Représentation et algorithmique des jeux à information incomplète. Application à la conservation de la biodiversité

Paul Jourdan

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *24 Novembre 2022* **par :**

Paul Jourdan

Représentation et algorithmique des jeux à information incomplète. Application à la conservation de la biodiversité

JURY

Aurélie BEYNIER	MdC-HDR, Sorbonne Université	Examinatrice
Hélène FARGIER	DR CNRS, IRIT	Co-directrice de thèse
Rida LARAKI	DR, Université Paris Dauphine	Examinateur
Régis SABBADIN	DR, INRAE	Co-directeur de thèse
Olivier SPANJAARD	MdC-HDR, Sorbonne Université	Rapporteur
Guillaume VIGERAL	MdC-HDR, Université Paris Dauphine	Rapporteur

---

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle
Unité de Recherche :
Unité MIAT-INRAE
Directeur(s) de Thèse :
Hélène FARGIER et *Régis SABBADIN*
Rapporteurs :
Olivier SPANJAARD et *Guillaume VIGERAL*

Remerciements

Et voilà, c'est la fin de plusieurs années de dur travail.

Je tiens tout d'abord à remercier mes directeurs de thèse, Régis et Hélène pour avoir rendu cette thèse possible. Merci pour m'avoir guidé et supporté durant ces trois années.

Je tiens à remercier Guillaume Vigeral et Oliver Spanjaard pour avoir accepté de rapporter ma thèse. Je souhaite également remercier Aurelie Beynier et Rida Laraki d'avoir accepté de participer à mon jury de thèse. Merci pour l'intérêt que vous avez porté à mes travaux.

Je remercie également Jérôme Lang, Jérôme Renault et Joon Kwon d'avoir accepté de participer à mes deux comités de thèse.

Je tiens à remercier tous les membres de l'unité MIAT de l'INRAE. J'ai de la chance d'avoir été accueilli dans une équipe si chaleureuse et bienveillante. Les repas du midi et les pauses-café ont toujours été des moments agréables. Être si bien entouré m'a permis de travailler sereinement. Merci, Eric et Patrick pour l'aide et les conseils apportés afin de mettre à disposition mes travaux et utiliser les serveurs de calculs. Merci, Sylvain pour les paroles rassurantes et motivantes dont tout thésard a besoin.

Merci à mes camarades thésards, Khaoula, Anwar, Valentin, Marianne, Pierre, Loïc, Hanna, Aurélie. Je vous souhaite le meilleur pour votre thèse. Ceux d'entre vous qui ont été dans le même bureau que moi, merci d'avoir supporté mes tirades et râlements.

Et puis finalement, en dehors du "monde scientifique", je tiens à remercier ma famille pour leur soutien constant.

Je souhaite remercier mes grands parents, qui se sont toujours soucié de mon bien-être lors des périodes difficiles.

Et enfin, mes parents, Annie et Stéphane, pour l'amour et le soutien qu'ils m'ont toujours apporté.

Résumé

La théorie des jeux est un cadre mathématique utilisé pour modéliser et résoudre des problèmes de décision multi-agent dans une multitude de domaines différents comme en économie ou pour la protection de la biodiversité.

Certains modèles ont été proposés pour des problèmes plus difficiles où l'information sur le jeu joué est incomplète. Le cadre des jeux bayésiens, par exemple, modélise des jeux où les joueurs possèdent une information partielle sur le jeu dans lequel ils vont jouer.

Ils ont connaissance d'un ensemble de jeux "possibles", sans savoir dans lequel ils jouent effectivement.

La représentation "concise" de jeux à grand nombre de joueurs est également un enjeu important, puisque la taille de représentation d'un jeu est exponentielle en le nombre de joueurs d'un jeu. Le cadre des jeux hypergraphiques par exemple, permet une représentation concise de jeux dans lesquels les joueurs n'interagissent pas forcément avec tous les autres. Un hypergraphe est utilisé pour modéliser ces interactions partielles. Les noeuds de l'hypergraphe modélisent l'ensemble des joueurs, et une hyperarête modélise les joueurs impliqués dans un jeu "local". L'utilité globale d'un joueur est la somme des utilités obtenues dans chacun des jeux locaux.

Cette thèse est dédiée à la conception de méthodes de calcul d'équilibre de Nash dans les jeux à information incomplète concis. Dans un premier temps, nous montrons qu'il existe une transformation entre les jeux à information incomplète et les jeux hypergraphiques. Ceci nous permet de définir le cadre des "jeux bayésiens hypergraphiques", généralisant les deux cadres précités.

Nous nous intéressons ensuite au calcul des équilibres de Nash mixtes "exacts" dans ces jeux. Nous décrivons un algorithme permettant de calculer ces équilibres dans des jeux sous forme normale ainsi que dans des jeux hypergraphiques et, par extension, bayésiens ou bayésiens hypergraphiques.

Nous illustrons finalement les contributions théoriques de la thèse sur un problème de conservation de la biodiversité dans un cadre de protection d'espèce animale face au braconnage. Nous modélisons ce problème dans le cadre des jeux bayésiens à forme normale, puis nous montrons comment obtenir un modèle équivalent dans le cadre des jeux bayésiens polymatriciels. Cette transformations permet le calcul d'équilibre dans des problèmes de conservation de grande taille.

Mots-clés

Intelligence artificielle, théorie des jeux, jeux hypergraphiques, jeux bayésiens, conservation de la biodiversité

Abstract

Game theory is a mathematical framework used to model and solve multi agent decision problems in a multitude of different domains like in economy or protection of biodiversity.

Some models were proposed for more difficult problems where information on the played game is incomplete. The Bayesian games framework, for instance, models games where players have a partial information on the game in which they will play. They have a knowledge of a set of "possible" games, without knowing the one in which they're actually playing.

The "concise" representation of games with a great number of players is also an important issue, because the representation size of a game is exponential in the number of players of the game. The framework of hypergraphical games, for instance, allows a concise representation of games where players do not necessarily interact with every other player. A hypergraph is used to model these partial interactions. The nodes of an hypergraph model the set of players and the hyperedges model the players involved in a "local" game. The global utility of a player is the sum of the utility obtained in every local games.

This PhD is dedicated to the design of Nash equilibrium computation in concise games with incomplete information. First, we show that there exists a transformation between games with incomplete information and hypergraphical games. This allows us to define the framework of "Bayesian hypergraphical games", a generalisation of the two previously mentionned frameworks.

We then focus on the computation of "exact" mixed Nash equilibria. We describe an algorithm allowing to compute these equilibria in normal form games as well as in hypergraphical games and, by extension, Bayesian and Bayesian hypergraphical games.

Finaly, we illustrate the theoretical contribution of the PhD on a problem of biodiversity conservation in a case of animal protection against poaching. We model this problem in the framework of normal form bayesian games, then show how to get an equivalent model in the framework of bayesian polymatrix games. These tranformation allows the computation of equilibria in problems of large size.

Keywords

Artificial intelligence, game theory, hypergraphical games, Bayesian games, biodiversity conservation

Table des matières

Introduction

La théorie des jeux [Von Neumann et Morgenstern, 1944] est un cadre d'étude mathématique utilisé pour modéliser des situations de prise de décisions entre plusieurs agents. Lorsque tous les agents choisissent et appliquent simultanément leur propre stratégie (définissant ainsi une stratégie jointe), ils obtiennent chacun une utilité individuelle. On considère qu'un joueur est un acteur rationnel qui a pour objectif de maximiser sa propre utilité. Lorsqu'il s'agit d'étudier un jeu, il existe plusieurs concepts de solution. Celui auquel nous allons principalement nous intéresser est l'équilibre de Nash. C'est une stratégie jointe dont aucun des joueurs n'a intérêt à dévier unilatéralement. Aussi, un équilibre de Nash a un double intérêt. Il permet tout d'abord de "prédire" le comportement d'agents rationnels en interaction. En effet, puisqu'il correspond à une stratégie jointe dont les joueurs n'ont pas intérêt à dévier, il est "plausible" qu'en cas d'interactions répétées, les comportements des agents "convergent" vers cette stratégie jointe. D'un autre côté, l'équilibre de Nash a un rôle "prescriptif". Si un outil d'aide à la décision propose à chaque agent sa stratégie individuelle correspondant à un équilibre de Nash, les agents "savent" qu'il ont tout intérêt à suivre la stratégie qui leur est proposée. Par ailleurs, il est établi qu'un jeu quelconque admet forcément (au moins) un équilibre de Nash [Nash, 1950]. Ceci explique pourquoi ce concept, bien que parfois discuté (un équilibre de Nash ne correspond pas forcément à la "meilleure" stratégie jointe, mais à une stratégie jointe "stable"), soit considéré comme le plus important de la théorie des jeux non-coopératifs. C'est pourquoi nous nous intéresserons particulièrement au calcul d'équilibres de Nash dans les familles de jeux que nous allons étudier.

La théorie des jeux est utilisée dans une multitude de domaines. A l'origine, elle a été conçue pour une application en économie, un domaine où elle reste très utilisée. Nous allons toutefois nous intéresser à son application dans le domaine de la protection de la biodiversité, plus précisément dans des situations de lutte contre le braconnage. D'autres travaux ont déjà exploité la théorie des jeux et le concept d'équilibre de Nash pour étudier des situations liées à la protection de la biodiversité. Par exemple, [Gibson et Marks, 1995] se sont intéressés à l'étude des raisons des échecs des politiques de lutte contre le braconnage en Afrique. Ou encore, [Frank et Sarkar, 2010] se sont préoccupés de diverses situation de gestion de la faune. Nous allons proposer, afin d'illustrer nos contributions méthodologiques et informatiques, un cadre de modélisation et des outils pour l'étude de problèmes de lutte contre le braconnage d'espèces protégées.

