :

1^er^ cas : $a - 4 \geq b'$

Alors A ayant joué 1 ($a' = a - 1$), la situation évolue ainsi :

$(a, b, 1) \rightarrow (a - 1, b', 0)$ et d'après le lemme 1, $\neg N(a - 1, b', 0)$

2^ème^ cas : $a - 4 < b'$

dans ce cas, $\Delta = b - b' < b - a + 4$ (inversion de l'inégalité et ajout de b)

$a \geq 1/3\ (2b - 8) \Rightarrow 3a \geq 2b + 8$

$3a \geq 2b + 8 \Rightarrow a \geq 2b - 2a + 8 > 2\Delta$

donc, $a > \Delta$

Supposons alors que A joue $\Delta+1$ points.

Dans ce cas, la situation évolue de la façon suivante :

$(a, b, 1) \rightarrow (a - \Delta - 1, b - \Delta, 2)$

Or, $3a \geq 2b + 8 \Rightarrow 2a - b - 5 \geq b - a + 3$

Comme $\Delta < b - a + 4$, on vérifie que $b - a + 3 \geq \Delta$

Donc, $2a - b - 5 \geq \Delta$

Ce qui peut s'écrire : $2a - \Delta \geq b + 5$

D'où : $2a - 2\Delta - 2 \geq b - \Delta + 3$

Finalement : $a - \Delta - 1 \geq \frac{1}{2}(b - \Delta + 3)$

D'après le lemme 2, on a donc $\neg N(a - \Delta - 1, b - \Delta, 2)$

Conclusion, les deux cas impliquent $\neg N(a, b, 1)$ cqfd.

théorème :

$$\boxed{a = b \geq 15 \Rightarrow \neg N(a, b, 0)}$$

1^{er} cas : $\Delta \geq 5$

Si A joue 1, au mieux pour B (il joue 5) la situation évolue ainsi :

$(b, b, 0) \rightarrow (b - 1, b - 5, -1)$

or d'après le lemme 1, $\neg N(b - 1, b - 5, -1)$ et donc $\neg N(b, b, 0)$ cqfd.

$2^{ème}$ cas : $\Delta < 5$

Supposons que A joue $\Delta + 1$ points, la situation évolue ainsi :

$(b , b, 0) \rightarrow (b - \Delta - 1, b - \Delta, 1)$

$b \geq 15 \quad \Rightarrow b - 11 \geq \Delta$

$\Rightarrow 3b - 3\Delta - 3 \geq 2b - 2\Delta + 8$

$\Rightarrow b - \Delta - 1 \geq 1/3 (2(b - \Delta) + 8)$

et donc, d'après le lemme 3, $\neg N(b - \Delta - 1, b - \Delta, 1)$

alors, $\neg N(b , b, 0)$ cqfd.

C.1.4. conclusions

Les règles d'Alésia sont telles que :

1) Il n'existe pas de stratégie pure garantissant la victoire.

2) Il n'existe pas de stratégie pure garantissant le nul.

10 Bibliographie

Aumann R. (1987) *"Game theory" The new Palgrave : a Dictionary of Economics*. The MacMillan Press.

Axelrod, R. 1984. *The evolutioon of Cooperation*. Basic Books, New-York.

Bakker ; Kuniyoshi. 1996. Robot See, Robot Do : An Overview of Robot Imitation *AISB'96 Workshop on Learning in Robots and Animals.*

Barto, A.G. ; Sutton, R.S. and Watkins, C.J. 1990. *Learning and Sequential Decision Making. Learning and Computational Neuroscience*. MIT Press.

Baudet, G. M. 1978. On the branching factor of the alpha-beta pruning algorithm. *Artificial Intelligence*. N°10(2).

Belew, R.K. and Booker, L.B. 1991*. Proceedings of the 4th International Conference on Genetic Algorithm*. San Diego, CA. Morgan Kaufmann.

Belew, R.K. and Vose, M. 1996*. Foundations of Genetic Algorithms IV*. San Diego, CA. Morgan Kaufmann.

Bellman, R. E. 1957. *Dynamic Programming*. Princeton University Press.