Quel que soit le cadre d'utilisation de la théorie des jeux, nous sommes confrontés à des problèmes récurrents. Le jeu peut tout d'abord être composé d'un grand nombre de joueurs, ce qui rend plus difficiles sa représentation (les tables d'utilités sont de taille exponentielle en le nombre de joueurs) et sa résolution (la recherche d'un équilibre de Nash notamment). Par ailleurs, dans certains jeux, la connaissance des joueurs sur le jeu est imparfaite. Les utilités des autres joueurs peuvent être mal connues, par exemple. Lorsque nous souhaitons modéliser des jeux avec de nombreux joueurs, ayant

possiblement une connaissance imparfaite du jeu auquel ils participent, nous sommes confrontés à des problèmes de taille de représentation et de complexité de la recherche d'une solution. La première contribution de cette thèse est la mise en place d'un cadre de représentation concise pour des jeux comprenant un grand nombre de joueurs et où les joueurs ne disposent que d'une information partielle sur le jeu joué.

La deuxième contribution de cette thèse est la définition d'un nouvel algorithme pour calculer un équilibre de Nash mixte de manière exacte pour les jeux que nous étudions. L'algorithme que nous avons proposé se base sur la l'approche de [Wilson, 1971] que nous avons "modernisée" et étendue aux représentations concises de jeux à information potentiellement incomplète.

Nos contributions ont donné lieu à la mise à disposition d'une boite à outils (Python, Sagemath) téléchargeable ou utilisable en ligne [1].

Plan du manuscrit : La première partie de la thèse, les chapitres 1, 2 et 3 présente les différentes notions de théorie des jeux qui interviendront dans la thèse. Dans le chapitre 1, nous présentons les concepts de base des jeux à informations complète ainsi que quelques algorithmes de calcul d'équilibre de Nash. Dans le chapitre 2, nous présentons des cadres de représentation succincte des jeux à information complète. Dans le chapitre 3, nous décrivons les jeux à information incomplète.

La seconde partie présente nos contributions. Dans le chapitre 4, le cadre des *jeux bayésiens hypergraphiques* est présenté, ainsi que ses propriétés (complexité de représentation, liens avec le cadre des jeux hypergraphiques, ...). Dans le chapitre 5, nous présentons un nouvel algorithme de calcul d'équilibre de Nash "exact" dans les jeux à plus de 2 joueurs. Les extensions de cet algorithme aux jeux dégénérés et hypergraphiques sont aussi présentées. Dans le chapitre 6, les contributions précédentes sont illustrées sur un problème de lutte contre le braconnage. Nous présentons enfin dans le chapitre 7 la boite à outil *gtnash* développée pour modéliser et résoudre différents types de jeux. Nous décrivons également les expérimentations effectuées.

1. `https://forgemia.inra.fr/game-theory-tools-group/gtnash-git/`

Première partie

Contexte et préliminaire

Chapitre 1

Jeux à information complète

1.1 Introduction

La théorie des jeux [Von Neumann et Morgenstern, 1944] désigne un cadre mathématique utilisé afin de modéliser et analyser des situations de prise de décision entre plusieurs acteurs. Une situation sera formulée sous forme d'un jeu où les acteurs prenant des décisions seront définis comme des joueurs dont les choix possibles sont nommés actions ou stratégies. On considère que les joueurs se comportent rationnellement, c'est-à-dire que leur objectif sera de maximiser leurs bénéfices en fonction de leurs connaissances et des stratégies des autres joueurs.

En fonction des problèmes étudiés les buts des joueurs et leurs relations les uns vis-à-vis des autres seront de différentes natures, il est donc possible de distinguer les jeux coopératifs et les jeux compétitifs.

Dans un jeu coopératif, les joueurs collaborent en formant des coalitions avec d'autres joueurs pour atteindre un but commun. Les coalitions sont en compétition les unes avec les autres. Une fonction spécifie quelle est la valeur de chacune des coalitions formées, les joueurs membres d'une même coalition peuvent avoir des valeurs différentes.

Un jeu compétitif est un jeu dans lequel chaque joueur dispose de sa propre fonction d'utilité et cherche à optimiser sa propre utilité. Un joueur joue donc selon son propre intérêt (son utilité) cependant la fonction d'utilité peut servir à capturer une préoccupation vis-à-vis des autres joueurs.

Dans cette thèse, nous nous préoccuperons principalement des jeux compétitifs [Von Neumann et Morgenstern, 1944, Nisan, 2007].

Il est possible de distinguer deux types de jeux compétitifs lorsqu'il s'agit d'analyser la prise de décision des joueurs, les jeux simultanés et les jeux séquentiels [Osborne et Rubinstein, 1994].

Dans un jeu simultané, aussi appelé jeu statique, les joueurs sélectionnent leurs stratégies en même temps, sans avoir connaissance des stratégies jouées par les autres joueurs. Par exemple le jeu "Pierre, feuille, ciseaux" peut être modélisé sous forme d'un jeu (compétitif) simultané. Nous avons deux joueurs qui ont chacun 3 stratégies : pierre, feuille et ciseau. Dans ce jeu, la pierre bat le ciseau, le ciseau bat la feuille et la feuille bat la pierre. Si la même stratégie est jouée par les deux joueurs alors il y a une égalité. Chaque joueur a connaissance des stratégies possibles de son adversaire mais ne sait pas laquelle il va jouer. Les joueurs choisissent leurs stratégies de manière simultanée.

		Joueur 2		
		P	F	C
Joueur 1	P	0,0	-1,1	1,-1
	F	1,-1	0,0	-1,1
	C	-1,1	1,-1	0,0

FIGURE 1.1 – Représentation d'un jeu simultané à 2 joueurs de "Pierre, Feuille, Ciseau"

Dans le cas de "Pierre, Feuille, Ciseau", ce jeu est à somme nulle, c'est-à-dire que si l'on somme les utilités de chacun des joueurs pour chaque stratégie jointe possible nous obtenons 0.

Un autre jeu à 2 joueurs, qui est lui à somme générale (non nulle), est le jeu de "Guerre des Sexes". Un couple s'est donné rendez-vous, ils ont le choix entre aller voir un match de football ou de boxe. L'homme préférerait aller voir le match de football tandis que la femme préférerait voir le match de boxe. Tous deux préféreraient être avec leur conjoint. Ils ne peuvent pas communiquer.

		Joueur 2	
		F	*B*
Joueur 1	*F*	3, 2	0, 0
	B	0, 0	2, 3

FIGURE 1.2 – Représentation d'un jeu simultané à 2 joueurs de "Guerre des sexes"

Dans un jeu séquentiel, les joueurs jouent à tour de rôle dans un ordre déterminé, ils ont connaissance des actions jouées par les autres joueurs qui ont joué avant eux. Les actions qu'un joueur peut jouer vont parfois dépendre des actions jouées précédemment par les autres joueurs. Les échecs et le morpion sont deux exemples de jeux séquentiels.

En fonction du problème à modéliser, il y a plusieurs cadres de représentations possibles. Les différents cadres existants permettant de tirer parti des différentes propriétés d'un problème et donc diffèrent type de jeux. Plusieurs représentations seront présentées comme les jeux sous forme normale généralement utilisés pour les jeux compétitifs simultanés, les jeux sous forme extensive pour les jeux séquentiels ou à information incomplète/imparfaite.

Plusieurs concepts de solution existent lorsqu'il s'agit de "résoudre" un jeu. Pour une "solution", il va s'agir de déterminer quelles sont les stratégies que chacun des joueurs doit jouer en tenant compte de celles des autres joueurs pour atteindre un état que l'on peut définir comme désirable pour tous les joueurs. L'un de ces concepts les plus importants en théorie des jeux est l'équilibre de Nash [Nash, 1950], une stratégie pour laquelle aucun des joueurs ne peut dévier de manière unilatérale et améliorer son utilité. Il existe d'autres types de "solution" comme les équilibres corrélés [Aumann, 1974], les stratégies Pareto optimales, etc...

Dans ce chapitre, nous allons parler des jeux à information complète, leurs représentations et les concepts de solution. Nous aborderons ensuite les représentations succinctes de ces jeux dans le chapitre 2 ainsi que les jeux à information incomplète dans le chapitre 3.

1.2 Jeux à information complète

Afin de pouvoir modéliser un problème sous la forme d'un jeu et le résoudre, il faut choisir la représentation à utiliser. La représentation du jeu est équivalente à une structure de données qui permet de représenter toutes les informations nécessaires à la définition du jeu. Dans la suite de cette section nous allons présenter les deux représentations les plus classiques.

1.2.1 Jeux sous forme normale

Il est possible de représenter les jeux simultanés via ce que l'on appelle, la forme normale standard (que nous désignerons par forme normale par la suite), aussi parfois appelée forme matricielle, pour les jeux à deux joueurs.

La forme normale consiste à représenter un jeu en listant toutes les stratégies jointes possibles et toutes les utilités associées à chacune de ces stratégies jointes pour tous les joueurs. L'utilité capture le gain ou la perte du joueur.