Berliner, H.J. 1977. Search and knowledge. *Proceedings of the International Conference on Artificial Intelligence*.

Berliner, H.J. 1980, *Artificial Intelligence*. N°14.

Berliner, H. J.; and Campbell, M. 1984. *Artificial Intelligence*. N°23.

Billings, D. 1995. *Computer Poker.* Master's thesis. University of Alberta, Dept of Computer science.

Billings, D. ; Papp, D. ; Schaeffer, J. Szrafron, D. 1998. Poker as a testbed for AI research. In *Advances in Artificial Intelligence*. Springer-Verlag.

Borel, E. 1938. Applications aux jeux de hasard*. Traité du calcul des probabilités et de ses applications.*, Paris. Gauthier-Villard.

Boursin, J.-L. 1996, *La décision rationnelle*. Economica.

Brown, G.W. 1951. Iterative Solutions of Games by Fictitious Play. In *Activity Analysis of Production and Allocation*. New-York : Willey.

Buro, M. 1997. Toward opening book learning. *Proceedings of the fifteenth International Joint Conference on Artificial Intelligence*. Workshop on Using Game as a testbed for AI Research. Morgan Kaufman editor.

Carmel D. & Markovitch S. 1993. Learning models of opponent's strategy in game playing. *Proceedings of the AAAI Fall Symposium on Intelligent Games : Planning and Learning*, number FS-93-02, Menlo Park, CA. The AAAI Press.

Carmel D. & Markovitch S. 1994. The M* Algorithm: Incorporing Opponent Models into Adversary Search. *CIS Report #9402*. Technion. Haifa. Israel.

Castillo; Melin. 1994. An intelligent system for discovering mathematical models for financial time series preiction. *IEEE Ninth Annual International Conference. Tencon. Singapour.*

Chaitin, G. 1996. Les suites aléatoires. *Le hasard.* Dossier pour la science. Avril 96.

Chapman. 1991. *Vision, instruction and action*'. The MIT Press.

Chase, W. G. and Simon, H. A. 1973. Perception in chess. *Cognitive Psychology.* 4, 55-81.

Chen. 1994. Neural network for financial market prediction. *IEEE Neural Networks '94*

Cliff, D.; Husband, P.; Meyer, J-A.; Wilson, S. 1994. *From Animals to Animats 3. Proceedings of the third International Conference on Simulation of Adaptive Behavior.* MIT Press.

Collins, G. Birnbaum, L., Krulwich, B., Freed, M. 1993. The role of self-models in Learning to plan. *Foundations of Knowledge Acquisition: Machine Learning.* Kluwer Academic Publisher, Boston.

Cypher, A. 1991. Eager : programming repetitive tasks by example. *proceedings of the ACM Conference on Human Factor in Coputing Systems (CHI).*

Debreu. G. 1984. *Théorie de la valeur.* Paris, Dunod.

DeGroot, A.D. 1965. *Thought and choice in chess.* The Hague. Mouton.

Delahaye, J.-P. 1995a. L'altruisme récompensé. *Logique, Informatique et paradoxes.* Pour la science, Diffusion Belin.

Delahaye, J.-P. 1995b. L'altruisme perfectionné. *Logique, Informatique et paradoxes.* Pour la science, Diffusion Belin.

Delahaye, J.-P. 1995c. Vote inconscient. *Logique, Informatique et paradoxes.* Pour la science, Diffusion Belin.

Delahaye, J.-P. et Mathieu, P. 1996a. Expérience sur le dilemme itéré des prisonniers. *Publication interne. LIFL USTL. IT-233.*

Delahaye, J.-P. 1996b. L'espérance mathématique. *Le hasard.* Dossier pour la science. Avril 96.

Delahaye, J.-P.1998. *Jeux Mathématiques et Mathématique des Jeux. Bibliothèque.* Pour la science. Diffusion Belin.