Définition 1 (Jeux sous forme normale)**.** *Un jeu sous forme normale est défini par un triplet $\langle S, \Omega, u \rangle$ où :*

- $S = \{1, ..., N\}$ *est l'ensemble des N joueurs participant au jeu*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des stratégies jointes.* Ω_n *est l'ensemble des stratégies du joueur n et* $\omega = (\omega_1, ..., \omega_N)$ *est une stratégie jointe.* ω_n *est la stratégie jouée par le joueur n.*
- $u = \{u_n : \Omega \to \mathbb{R}\}_{n \in S}$ *est l'ensemble des fonctions d'utilité des N joueurs.* $u_n(\omega)$ *correspond à l'utilité du joueur n lorsque la stratégie jointe* ω *est jouée.*

L'écriture ω_{-n} correspondra à l'ensemble des stratégies jouées par les joueurs autres que n : $\omega_{-n} = (\omega_1, ..., \omega_{n-1}, \omega_{n+1}, ..., \omega_N)$. Donc pour une action jointe $\omega \in \Omega$ nous avons $\omega = (\omega_n, \omega_{-n}), \forall n \in \{1, .., N\}$ où la notation (ω_n, ω_{-n}) désignera la concaténation de la stratégie ω_n et de la stratégie jointe ω_{-n}

Dans un jeu sous forme normale, les utilités peuvent être représentées dans une table d'utilité de N dimensions où les données de chaque cellule possible représentent l'utilité des joueurs. Dans le cas d'un jeu à deux joueurs, les lignes de la table correspondent aux stratégies du premier joueur et les colonnes aux stratégies du second joueur.

Exemple 1 (Dilemme du prisonnier)**.** *Deux criminels sont arrêtés par la police et placés dans deux pièces isolées l'une de l'autre pour y être interrogés. Chacun des criminels se voit proposer une offre permettant de lui éviter la prison s'il dénonce son acolyte. Il y a trois situations possibles. L'un des criminels dénonce l'autre, lui permettant d'échapper à la prison mais condamnant son complice à 5 ans de prison. Si les deux criminels se dénoncent l'un l'autre, ils devront tous les deux faire 3 années de prison. Si aucun des criminels ne parle alors ils feront juste 1 an de prison. Cette situation peut être formulée sous la forme d'un jeu à 2 joueurs où chaque joueur a deux actions, rester silencieux (S) ou trahir l'autre prisonnier (T).*

- $S = \{1, 2\}$
- $\Omega = \Omega_1 \times \Omega_2 = \{(S, S), (S, T), (T, S), (T, T)\}$ *avec* $\Omega_n = \{S, T\}$
- $u = \{u_1, u_2\}$

La table d'utilité du jeu correspondant se trouve dans la figure 1.3 où la valeur de l'utilité correspond au nombre d'années passées en prison.

		Joueur 2	
		S	T
Joueur 1	S	$-1, -1$	$-5, 0$
	T	$0, -5$	$-3, -3$

FIGURE 1.3 – Représentation du dilemme du prisonnier sous forme normale standard

Cependant représenter les utilités comme dans la figure 1.3 n'est pas toujours très pratique lorsque nous avons des jeux avec 3 joueurs ou plus. Sur un dilemme du prisonnier entre 3 joueurs, nous avons un table d'utilité comme dans la figure 1.4.

ω_1	ω_2	ω_3	u_1	u_2	u_3
S	S	S	-3	-3	-3
S	S	T	-7	-7	-1
S	T	S	-7	-1	-7
S	T	T	-10	-5	-5
T	S	S	-1	-7	-7
T	S	T	-5	-10	-5
T	T	S	-5	-5	-10
T	T	T	-9	-9	-9

FIGURE 1.4 – Représentation du dilemme du prisonnier à 3 joueurs

Cette seconde représentation graphique est plus pratique lorsqu'il y a plus de 2 joueurs, donc nous utiliserons celle-ci lorsque ce sera le cas.

Dans le jeu de l'exemple 1, les utilités de la figure 1.3 se lisent de la manière suivante, pour l'action jointe $\omega = (S, T)$ l'utilité du joueur 1 sera -5 et l'utilité du joueur 2 sera 0. Dans la figure 1.4, les utilités se lisent de la manière suivante, pour l'action jointe $\omega = (S, T, S)$ les utilités des joueurs 1 et 3 seront -7 (condammés à 7 ans) et celle du joueur 2 sera -1 (condammé à 1 an).

1.2.2 Forme extensive

La forme extensive est utilisée pour représenter les jeux où les joueurs jouent de manière séquentielle dans un ordre déterminé. Un joueur a connaissance des stratégies jouées par les autres joueurs ayant joué avant lui. Ces jeux peuvent, entre autres, être utilisés pour modéliser les jeux répétés, jeux multi étape et jeux à information incomplète [Nisan, 2007, Osborne et Rubinstein, 1994] (voir section 3.1). Par exemple, le jeu de morpion mentionné précédemment est un jeu séquentiel que nous pouvons représenter sous une forme extensive.

La structure de représentation utilisée pour cette forme de jeu est un arbre orienté. Dans cet arbre, chaque noeud représente un état du jeu. Le jeu commence à la racine et se termine à une des feuilles, où se trouve l'utilité des joueurs obtenue lorsqu'ils ont tous terminé de jouer. Cette représentation permet de conserver l'historique du jeu (quelles actions sont jouées et quand elles le sont) pour chaque état possible (dans le cas d'un jeu à information complète). Pour chaque noeud, un seul joueur jouera une action. Chaque arête partant d'un noeud correspondra à une action possible qu'un joueur peut jouer. Lorsque l'on atteint une feuille de l'arbre, le jeu est terminé, l'ensemble des actions empruntées pour atteindre ce noeud vont correspondre à l'ensemble des

stratégies jouées par les joueurs. L'exemple 2 représente un jeu séquentiel en utilisant cette représentation.

Dans un jeu sous forme extensive, la stratégie d'un joueur est une fonction associant une action/arête à jouer pour chaque état possible non final qui peut être atteint.

Exemple 2 (Crise internationale)**.** *Deux pays sont en conflit sur la possession d'un territoire. L'un des pays a le choix entre réclamer ou pas le territoire. Si une réclamation est effectuée, le second pays a le choix entre accepter la réclamation ou se battre. Si aucune réclamation n'est faite, le second pays garde son territoire.*

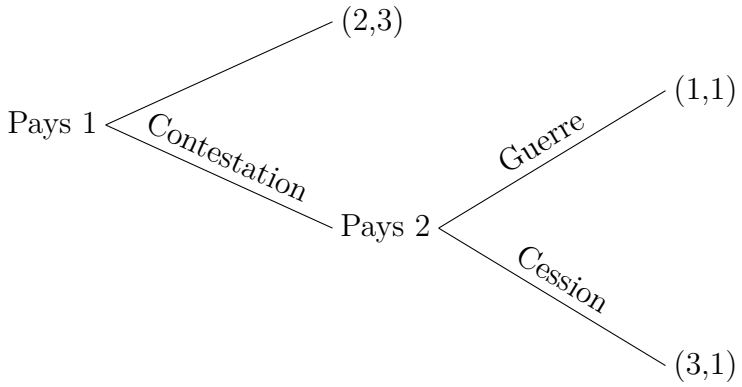


FIGURE 1.5 – Forme extensive du jeu de crise international

Un jeu sous forme extensive peut être ramené à un jeu sous forme normale de la manière suivante. Chaque feuille correspond à un ensemble de stratégies jouées, ce qui est équivalent à une stratégie jointe par feuille. Pour chaque noeud atteignable, l'ensemble d'arêtes partant de ce noeud correspondra à un ensemble d'actions qu'un joueur peut jouer, si un joueur joue plusieurs fois, le produit de ces ensembles d'actions correspondra à l'ensemble de ses actions possible pour le jeu équivalent sous forme normale. Dans le cas où l'arbre n'est pas équilibré, c'est a dire qu'il y a des feuilles à des niveaux différents de l'arbre, il est possible de considérer que les joueurs jouent tout de même une action pour chaque niveau possible indépendamment de si elle a une influence.

Nous pouvons reprendre la figure 1.5 de l'exemple 2 et la représenter sous la forme d'un arbre équilibre.

Exemple 3 (Crise internationale sous sa forme normale)**.** *En reprenant l'exemple précèdent, nous notons les actions du joueur 1 : C pour la contestation, NC pour l'absence de contestation et celles du joueur 2 : C est la cession et G est la guerre.*

L'arbre n'étant pas équilibré, nous allons quand même considérer que le pays 2 "choisit" une action lorsque le pays 1 choisit NC (elle ne sera pas jouée à proprement parler, car le jeu se "termine" après l'action NC) :

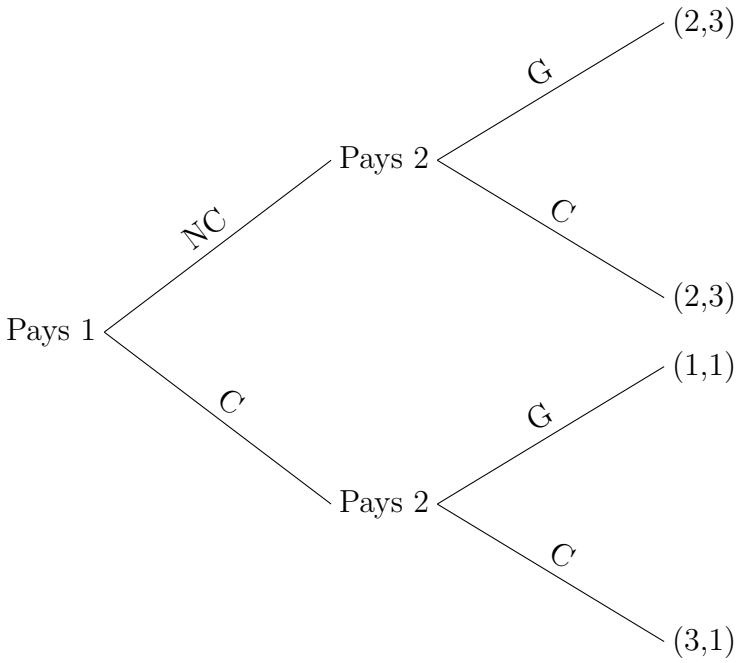


FIGURE 1.6 – Forme extensive du jeu de crise international non réduite

En prenant l'historique des actions jouées pour chacune des feuilles atteignables nous avons le jeu suivant

ω_1	ω_2	u_1	u_2
C	C	3	1
C	G	1	1
NC	C	2	3
NC	G	2	3

FIGURE 1.7 – Représentation du jeu sous forme normale

Cependant lorsque nous passons de la forme extensive à sa forme normale équivalente, il y a une perte d'information. Dans cet exemple, le fait que le joueur 2 ne "joue" pas lorsque le joueur 1 décide de ne pas faire contestation n'est plus visible. C'est-à-dire que le joueur 2 peut déclarer la guerre ou céder son territoire même si le joueur 1 ne fait aucune contestation, chose qu'il ne fera pas.