Delbecq. 1995. P6 : Intel lance sa bombe. *Science et vie Micro N°126.*

Demiris; Hayes. 1994. A Robot Controller Using Learning by Imitation. *Proceedings of the 2nd International Symposium on Intelligent Robotic Systems '94.*

Demiris; Hayes. 1996. Imitative Learning Mechanisms In Robots and Humans. *Proceedings of the 5th European Workshop on Learning Robots. '96*

Donnart, J.-Y. 1998. *Architecture Cognitive et Propriétés Adaptatives d'un Animat Motivationnellement Autonome.* Thèse de doctorat Paris 6.

Dent , Boticario. 1992. A personal learning apprentice. *Proceedings of the tenth national conference on artificial intelligence. '92.*

Drevets; Burton; Videen. 1995. Blood flow changes in human somatosensory cortex during anticipated stimulation. *Nature Vol373 '95.*

Du Boisayné, 1985. *De la courbe que décrit un chien en courant après son maitre*. Rapport interne. Ecole polytechniques.

Eisenstein; Kanter; Kessler. 1995. Generation and prediction of time series by a neural network. *Physical Review Letters Vol74 '95.*

Fanning; Cogger. 1994. A comparative analysis of artificial neural networks using financial distress prediction. *International Journal of Intelligent Systems Vol3 '94*

Faller, B. 1985. *La recherche*. N°77.

Fahringer; Basko; Zima. 1992. Automatic performance prediction to support parallelization of FORTRAN programs for massively parallel systems. *Proceedings of the International Conference on Supercomputing.*

Findler, N. 1961. Computer model of gambling and bluffing. *IRE Transactions on Electronic Computer*, EC-10(1):5-6.

Findler, N. 1977. Studies in Machine Cognition using the game of Poker. *Communication of the ACM*, N°20(4).

Findler, N. 1978. Computer Poker. *Scientific American*.

Fondberg. 1986. Amygdala, emotions, motivation, and depressive states. In Plutchik & Kellerman *Emotion : theory, research and experience*. Academic Press.

Fortnow, L. & Whang, D. 1994. *Optimality and Domination in Repeated Games with Bounded Players*. Technical Report TR-94-01. Dept of Computer science. University of Chicago.

Fudenberg D. & Tirole J. 1991. *Game theory*, The MIT Press.

Fudenberg D. & Levine D.K. 1998. *The Theory of Learning in Games*. The MIT Press.

Ganascia, J.-G. 1990. *l'Ame-machine*. Le seuil.

Ganascia, J.-G. 1993. *l'Intelligence Artificielle*. Dominos, Flammarion.

Ganascia, J.-G. 1996. *Les Sciences Cognitives*. Dominos, Flammarion.

Ganascia, J.-G. 1998. *Le trésor de l'informatique et des sciences de l'information* , Edition Flammarion.

Gilboa, I. and Samet, D. 1989. Bounded versus unbounded rationality: The tyranny of the weak. *Games and Economic Behavior*. N°1(3).

Gobet F. & Janson, P. 1994. Towards a chess program based on a model of human memory. In Van den Herik, H.J. & Herschberg I.S. *Advances in computer Chess* N°7. University of Limburg.

Goldberg D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine learning*. Addison-Wesley.

Grefenstette, J.J. 1988. Credit Assignment in Rule discovery Systems Based on Genetic Algorithms. *Machine Learning* . N°3.

Guerrien, B. 1995. *La théorie des jeux*. Economica.

Harsanyi J.C. (1967) Games with incomplete information played by bayesian players. *Management science.* N°14.

Hoffman. 1994. High performance computing and networking for numerical weather prediction. *High-performance Computing and networking. Proceedings Vol2 '94.*

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Holland J. H. ; Holyoak K.J. ; Thagard P.R. ;and Nisbet R.E. 1986. *Induction : Processes of Inference, Learning, and Discovery*. MIT Press, Cambridge.

Hopcroft, J. E. and Ullman, J. D. 1979. *Introduction to Automata Theory, Langages and Computation*. Addison-Wesley, Reading, Mass.

Hsu, F-H. ; Ananthraman, T.S. ; Campbell, M.S. and Nowatzyk, A. 1990. Deep thought. In T.A. Marsland and J. Schaeffer, editors. *Computers, chess and Cognition*. Springer New York.

Iida, H. ; Kotani, I. ; Uiterwijk, J.W.H.M. anf Van der Herik, H.J. 1997, Gains and Risks of OM search. In *Advances in Computer Chess.* N°8. CS dept. of University of Maastricht.

Janikow, C.Z. and Michalewicz, Z. 1991. An experimental comparison of binary and floating point representations in genetic algorithm. *Proceddings of the 4th International Conference on Genetic Algorithm*. San Diego, CA. Morgan Kaufmann.

Janson, P. Problematic positions and speculative play. In T.A. Marsland and J. Schaeffer, editors, *Computers, chess and Cognition*. Springer nesw York.

Jin-Jen; Yagle, 1995. Similarities and differences between one-sided and two-sided linear prediction. *IEEE TSP Vol43 '95.*

Jung; Park. 1994. Prediction of human reach posture using a neural network for ergonomic man models. *Computers & Industrial Engineeing Vol27 '94.*

Kalai, E. 1990. Bounded ratio-nality and strategic complexity in repeated games. In Ichiishi, I. Neyman, A. and Tauman, Y. *Game Theory and Applications*. Academic Press, San-Diego.

Kandel; Boe; Orliaguet. 1993. Visual detection of coarticulatory anticipation or ... guessing what has not yet been written. *IEEE Virtual Reality '93.*

Knoblauch, V. 1994. Computable strategies for repeated prisoner's dilemma*. Games and Economic Behavior.* N°7(3).

Knuth, D. E. and Moore, R.W. 1975. An analysis of alpha-beta pruning. *Artificial Intelligence*. N°6(4).

Korf, R.E. 1989. Generalized games trees. In *Proceedings of the International Joint Conference on Artificial Intelligence*. Detroit, MI.

Koza, J.R. 1992. *Genetic Programming*. MIT Press, Cambridge, MA.

Koza, J.R. 1994. *Genetic Programming - 2*. MIT Press, Cambridge, MA.

Kreps D. 1991. *Games theory and Economic Modelling*, Oxford University Press.

Krishnan; Vitter. 1994. Optimal prediction for perfecting in the worst case. *ACM-SIAM Symposium on Discrete Algorithms '94*

Lenat, D.B. 1983. *Artificial Intelligence*. N°21.

Lenat, D.B. 1984. The role of heuristics in learning by discovery. *Machine learning : An artificial intelligence approach*. Springer-Verlag.

Lipton, R.J. and Young N.E. Simple Strategies for Large Zero-sum Games with Application to Complexity Theory. (Publication inconnue à ce jour).

Littman, M.L. 1994. Markov games as a framework for multi-agent reinforcement learning. *Proceedings of the Internationale Conference on Machine Learning*. Morgan Kaufmann.

Maes, P. 1995. Les agents informatiques. *Pour la science N°217.*

Maes, Kozierok. 1993. Learning interface agents. *Proceedings of the eleventh national conference on artificial intelligence.*

Maes, P.; Mataric, M-J.; Meyer, J-A.; Pollack, J.; Wilson, S. 1996. *From Animals to Animats 4. Proceedings of the fourth International Conference on Simulation of Adaptive Behavior*. MIT Press.

Mahlke; Hank; Bringmann. 1994. Characterizing the impact of predicated execution on branch prediction. *Proceedings of the 27th international Symposium on Microarchitecture. Micro 27.*

Mammela; Kaasila. 1994. Prediction,smoothing and interpolation in adaptive diversity reception. *IEEE ISSSIA '94*

Maynard Smith J. 1982. *Evolution and theory of Games*. Cambridge University Press

Meyer C. and Ganascia J.G. 1996a. *S.A.G.A.C.E. Solution Algorithmique Génétique pour l'Anticipation de Comportements Evolutifs*. Paris VI, LAFORIA N°96/32.

Meyer C. and Ganascia J.G. 1996b. Utilization of imitation and anticipation mechanisms to bootstrap an evolutive distributed artificial intelligence system. *Animal Societies as an Alternative Metaphorical Basis for DAI - Workshop International Conference on Multi-Agent Systems.*

Meyer C.; Ganascia J-G ; Zucker J-D. 1997a. Learning Strategies in Games by Anticipation. *Proceedings of the fifteenth International Joint Conference on Artificial Intelligence, IJCAI'97*. Morgan Kaufman editor.