Dans le cas où un joueur joue plusieurs fois comme le jeu représenté dans la figure 1.8, si nous le transformons sous sa forme normale nous avons maintenant la table représentée à gauche dans la figure 1.9.

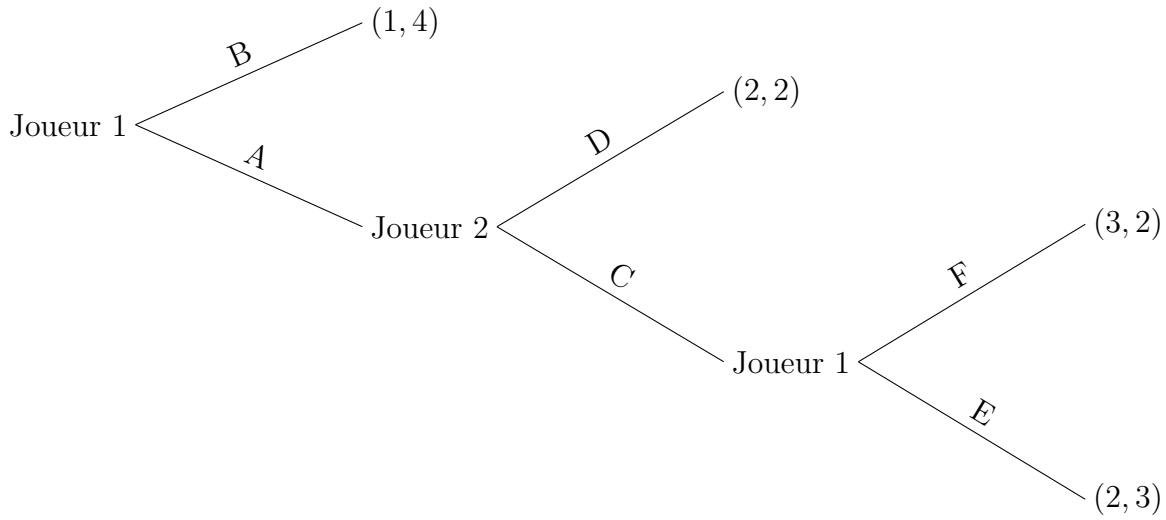


FIGURE 1.8 – Forme extensive du jeu où le joueur 1 joue 2 fois

Il est possible de représenter ce jeu séquentiel sous une forme normale simultanée (mais sans avoir une perte d'information comme mentionné précédemment).

En le représentant sous forme d'une table d'utilité nous avons les deux représentations possibles dans la figure 1.9 avec la table de droite étant une forme réduite.

ω_1	ω_2	u_1	u_2
AE	C	2	3
AE	D	2	2
AF	C	3	2
AF	D	2	2
BE	C	1	4
BE	D	1	4
BF	C	1	4
BF	D	1	4

ω_1	ω_2	u_1	u_2
AE	C	2	3
AE	D	2	2
AF	C	3	2
AF	D	2	2
B	C	1	4
B	D	1	4

FIGURE 1.9 – Représentation du jeu sous forme normale et sous la forme normale réduite

Lorsque le joueur 1 joue B, dans le jeu séquentiel une feuille sera atteinte, donc les autres actions que ce joueur peut jouer n'auront aucune influence, il est donc possible de considérer B comme une action sans tenir compte de F et E (sous la forme normale on peut voir que toutes les utilités du joueur 1 sont égales pour BE et BF).

1.2.3 Concepts de "solution" d'un jeu

En général, la fonction d'utilité d'un joueur capture les préférences associées à chaque issue possible du jeu. Un agent rationnel souhaitera maximiser son utilité tout en tenant compte des stratégies possibles des autres joueurs. Il existe différents concepts de solution en théorie des jeux, utilisés afin d'analyser le comportement des joueurs, des concepts tels que les stratégies dominées et dominantes, les meilleures réponses, les équilibres de Nash purs et mixtes, les équilibres corrélés. Les concepts

mentionnés dans le texte qui suit ne correspondent pas à la totalité de l'existant. Il existe une multitude d'autres concepts que nous décrirons rapidement comme les stratégies maximin, les stratégies Pareto optimale [Osborne et Rubinstein, 1994, Nisan, 2007].

Stratégie dominée, stratégie dominante

En prenant le point de vue d'un joueur, nous devons prendre une décision sans avoir connaissance du choix des autres joueurs. Ce qu'il est possible de faire est d'analyser le jeu afin de déterminer s'il n'y a pas des stratégies que nous ne devons pas choisir quelle que soit la situation. En tenant compte du fait qu'un joueur considérera que les autres joueurs sont, comme lui, rationnels, cette analyse ne se limitera pas seulement à ses stratégies, il s'intéressera aussi aux stratégies des autres joueurs.

Il est possible qu'un joueur ait une stratégie qui ne soit jamais la meilleure alternative possible (c'est-à-dire qu'il existe d'autre stratégies qui donneront toujours une meilleure utilité). Il est aussi possible qu'une stratégie soit toujours la meilleure alternative possible, c'est-à-dire qu'elle donnera toujours la meilleure utilité. Dans le premier cas on parlera de stratégie dominée et dans le second cas de stratégie dominante.

Les stratégies dominées peuvent être définies de la manière suivante :

Définition 2 (Stratégie faiblement dominée)**.** *Une stratégie ω_n est dite faiblement dominée s'il existe une stratégie ω'_n telle que*

$$u_n(\omega'_n, \omega_{-n}) \geqslant u_n(\omega_n, \omega_{-n}), \forall \omega_{-n} \in \Omega_{-n}$$

et qu'il existe ω_{-n} tel que

$$u_n(\omega'_n, \omega_{-n}) > u_n(\omega_n, \omega_{-n})$$

Définition 3 (Stratégie strictement dominée)**.** *Une stratégie ω_n est dite strictement dominée s'il existe une stratégie pure ω'_n tel que*

$$u_n(\omega'_n, \omega_{-n}) > u_n(\omega_n, \omega_{-n}), \forall \omega_{-n} \in \Omega_{-n}$$

Les stratégies dominantes peuvent être définies de la manière suivante :

Définition 4 (Stratégie faiblement dominante)**.** *Une stratégie ω_n est dite faiblement dominante si pour toute autre stratégie $\omega'_n \in \Omega_n$ nous avons :*

$$u_n(\omega_n, \omega_{-n}) \geqslant u_n(\omega'_n, \omega_{-n}), \forall \omega_{-n} \in \Omega_{-n}$$

et qu'il existe un ω_{-n} tel que

$$u_n(\omega_n, \omega_{-n}) > u_n(\omega'_n, \omega_{-n}),$$

Définition 5 (Stratégie strictement dominante)**.** *Une stratégie ω_n est dite strictement dominante si pour toute autre stratégie $\omega'_n \in \Omega_n$ nous avons :*

$$u_n(\omega_n, \omega_{-n}) > u_n(\omega'_n, \omega_{-n}), \forall \omega_{-n} \in \Omega_{-n}$$

Les stratégies dominées peuvent être utilisées afin de transformer un jeu G en un jeu G' dans lequel les stratégies (strictement et faiblement) dominées ont été retirées.