Meyer C., Ganascia J.-G., Zucker J.-D. 1997b. Modélisation de stratégies par Apprentissage et Anticipation génétiques. *Actes de conférence des Journées Ingénierie des Connaissances et Apprentissage Automatiques.*

Meyer C. ; Akoulchina I. ; Ganascia J.G. 1997c. Two Approaches of Human Behavior Anticipation . Proceedings of *ICTAI'97 International Conference of Tools with Artificial Intelligence*. USA. IEEE Press.

Meyer C., Zucker J.-D. 1999. Mind-Reading Machine: an inquiry into adversary modelling and anticipation on imperfect information games. *Actes de conférence de : Conférence d'Apprentissage (CAP'99)*. Ecole Polytechnique.

Meyer J.-A. 1980. *La théorie des jeux a 2 joueurs et ses applications écologiques*. UA 258 CNRS.

Meyer, J-A. 1995. *AnimatLab - Modélisation des comportements adaptatifs.* Rapport d'activités. CNRS. Juin 95.

Meyer, J-A. & Wilson, S. 1990. *From Animals to Animat: Proceedings of the first International Conference on Simulation of Adaptive Behavior*. MIT Press.

Meyer, J-A.; Roitblat, H.; Wilson, S. 1992*. From Animals to Animats 2. Proceedings of the second International Conference on Simulation of Adaptive Behavior*. MIT Press.

Michalewicz, Z. 1996. *Genetic Algorithm + Data Structures = Evolution Programs*. Third, Revised and Extented Edition. Springer.

Miller, G.A. 1956. The magical number seven plus or minus two : Some limits on our capacity for processing information. *Psychological Review*. N°63.

Mills. 1994. COMET III : object-oriented network performance prediction. *CSC été '94*

Minasi M. 1991. Recognizing Patterns. *AI Expert*, February.

Minsky, M.L. 1961. Steps toward artificial intelligence*. Proceedings of the institute of Radio Engineers*, N°49.

Mitchell, M. 1996. *An introduction to Genetic Algorithms*. MIT Press.

Moore, A. et atkeson, C.G. 1993. Prioritized sweeping: Reinforcement Learning with less data and less real time. *Machine Learning*, 13.

Morris P. 1994. *Introduction to Game Theory*. Springer.

Mozer. 1994. Neural network music composition by prediction : exploring the benefits of psychoacoustic constraints and multi-scale processing. *Connection Science Vol6.*

Myerson R.B. 1991. *Game Theory Analysis of Conflict.* Harvard University Press.

Nash, J.F. 1950. Equilibrum point in n-person games *. Proceedings of national Acadmy of Sciences*. U.S.A. 36:46-49.

Nash, J.F. 1951. NonCooperative games. *Annals of Mathematics*., 54-289.

Neves; de Almeida. 1994. B-ISDN connection admission control and routing strategy with traffic prediction by neural networks. *Proceedings of SUPERCOMM '94.* New Orleans*.*

Newborn, M. 1977. The efficiency of the alpha-beta search on trees with branch-dependent terminal node scores. *Artificial Intelligence*. N°8(2).

Newell, A.; Shaw, J.; and Simon H.A. 1957. Empirical explorations of the logic theory machine. *Proceedings of the West Joint Computer Conference*. Vol 15.

Newell, A.; Shaw, J.; and Simon H.A. 1958. Chess-playing programs and the problem of complexity. *IBM journal R&D*. vol 2.

Noviello; Serio; Plaitano. 1993. Battery diagnostics and performance prediction : computational vs. expert system based approach. *INTELEC 93. 15th International Telecommunication Energy Conference.*

Owen G. 1995. *Game Theory* . Academic Press.

Papp, D. R. 1998. *Dealing with Imperfect Information in Poker*. Master's thesis. University of Alberta, Dept of Computer science.