L'élimination répétée des stratégies dominées" (ERSD, en anglais "Iterated removal of dominated strategy",IRDA) permet d'effectuer cette réduction. Pour un jeu G, s'il existe une stratégie pour n'importe quel joueur qui est strictement dominée, cette stratégie est retirée de l'ensemble des actions disponibles pour le joueur auquel elle appartient

Exemple 4 (Élimination répétée des stratégies dominées)**.** *Considérons le jeu suivant :*

		Joueur 2		
		0	*1*	*2*
Joueur 1	*0*	*3,6*	*7,1*	*4,8*
	1	*5,1*	*8,2*	*6,1*
	2	*6,0*	*6,2*	*3,2*

La stratégie 0 *du joueur 1 est strictement dominée par sa stratégie* 1*. La stratégie* 0 *permet d'avoir comme utilité* 3*,* 7 *ou* 4*, la stratégie* 1 *la domine quelque soit la stratégie que le joueur* 2 *joue car il peut améliorer son utilité à* 5*,* 8 *ou* 6*. La stratégie* 0 *du joueur* 1 *peut être retirée car elle ne sera jamais la meilleure alternative.*

		Joueur 2		
		0	*1*	*2*
Joueur 1	*1*	*5,1*	*8,2*	*6,1*
	2	*6,0*	*6,2*	*3,2*

La stratégie 0 *du joueur* 2 *est strictement dominée par la stratégie* 1 *et la stratégie* 2 *du joueur* 2 *est faiblement dominée par la stratégie* 1 *du joueur* 2*. Lorsque la stratégie* 0 *est jouée les utilités du joueur* 2 *seront soit* 1 *soit* 0*, elles peuvent être améliorées à* 2 *et* 2 *par sa stratégie* 1*. La stratégie* 2 *du joueur* 2 *peut aussi être améliorée par la stratégie* 1 *lorsque le joueur* 1 *joue la stratégie* 1*, lorsque ce dernier joue la stratégie* 2 *il n'y a pas d'amélioration possible mais pas de diminution de l'utilité qui restera à* 2*. La stratégie* 0 *peut être retirée puisqu'elle est strictement dominée par la stratégie* 1*.*

		Joueur 2	
		1	*2*
Joueur 1	*1*	*8,2*	*6,1*
	2	*6,2*	*3,2*

La stratégie 2 *du joueur* 1 *est strictement dominée par la stratégie* 1*. Elle permet d'avoir une utilité de* 6 *ou* 3 *qui peut être améliorée à* 8 *ou* 6 *par la stratégie 1. Elle est donc retirée.*

		Joueur 2	
		1	*2*
Joueur 1	*1*	*8,2*	*6,1*

La stratégie 2 *du joueur* 2 *est strictement dominée par la stratégie* 1*, elle permet d'avoir une utilité de* 1 *alors que la stratégie* 1 *permet d'avoir une utilité de* 2*, elle est retirée.*

		Joueur 2
		1
Joueur 1	*1*	*8,2*

Nous avons donc obtenu un jeu à 2 joueurs avec une action par joueur.

Pour ce jeu, jouer la stratégie jointe $(1,1)$ *est une situation souhaitable pour les deux joueurs. Aucun des deux joueurs ne pourra choisir une alternative lui donnant une meilleure utilité.*

La procédure d'élimination des stratégies dominées peut converger vers un jeu où chaque joueur a une seule stratégie comme dans l'exemple, cependant elle peut également s'arrêter avant. Il peut exister plusieurs ordres d'élimination différents. Mais tous conduiront au même jeu réduit.

Meilleure Réponse

Lorsque nous étudions les stratégies des joueurs nous allons parler de "meilleure réponse". Une meilleure réponse est la stratégie d'un joueur permettant d'obtenir la meilleure utilité possible pour une stratégie jointe donnée des autres joueurs (en partant du principe que les autres joueurs sont statiques).

Considérons une stratégie ω, un joueur n et son utilité $u_n(\omega)$. En changeant la stratégie en ω', on suppose que les joueurs autres que n gardent leurs stratégies ω_{-n}. On considérera ω'_n comme une amélioration si $u_n(\omega'_n, \omega_{-n}) > u_n(\omega)$.

Définition 6 (Meilleure réponse)**.** *Une stratégie* ω_n *sera la meilleure réponse à une stratégie* ω_{-n} *si elle maximise l'utilité du joueur* n *:*

$$BR_n(\omega_{-n}) = \{\omega_n \in \Omega_n, u_n(\omega_n, \omega_{-n}) = max_{\omega'_n \in \Omega_n} u_n(\omega'_n, \omega_{-n})\}$$

Exemple 5 (Reprise de l'exemple 1)**.** *En reprenant l'exemple du dilemme du prisonnier, lorsque la stratégie jointe* (S,S) *est jouée, la meilleure réponse des deux prisonniers à l'action* S *(joué par l'autre joueur) est l'action* T*. C'est a dire que dans* (S,S)*, aucun des joueurs ne jouent leur meilleur réponse. Pour l'action* (S,S) *nous avons les utilités* $(-1,-1)$*. Donc si le joueur 1 choisit l'action* T *à la place il aura comme utilité* 0 *ce qui est une amélioration (la même chose se produit pour le joueur 2 s'il change son action) donc c'est une meilleure réponse.*

Dans certains jeux, cette notion permet d'atteindre ce que nous allons appeler un équilibre de Nash pur.

Équilibre de Nash Pur

Nous allons dire qu'un joueur n joue une stratégie pure lorsqu'il joue une de ses stratégie $\omega_n \in \Omega_n$. Par extension nous disons qu'une stratégie jointe pure ω est jouée lorsque tous les joueurs jouent une stratégie pure.

Un équilibre de Nash est un concept de solution qui correspond à un état stable dans un jeu lorsque les joueurs agissent de manière rationnelle. Dans un équilibre de Nash, aucun des joueurs ne peut changer sa stratégie de manière unilatérale et améliorer son utilité. Cela signifie que compte tenu des stratégies jouées par les autres joueurs, chaque joueur va avoir maximisé son utilité pour la situation donnée. En d'autres termes, un équilibre de Nash pur est une stratégie jointe où chaque joueur joue une meilleure réponse aux stratégies des autres joueurs

Définition 7 (Équilibre de Nash pur)**.** *Une stratégie jointe* ω^* *est un équilibre de Nash pur si et seulement si*

$$u_n(\omega^*) \geqslant u_n(\omega_n, \omega^*_{-n}), \forall \omega_n \in \Omega_n, n \in S$$

Alternativement, en utilisant la définition de meilleure réponse on peut aussi caractériser un équilibre de Nash ω^* par :

$$\omega_n^* \in BR_n(\omega_{-n}^*), \forall n \in S$$

Exemple 6 (Reprise de l'exemple 1)**.** *En reprenant le dilemme du prisonnier de l'exemple 1, dans ce jeu, la stratégie pure $\omega = (T, T)$ est un équilibre de Nash pur. L'utilité des joueurs pour cette stratégie est $(-3, -3)$ si le joueur 1 change de stratégie, son utilité diminuera à -5 ce qui n'est pas une amélioration donc il n'a pas d'intérêt à changer de stratégie. C'est la même situation pour le joueur 2. Aucun joueur ne peut changer de stratégie et améliorer son utilité par conséquent la stratégie est bien un équilibre de Nash pur. La stratégie (S, S) n'est pas un équilibre de Nash pur car chaque joueur a -1 comme utilité cependant si l'un des joueurs décide de trahir l'autre joueur en changeant son action en T, tandis que l'autre joueur reste silencieux (ne change pas d'action), nous obtenons la stratégie jointe (S, T) ou (T, S) dans laquelle le joueur jouant T va passer son utilité de -1 à 0 ce qui est une amélioration.*

ERSD est lié au concept de "meilleure réponse", puisque ERSD "élimine" les stratégies dominées. Lorsque l'utilisation de ERSD amène à un jeu avec une stratégie unique, comme dans l'exemple 4, cette stratégie jointe correspondra à un équilibre de Nash pur.

Cependant il peut y avoir un équilibre de Nash pur dans un jeu où ERSD n'élimine aucune stratégie.

L'existence d'un équilibre de Nash pur n'est pas garanti dans tous les jeux. Par exemple dans le jeu "Pierre, feuille, ciseau", il n'y a pas d'équilibre pur.

Exemple 7 (Pierre, feuille, ciseau)**.** *Si nous représentons ce jeu par une table d'utilité nous avons la table suivante :*

		Joueur 2		
		P	*F*	*C*
Joueur 1	*P*	*0,0*	*-1,1*	*1,-1*
	F	*1,-1*	*0,0*	*-1,1*
	C	*-1,1*	*1,-1*	*0,0*

Si les deux joueurs jouent pierre (P, P), l'un des deux peut changer sa stratégie en feuille pour améliorer son utilité. Supposons que le joueur 1 fasse ceci, nous sommes maintenant dans (F, P), le joueur 2 peut changer sa stratégie en ciseau et améliorer son utilité. Maintenant pour stratégie (F, C), le joueur 1 peut changer sa stratégie en pierre est améliorer son utilité, nous sommes donc maintenant dans (P, C). Nous pouvons continuer ceci indéfiniment car nous nous trouvons dans un cycle tel que : $(F, P) \to (F, C) \to (P, C) \to (P, F) \to (C, F) \to (C, P) \to (F, P)$

Lorsque les deux joueurs jouent la même action ((F, F), (P, P) et (C, C)) les deux joueurs ont une meilleure réponse qui amènera à une stratégie jointe appartenant au cycle mentionné précédemment.

Stratégies maximin

Pour un joueur, jouer une stratégie maximin, parfois appelée "secure strategy", revient à jouer une action pour laquelle il obtiendra la meilleure utilité possible pour les pires conditions possibles, c'est à dire quel que soit le choix des autres joueurs.

Définition 8 (Stratégie maximin). *Une stratégie ω_n d'un joueur n est une stratégie maximin si et seulement si*

$$min_{\omega_{-n}} u_n(\omega_n, \omega_{-n}) \geqslant max_{\omega'_n} min_{\omega_{-n}} u_n(\omega'_n, \omega_{-n})$$

Exemple 8 (Stratégie maximin). *En reprenant le jeu utilisé dans l'exemple 4*

		Joueur 2		
		0	*1*	*2*
Joueur 1	*0*	*3,6*	*7,1*	*4,8*
	1	*5,1*	*8,2*	*6,1*
	2	*6,0*	*6,2*	*3,2*

La stratégie maximin du joueur 1 *est la stratégie* 1. *Les minimums des utilités que ce joueur peut obtenir pour chacune de ses stratégies possibles sont* 3, 5 *et* 3 *pour respectivement les actions* 0, 1 *et* 2. *Le maximum de ces minimums étant* 5, *c'est donc la stratégie* 1 *qui sera la stratégie maximin du joueur* 1

Les stratégies maximin du joueur 2 *sont les stratégies* 1 *et* 2. *Les minimums des utilités que ce joueur peut obtenir pour chacune de ses stratégies possibles sont* 0, 1 *et* 1 *pour respectivement les actions* 0, 1 *et* 2. *Le maximum de ces minimums étant* 1, *les stratégies* 1 *et* 2 *sont les stratégies maximin du joueur* 2

Équilibre de Nash mixte

La notion de stratégie mixte consiste à choisir les actions jouées de manière aléatoire selon une distribution de probabilité choisie au préalable.