Pfeifer, R.; Blumberg, B.; Meyer, J-A.; Wilson, S. 1998. *From Animals to Animats 5. Proceedings of the fifth International Conference on Simulation of Adaptive Behavior*. MIT Press.

Pingaud, F. ; Germe J.F. 1984. *50 jeux avec du papier et des crayons*. Edition du Rocher.

Pitrat, J. 1977. *Artificial Intelligence*. N° 8.

Popescu, R. 1984. *Chelem, un système expert pour trouver la ligne du déclarant au bridge*. Thèse de l'Université Paris *6*.

Press, W. H.; Flannery,B.P.;Teukolsky, S.A.; and Vetterling, W.T. 1986. *Numerical Recipes : the art of Scientific Computing*. Cambridge university Press.

Prochazka. 1994. Time series prediction using genetically trained wavelet networks'. *Proceedings of IEEE Workshop '94.*

Rapoport, A. and Chammah, A. 1965. *Prisonner's dilemma*. Ann Arbor. University of Michigan Press.

Rattermann, M.-J. and Epstein, S.L. 1995. Skilled like a person: A comparison of human and computer game playing . *Proceedings of the Seventennth Annual Conference of the Cognitive Science Society*. Pittsburgh. L. Erlbaum Associates.

Ricaud, P. 1995. *GOBELIN : Une approche Pragmatique de l'Abstraction Appliquée à la Modélisation de la Stratégie Elémentaire du Jeu de Go*. Thèse de doctorat Paris 6.

Ricaud, P. 1997. A model of Strategy for the Game of Go using Abstraction Mechanisms. *Proceedings of the fifteenth International Joint Conference on Artificial Intelligence, IJCAI'97*. Morgan Kaufman editor.

Riolo. 1992. Lookahead planning and latent learning in a classifier system. *Proceedings of Simulation of Adaptive Behavior. MIT Press.*

Robinson, J. 1951. An Iterative Method of Solving a Game. *Annals of Mathematics*, 54. pp296-301.

Rochenstein. 1985. Formal theories of knowledge in AI and robotics. *New generation computing '85.*

Rolland, P.Y. 1998. FlExPat : a Novel Algorithm for Musical Pattern Discovery. *Proceedings of the 12th Colloquium on Musical Informatics (XII CIM)*. Co-sponsored by IEEE (CS/TCCGM). Gorizia, Italy, September 24-26, 1998.

Rong Li; Bar-Shalom. 1994. A hybrid conditional averaging technique for performance prediction of algorithms with continuous and discrete uncertainties. *Proceedings of the 1994 American Control Conference*

Rosin C.D. and Belew. R.K. 1995. “Finding opponents worth beating: Methods for competitive co-evolution”. *Proceedings of the 6th International Conference on Genetic Algorithm*.

Russ; Farber. 1993. Real-time prediction of sensor images using computer graphic hardware for autonomous mobile robots in complex structured environments. *EDUGRAPHICS ’93.*

Samuel, A.L. 1959. Some studies in machine learning using the game of checkers. *IBM Journal on Research and Development*, 3:211-229.

Samuel, A.L. 1967. Some studies in machine learning using the game of checkers. II-Recent progress. *IBM Journal on Research and Development*, 11:601-617.

Samuelson. 1947. *Foundation of Economic Analysis*. Cambridge, Mass.

Satoh, Funatzu. 1995. SOPHIA, a knowledge base-guided reaction prediction system-utilization of a knowledge base derived from a reaction database. *Journal of chemical information Vol35 ’95.*

Schaeffer, J. ; Culberson, J. ;Treolar, N. ; Knight, B. ;Lu, P. ; and Szafron, D. 1992. A world championship caliber checkers program. *Artificial Intelligence* N°53.

Schaeffer, J. ; Lake, R ; Lu, P ; and Bryant, M. 1996. « Chinook : the man-machine world checkers champion ». *AI Magazine* N°17:1.

Shannon C.E. 1950. Programming a computer for playing chess*. Philosophical Magazine* (Series 7) Vol.41.

Shannon C.E. 1953. *A Mind-Reading (?) Machine*, Bell Laboratories Memorandum, March.