Définition 9 (Stratégie mixte). *Une stratégie mixte jointe est un ensemble de distributions de probabilité (une par joueur) $\xi = (\xi_1, ..., \xi_N)$ avec ξ_n une distribution de probabilité sur les actions du joueur n, telle que :*

$$\xi_n : \Omega_n \to [0, 1] \ et \sum_{\omega_n \in \Omega_n} \xi_n(\omega_n) = 1$$

Si la distribution de probabilité vérifie $\exists \omega_n \in \Omega_n, \xi_n(\omega_n) = 1, \forall n \in S$ alors la stratégie est pure (ces dernières sont, au final, des cas particuliers de stratégie mixte)

Par extension nous pouvons aussi écrire $\xi(\omega) = \prod_{\nu \in S} \xi_\nu(\omega_\nu)$ la probabilité que l'action jointe ω soit jouée lorsque la stratégie mixte ξ est utilisée.

Compte tenu qu'une stratégie mixte est une distribution de probabilité, les utilités seront des utilités espérées.

Définition 10 (Utilité espérée). *Lorsque nous avons une stratégie mixte ξ, l'utilité espérée d'un joueur n est définie par :*

$$EU_n(\xi) = \sum_{\omega \in \Omega} u_n(\omega) * \prod_{\nu \in S} \xi_\nu(\omega_\nu)$$

où $\xi_\nu(\omega_\nu)$ est la probabilité que le joueur ν joue l'action ω_ν lorsqu'il utilise la stratégie mixte ξ_ν.

Il est possible d'étendre les notions de stratégie dominée et dominante aux stratégies mixtes.

Notez que dans un jeu sous forme extensive, la notion de stratégie mixte existe.

Une stratégie mixte peut être interprétée de différentes manières. On peut considérer qu'une stratégie revient à dire qu'un joueur ajoute une notion d'aléatoire dans sa prise de décision. Par exemple, les joueurs de poker peuvent choisir aléatoirement de bluffer, un gouvernement d'effectuer un contrôle fiscal aléatoirement...

Une autre interprétation possible est de considérer que la stratégie mixte représente une croyance. C'est-à-dire que les joueurs ont une croyance commune sur les stratégies de chacun des joueurs. Plusieurs interprétations possibles sont décrite dans [Osborne et Rubinstein, 1994].

La notion d'équilibre de Nash peut être reformulée : une stratégie mixte sera un équilibre de Nash mixte s'il n'est pas possible de changer la stratégie mixte d'un joueur et d'améliorer son utilité. Tous les jeux en forme normale admettent au moins un équilibre de Nash mixte (qui peut être pur) [Nash, 1950].

Définition 11 (Équilibre de Nash mixte)**.** *Une stratégie ξ^* est un équilibre de Nash mixte d'un jeu $\langle S, \Omega, u\rangle$ si et seulement si :*

$$EU_n(\xi_n^*.\xi_{-n}^*) \geqslant EU_n(\xi_n.\xi_{-n}^*), n \in S, \forall \xi_n$$

Il est possible de vérifier si une stratégie mixte est bien un équilibre en utilisant les stratégies pures de chacun des joueurs.

Proposition 1 (Vérification en utilisant des stratégie pures)**.** *Une stratégie mixte ξ^* est un équilibre de Nash mixte si et seulement si pour toutes les stratégies pures ω_n de tous les joueurs n nous avons :*

$$EU_n(\xi_n^*.\xi_{-n}^*) \geqslant EU_n(\omega_n.\xi_{-n}^*), n \in S, \forall \omega_n \in \Omega_n$$

Preuve (Proposition 1)**.** *Voir [Myerson, 1991, Osborne et Rubinstein, 1994].*

Exemple 9 (Pierre, Feuille Ciseau)**.** *En reprenant le "Pierre, feuille, ciseau" de l'exemple 7, comme mentionné précédemment, il n'y a pas d'équilibre pur, cependant il existe au moins un équilibre de Nash mixte pour tout jeu comme démontré par Nash [Nash, 1950]. Ici, la stratégie mixte $\xi = \{(P : \frac{1}{3}, F : \frac{1}{3}, C\frac{1}{3}), (P : \frac{1}{3}, F : \frac{1}{3}, C : \frac{1}{3})\}$ est un équilibre de Nash mixte :*

$$EU_1(\xi) = EU_2(\xi) = \frac{1}{9}*0+\frac{1}{9}*-1+\frac{1}{9}*1+\frac{1}{9}*1+\frac{1}{9}*0+\frac{1}{9}*-1+\frac{1}{9}*-1+\frac{1}{9}*1+\frac{1}{9}*0 = 0$$

Si l'on change la stratégie mixte de l'un des joueurs en l'une des 3 stratégies pures possibles (sans changer la stratégie mixte du second joueur) nous avons :

- $EU_1(P, \xi_2) = EU_2(\xi_1, P) = 0$
- $EU_1(F, \xi_2) = EU_2(\xi_1, F) = 0$
- $EU_1(C, \xi_3) = EU_2(\xi_1, C) = 0$

Il n'y a aucune alternative qui permet à l'un des deux joueurs d'améliorer son utilité, elle sera toujours à 0 donc la stratégie est bien un équilibre de Nash mixte.

Les équilibres de Nash mixtes sont un concept qui occupe une place très importante en algorithmique des jeux. Déterminer l'existence d'un équilibre pur et calculer un équilibre de Nash mixte sont des problèmes très souvent abordés que nous verrons plus en détail dans la section 1.3.

Nous pouvons nous demander quelle est l'utilité de cette notion au delà du fait qu'elle est présente dans tous les jeux.

Il y a deux interprétations possibles des équilibres de Nash mixte, une qui est "prescriptive" l'autre qui est "descriptive".

Une interprétation "prescriptive" peut être vue comme une situation où les stratégies mixtes sont interprétées comme les modes de décision des joueurs. Chacun des joueurs choisit son action selon la distribution de probabilité calculée, comme un lancé de dés, car selon cette distribution ils obtiendront une utilité qui sera "en moyenne" la plus satisfaisante possible. Cette stratégie peut être choisie par les agents après concertation ou bien par un acteur externe au jeu.

Une interprétation "descriptive" peut être de voir l'équilibre comme si nous avions un point de vue extérieur au jeu qui nous permet de déterminer que, si les joueurs devaient jouer plusieurs fois à des jeux représentant la même situation, nous aurions une fréquence des actions jouées qui correspondrait aux stratégies mixtes calculées.

Cependant, notez qu'un équilibre de Nash (mixte comme pur) est une stratégie "stable" mais n'est pas nécessairement le meilleur "choix" possible de stratégie que les joueurs peuvent faire. Par exemple dans le dilemme du prisonnier, si l'on regarde la peine de prison de chaque joueur ainsi que la somme totale de leurs peines, il est plus logique que les deux joueurs restent silencieux (pour une peine d'un an chacun). Cependant l'équilibre de Nash pur de ce jeu demande qu'ils se trahissent mutuellement, amenant la peine à trois ans pour chacun d'entre eux. L'équilibre amène à un état stable, cela signifie qu'aucun des joueurs n'a d'intérêt à dévier du choix effectué.

Stratégie Pareto optimale

De manière similaire à la notion de stratégie dominée, on peut considérer qu'une stratégie jointe pure est dominée par une autre lorsque cette dernière permet d'augmenter l'utilité d'un joueur sans diminuer celle des autres, nous allons dire qu'une stratégie jointe est Pareto dominée dans ce cas là. De manière similaire, nous dirons qu'une stratégie jointe Pareto domine une autre si elle permet d'améliorer l'utilité d'au moins un joueur sans diminuer celle des autres joueurs.

Définition 12 (Pareto dominante)**.** *Une stratégie ω Pareto domine une stratégie ω' si et seulement si :*

$$\exists n \in S, u_n(\omega) > u_n(\omega') et\ \forall \nu \in S, u_\nu(\omega) \geqslant u_\nu(\omega')$$

On dira qu'une stratégie jointe pure est Pareto optimale si aucun des joueurs ne peut améliorer son utilité sans diminuer celles des autres joueurs, c'est à dire :

Définition 13 (Stratégie Pareto optimale)**.** *Une stratégie jointe mixte ω est Pareto optimale si et seulement si il n'existe pas de stratégie ω' qui la Pareto domine.*

Exemple 10 (Dilemme du prisonnier : Stratégie Pareto optimale)**.** *En reprenant le dilemme du prisonnier de l'exemple 1, il est possible d'identifier trois stratégies Pareto optimales. Les stratégies (S,S),(T,S) et (S,T) sont toutes les trois Pareto optimales avec comme utilités respectives $(-1,-1)$,$(0,-5)$ et $(-5,0)$. Par exemple, lorsque (S,S) est jouée les deux joueurs peuvent tout deux améliorer leur utilité avec la stratégie (T,S) pour le joueur 1 et (S,T) pour le joueur 2, l'utilité d'un joueur augmentera à 2 cependant l'utilité de l'autre joueur diminuera à -5 donc (S,S) n'est pas Pareto dominée par ces stratégies. La stratégie (T,T) est Pareto dominée par (S,S) car les utilités des joueurs sont toutes inférieure à -1. La stratégie (S,S) n'est Pareto dominée par aucune des autres stratégies.*

Concernant les stratégies (T, S) et (S, T), elles ne sont pas Pareto dominées l'une par l'autre car même si l'utilité d'un des joueurs est augmentée de -5 à 0 celle de l'autre joueur passe de 0 à -5. La stratégie (S, S) n'en domine aucun des deux car si l'un des joueur augmente son utilité de -5 à -1 l'autre la diminue de 0 à -1. La même chose se produit pour (T, T) où l'utilité d'un joueur passera de -5 à -3 mais celle de l'autre diminuera de 0 à -3.

Notez que l'équilibre de Nash pur (T, T) n'est pas Pareto optimal car il est Pareto dominé par la stratégie jointe (S, S).

Équilibre corrélé

La notion d'équilibre corrélé est plus générale qu'un équilibre de Nash mixte.

Elle consiste à considérer que les joueurs ne prennent plus de décision via une stratégie mixte individuelle mais selon une distribution de probabilité sur les stratégies jointes de tous les joueurs choisie au préalable par un coordinateur qui n'est pas un acteur "dans" le jeu. On peut aussi considérer ceci comme une situation où les joueurs choisissent de jouer selon une distribution de probabilité sur les stratégies jointes qui est choisie après une négociation entre les différents joueurs.

Selon cette distribution de probabilité, le coordinateur informe les joueurs sur l'action à jouer. Les joueurs peuvent choisir de ne pas obéir au coordinateur. Si un joueur ne peut pas changer unilatéralement de stratégie et améliorer son utilité alors la stratégie du coordinateur est un équilibre corrélé [Aumann, 1974, Aumann, 1987]. Cela signifie que le joueur n'aura aucun intérêt à désobéir au coordinateur, à la stratégie choisie après négociation.

Définition 14 (Équilibre corrélé [Osborne et Rubinstein, 1994])**.** *Une distribution de probabilité sur les actions jointes $\bar{\xi}$ est un équilibre corrélé si pour toute fonction de transformation $\phi_n : \Omega_n \to \Omega_n$ et pour tous les joueurs nous avons :*

$$\sum_{\omega \in \Omega} \bar{\xi}(\omega) u_n(\omega_n, \omega_{-n}) \geqslant \sum_{\omega \in \Omega} \bar{\xi}(\omega) u_n(\phi_n(\omega_n), \omega_{-n}) \forall n \in S$$

Propriété 1 (Définition alternative [Nisan, 2007])**.** *La définition précédente est équivalente à :*

$$\sum_{\omega_{-n} \in \Omega_{-n}} \bar{\xi}(\omega) u_n(\omega_n, \omega_{-n}) \geqslant \sum_{\omega_{-n} \in \Omega_{-n}} \bar{\xi}(\omega) u_n(\omega'_n, \omega_{-n}) \forall \omega'_n \in \Omega_n, n \in S$$

Un équilibre de Nash est un équilibre corrélé qui remplit une condition supplémentaire, à savoir que la distribution de probabilité de l'équilibre corrélé correspond à un produit de stratégies mixtes (de chaque joueur).

Exemple 11 (Céder le passage)**.** *Exemple repris de [Papadimitriou et Roughgarden, 2008] En prenant un jeu dans lequel deux joueurs se croisent en voiture à une intersection, avec les utilités suivantes, où S correspond à un arrêt et G à continuer son chemin.*

		Joueur 2	
		S	G
Joueur 1	S	4,4	1,5
	G	5,1	0,0

FIGURE 1.10 – Jeu de céder le passage

Dans ce jeu, les stratégies $\xi = \{(1,0)(0,1)\}$, $\xi = \{(0,1)(1,0)\}$ *et* $\xi = \{(\frac{1}{2},\frac{1}{2})(\frac{1}{2},\frac{1}{2})\}$ *sont des équilibres de Nash (purs pour les deux premiers et mixte pour le dernier). Ces stratégies correspondent aussi à des équilibres corrélés. Les distributions de probabilité sur les stratégies jointes* $\bar{\xi}$ *équivalente à ces stratégies mixtes vont correspondre aux distributions de probabilité représentés dans la figure 1.12 où les distributions sont calculées comme dans la figure 1.11*

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	$\bar{\xi}(SS) = \xi_1(S) \times \xi_2(S)$	$\bar{\xi}(SG) = \xi_1(S) \times \xi_2(G)$
	G	$\bar{\xi}(GS) = \xi_1(G) \times \xi_2(S)$	$\bar{\xi}(GG) = \xi_1(G) \times \xi_2(G)$

FIGURE 1.11 – Représentation du calcul des distributions de probabilité

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	0	1
	G	0	0

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	0	0
	G	1	0

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	0.25	0.25
	G	0.25	0.25

FIGURE 1.12 – Distributions de probabilité sur les actions jointe correspondant à des équilibres de Nash *et* corrélé

Ces stratégies ont comme utilité espérée $(1,5)$, $(5,1)$ *et* $(2.5,2.5)$.

Il existe aussi 2 équilibres corrélés qui ne sont pas des équilibres de Nash (il n'y a pas de stratégie mixte permettant d'obtenir les distributions de probabilité), qui sont les suivants :

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	0	0.5
	G	0.5	0

		Joueur 2 : S	Joueur 2 : G
Joueur 1	S	$\frac{1}{3}$	$\frac{1}{3}$
	G	$\frac{1}{3}$	0

FIGURE 1.13 – Deux équilibres corrélés pour le jeux "Céder le passage"

Ces stratégies ont des utilités espérées $(3,3)$ *et* $(\frac{10}{3},\frac{10}{3})$

Supposons que le joueur 1 *décide de ne pas "obéir" à la stratégie choisie par le coordinateur et de toujours jouer la stratégie* G. *Nous avons donc l'utilité espérée de l'équilibre corrélé qui est :*

$$\bar{\xi}(S,S) * u_1(S,S) + \bar{\xi}(S,G) * u_1(S,G) + \bar{\xi}(G,S) * u_1(G,S) + \bar{\xi}(G,G) * u_1(G,G)$$

Que l'on comparera à l'utilité espéré lorsque nous utilisons ϕ :

$$\bar{\xi}(S,S)*u_1(\phi_1(S),S)+\bar{\xi}(S,G)*u_1(\phi_1(S),G)+\bar{\xi}(G,S)*u_1(\phi_1(G),S)+\bar{\xi}(\phi_1(G),G)*u_1(G,G)$$

Par exemple, pour la distribution $(0,0.5,0.5,0)$, *si le joueur* 1 *décide de dévier de la stratégie, nous avons :*

Pour $\phi_1(S) = G, \phi_1(G) = G$

$$0 * 0 + 0.5 * 1 + 0.5 * 5 + 0 * 0 \geqslant 0 * 5 + 0.5 * 0 + 0.5 * 5 + 0 * 0$$
$$3 \geqslant 2.5$$

Maintenant supposons que le joueur 1 *décide de toujours jouer la stratégie* S*. C'est-à-dire, pour* $\phi_1(S) = S, \phi_1(G) = S$

$$0 * 0 + 0.5 * 1 + 0.5 * 5 + 0 * 0 \geqslant 0 * 4 + 0.5 * 1 + 0.5 * 4 + 0 * 1$$
$$3 \geqslant 2.5$$

Supposons maintenant que le joueur 1 *joue l'inverse de ce qu'il est indiqué par la stratégie donnée. Pour* $\phi_1(S) = G, \phi_1(G) = S$

$$0 * 0 + 0.5 * 1 + 0.5 * 5 + 0 * 0 \geqslant 0 * 5 + 0.5 * 0 + 0.5 * 4 + 0 * 1$$
$$3 \geqslant 2$$

Aucune des "déviations" possible du joueur 1 *ne lui permet d'améliorer son utilité. Les utilités étant symétriques, c'est aussi le cas pour le joueur* 2 *avec cette stratégie.*

Il y a 5 équilibres corrélés, avec 3 d'entre eux correspondant aussi à des équilibres de Nash classique.

Dans la suite de la thèse, nous allons nous limiter aux jeux compétitifs simultanés et à la recherche d'équilibre de Nash mixte.

1.3 Complexité du calcul d'un équilibre de Nash mixte sous forme normale

Nous avons vu précédemment la notion d'équilibre de Nash et rappelé que tout jeu admet au moins un équilibre de Nash mixte. Étant donné que nous avons connaissance de l'existence de cette solution, nous pouvons nous demander si trouver cette solution est quelque chose de faisable "facilement", quels sont les algorithmes et méthodes efficaces permettant de trouver cette solution ?

La réponse à cette question dépend aussi de "l'applicabilité" de la notion d'un équilibre de Nash dans des problèmes de grande taille. S'il n'est pas possible de le calculer raisonnablement facilement, alors pour des situations réelles (comme la modélisation d'un problème de part de marché), il est difficile de supposer qu'il soit possible de l'utiliser dans l'analyse d'un jeu s'il n'est pas possible pour une machine de le calculer...

Dans cette section, nous allons nous intéresser aux questions de complexité du calcul des équilibres de Nash mixtes dans les jeux en forme normale. Dans la section suivante, nous allons lister quelques-uns des algorithmes existants permettant de calculer un équilibre de Nash mixte dans les jeux sous forme normale.