Schlimmer, Fisher. 1986. A case study of incremental concept induction. *Proceedings of the fifth national conference on artificial intelligence ’86.*

Schlimmer,Hermens. 1993. Software Agents : Completing pattern and constructing user interface. *Journal of intelligence research ’93.*

Shiffrin R.M., and Atkinson, R.C. 1969. Storage and retrival processes in long-term memory. *Psychological review*. 76, 179-193.

Simon, H. A. 1947. *Administrative Behaviour, a Study of Decision Making Processus in Administrative Organization*. Macmilan. New York.

Syrmos; Misra; Aripirala. 1995. On the discrete generalized Lyapunov equation. *Automatica Vol31 Feb.1995.*

Sutton, R.S. 1984. *Temporal Credit Assignment in Reinforcement Learning*. Doctoral dissertation. Dept. of Computer Science. University of Massachusetts.

Sutton & Pinette. 1985. The learning of world models by connectionist networks. *Proceeding of the Conference of the cognitive science society.*

Sutton, R.S. 1992. Reinforcement learning architectures for animats. *Proceedings of Simulation of Adaptive Behavior.* Mit Press.

Sutton, R. S. ; Barto, G. 1998. *Reinforcement Learning : An introduction*. MIT Press. Cambridge.

Stern L. 1985. *The structure and strategies of Human memory*. The Dorsey Press.

Tadokoro; Ishikawa; Takebe. 1993. Stochastic prediction of human motion and control of robots in the service of human. *Proceedings of the International Conference on Systems, man and cybernetics.*

Tadokoro; Takebe; Ishikawa. 1996. Control of human cooperative robots based on stochastic prediction of human motion. *Proceedings of the 2nd IEEE International Workshop on Robot and Human Communication.*

Tesauro, G.J. 1994. TD-Gammon, a self-teaching backgammon program, achieves mastrelevel play. *Neural Computation*, 6(2) :215-219.

Tesauro, G.J. 1995. Temporal difference learning and TD-Gammon. *Communications of the ACM*, N°38.

Triantafyllos; Vassiliadis; Kobrosly. 1995. On the prediction of computer implementation faults via static error prediction models. *Journal of Systems and software Vol28 '95.*

Tyson. 1994. The effect of predicated execution on branch prediction. *94' Micro 27*

Uther, W. and Veloso, M. 1997. *Adversal Reinforcement Learning*. Carnegie Mellon University. Pittsburgh, PA.

Vajda, S. 1961. *Mathematical programming*. Addison-Wesley.

Van Damme E. 1987. *Stability and Perfection of Nash Equilibrum*, Springer Verlag, Berlin.

von Neumann J. and Morgenstern O. 1944. *Theory of Games and Economic Behavior*, Princeton University Press.

Vuurpijl; Shouten; Vytopil. 1994. A scalable performance prediction method for parallel neural network simulations. *Proceedings of High-Performance Computing and Networking Vol1 '94*

Wasserman, A. 1970. Realization of a skillfull bridge bidding program. *Proceedings of Fall joint computer conference*.

Waterman, D.A. 1970. *Artificial Intelligence*. N°1.

Watkins, C.J.C.H. 1989. *Learning from Delayed Rewards*. Ph.D. thesis. Cambridge University.

Watkins, C.J.C.H. and Dayan, P. 1992. Q-Learning. *Machine Learning*, 8 :279-292.

Waugh, N.C. and Norman, D.A. 1965. Primary memory. *Psychological review*. 72, 89-104.

Widrow, B. ; Gupta, N. K. ; and Maitra, S. 1973. Punish/reward : Learning with a critic in adaptive threshold systems. *IEEE Translation on Systems, Man, and Cybernetics*, 3 :455-465.

Wilkins, D. 1979, Using plans in Chess. *Proceedings of the International joint Conference on Artificial intelligence*. IEEE Press.

Wilson, S.W. and Goldberg, D.E. 1989. A critical review of Classifier Systems. *Proceedings Of the third International conference on Genetic Algorithm.*

Wust; Van Noort. 1994. Neural network current prediction for shipping guidance'. *Proceedings* of the OC*EANS Conference.*