Le problème de calcul d'un équilibre de Nash mixte dans un jeu sous forme normale n'appartient sans doute pas à la classe de complexité P. Aucun algorithme polynomial n'a été trouvé pour calculer un équilibre de Nash. Par conséquent et de manière similaire à d'autres problèmes, nous pourrions supposer que ce problème de calcul est NP-difficile. Cependant, ce n'est pas non plus le cas pour ce problème. Comme mentionné précédemment, pour tout jeu, l'existence d'un équilibre de Nash mixte est garanti. Alors qu'un problème comme SAT (satisfaisabilité booléenne) sera NP-complet car il n'y a pas la certitude de l'existence d'une solution à ce problème, pour une instance quelconque. SAT est un problème de *décision* : nous souhaitons prouver s'il existe ou pas une solution. Le problème de calcul d'un équilibre de Nash mixte (que l'on désignera par *problème NASH*) est quant à lui un (total) "function problem",

nous souhaitons trouver une solution à ce problème, non pas vérifier l'existence d'une solution car son existence est garantie, contrairement à un problème de décision.

Pour le problème de calcul d'un équilibre de Nash mixte, il est donc nécessaire de définir une classe de complexité pour les problèmes difficiles à résoudre similaire à ce dernier (où le problème admet l'existence d'au moins une solution). Cette classe de complexité s'appele PPAD (Polynomial Parity Arguments on Directed graphs), c'est une sous-classe de TFNP (total function nondeterministic polynomial) [Meggido et Papadimitriou, 1989] qui est elle-même une sous-classe de FNP (function nondeterministic polynomial) [Meggido et Papadimitriou, 1989].

1.3.1 Classe TFNP

Avant de définir la classe PPAD, nous allons faire un retour rapide sur les "function problems" et les "function problems" admettant au moins une solution. Un "function problem" est un problème algorithmique où pour toute entrée possible une sortie est attendue sous une forme qui peut être différente de "Vrai" ou "Faux", à la différence d'un problème de décision.

Les problèmes qui sont NP sont des problèmes de décision. La classe équivalente pour les "function problems" est la classe FNP (Function Nondeterministic Polynomial), elle généralise la classe NP. Cependant dans cette classe, toute instance du problème n'admet pas forcément une solution.

En 1989, la classe TFNP (Total Function Nondeterministic Polynomial) a été définie [Meggido et Papadimitriou, 1989], cette classe regroupe les "function problems" qui admettent toujours une solution et pour lesquels la vérification de cette solution se fait en temps polynomial. Les problèmes appartenant à cette classe peuvent être résolus en temps polynomial sur une machine non déterministe.

Cependant, en général, lorsqu'il s'agit d'étudier la complexité de problèmes appartenant à TFNP, ces études se font via les sous classes de TFNP.

Ces sous-classes comprennent notamment PPA, PLS, PPP et PPAD :

— PPA (Polynomial Time Parity Argument) [Papadimitriou, 1994] : Cette classe regroupe les problèmes pour lesquels il est possible de démontrer qu'il existe toujours une solution en utilisant le lemme des poignées de main, à savoir que pour un graphe non orienté fini, il y a un nombre pair de noeuds de degré impair.

— PLS (Polynomial Local Search) [Johnson *et al.*, 1988] : Cette classe regroupe les problèmes pour lesquels il est possible de démontrer qu'il existe une solution par la démonstration que pour un graphe dirigé acyclique il existe au moins un puits.

— PPP (Polynomial Time Pigeonhole Principle) [Papadimitriou, 1994] : Pour cette classe, l'existence d'une solution peut être démontrée par l'utilisation du principe des tiroirs, pour un circuit avec n entrées et $n-1$ sorties, il existe deux variables différentes ayant la même sortie (principe de collision).

Nous allons décrire la classe PPAD plus en détails que les autres car elle a originellement été définie spécifiquement pour le *problème NASH*.

1.3.2 La classe PPAD

La classe de complexité PPAD (Polynomial Parity Arguments on Directed graphs) est une classe de complexité dans laquelle se trouve le problème de recherche d'un

équilibre de Nash et pour laquelle *NASH* est PPAD-complet [Papadimitriou, 1994]. Un "function problem" appartient à la classe PPAD s'il est possible de le réduire à un problème *End of line* [Papadimitriou, 1994].

Définition 15 (End of line)**.** *Un graphe dirigé G est spécifié de manière succincte. Il a les propriétés suivantes :*

- *Chaque noeud a au plus un arc sortant et un arc entrant*
- *Pour un noeud, il est possible de calculer en temps polynomial le noeud suivant et le noeud précédent*
- *Il existe une source nommée standard source, c'est à dire un noeud pour lequel il n'y a pas d'arc entrant*
- *Il existe au moins un puits : un noeud pour lequel il n'y a pas d'arc sortant*

Nous cherchons pour chaque entrée possible à avoir un puits en sortie

Pour une entrée constituée d'un graphe et d'un noeud déséquilibré de ce graphe (source ou puits), la sortie (solution du problème *end of line*) sera un autre noeud déséquilibré (puits). Si le problème est posé avec une liste de noeuds et d'arêtes, il est tout à fait possible de le résoudre en temps polynomial, cependant nous partons du principe que le problème est trop grand pour être représenté de cette façon et il est plutôt représenté sous la forme d'un programme qui calcule en temps polynomial pour un noeud et un arc entrant (sauf la source standard), quel est l'arc sortant.

Étant donné qu'un noeud a au plus un arc sortant et au plus un arc entrant, le graphe sera un ensemble de chemins et de circuits. Si nous identifions un noeud qui n'a pas d'arc entrant (la source standard), ce noeud appartiendra à un chemin menant à une solution (un puits). Donc le problème "End of line" a toujours une solution.

1.3.3 Complexité du problème de calcul d'un équilibre de Nash

Il a été démontré dans [Daskalakis *et al.*, 2009] que le *problème NASH* est PPAD complet.

Le *problème NASH* est dans PPAD car il est possible faire une réduction en temps polynomial de ce problème sous la forme du problème *End of Line* présenté précédemment [Papadimitriou, 1994]. Cependant la démonstration de PPAD complétude du problème n'a été effectuée que plus tard.

La PPAD complétude de *NASH* a été démontrée progressivement dans les années 2000. Pour les jeux avec 4 joueurs [Goldberg et Papadimitriou, 2006], 3 joueurs [Daskalakis et Papadimitriou, 2005, Chen et Deng, 2005] et 2 joueurs [Chen et Deng, 2006].

Dans le cas à 2 joueurs, la démonstration se base en partie sur l'appartenance du problème à PPAD-difficile et à PPAD via l'utilisation de l'algorithme de Lemke Howson [Lemke et Howson, 1964, Papadimitriou, 1994].

1.4 Résolution des jeux sous forme normale

Comme mentionné précédemment, il existe plusieurs concepts de solution dans les jeux. Par conséquent il existe différentes méthodes et algorithmes pour les trouver. Nous allons principalement nous intéresser aux méthodes et algorithmes utilisés pour trouver des équilibres de Nash mixte.

Les jeux sous forme normale étant la forme la plus basique de représentation des jeux, nous trouvons une multitude d'algorithmes différents permettant de calculer des

équilibres de Nash mixte, ces algorithmes utilisent des méthodes très différentes les une des autres et renvoient des solutions qui peuvent être exactes ou approchées.

Nous pouvons catégoriser les différentes approches existantes :

1. Les méthodes "historiques" à base de parcours de chemin et de résolution de systèmes d'équations [Lemke et Howson, 1964, Wilson, 1971]. Étant donné que nos travaux du chapitre 5 s'inspirent de ces méthodes, nous les décrirons en détail. Lemke Howson dans ce chapitre et notre ré-interprétation de Wilson dans le chapitre 5.
2. D'autres méthodes à base de systèmes d'équations polynomiales : [Porter *et al.*, 2008, Lipton et Markakis, 2004, Datta, 2010]. Ces méthodes peuvent être exactes ou approchées, suivant que l'on utilise un solveur exact ou approché
3. Méthodes numériques, à base d'optimisation [van der Laan *et al.*, 1987, Sandholm *et al.*, 2005] et d'utilisation des propriétés d'homotopie [Govindan et Wilson, 2003, Govindan et Wilson, 2004].

1.4.1 Méthodes de parcours de chemin

Algorithme de Lemke Howson

Le premier algorithme développé pour trouver un équilibre de Nash mixte exact dans les jeux bimatriciels est l'algorithme de Lemke Howson [Lemke et Howson, 1964, Lemke, 1965, Cottle et Dantzig, 1967]. C'est un algorithme dont le principe général consiste à parcourir un chemin jusqu'à trouver une solution. Cet algorithme permet de converger vers un équilibre de Nash mixte. L'algorithme de Lemke Howson transforme tout d'abord le problème de calcul d'un équilibre de Nash mixte dans un jeu bimatriciel en un problème de complémentarité linéaire (LCP).

Définition 16 (LCP)**.** *Un problème de complémentarité linéaire (Linear Complementarity Problem : LCP) est un problème de programmation linéaire où pour une matrice M et un vecteur colonne q nous devons trouver les vecteurs colonnes x et y tel que :*

$$\begin{cases} x & \geqslant & 0 \\ y & \geqslant & 0 \\ y & = & Mx + q \\ x^T.y & = & 0 \end{cases}$$

Proposition 2. *Un jeu bimatriciel peut être réduit vers un LCP[Lemke et Howson, 1964]*

La solution d'un LCP, un vecteur faisable et complémentaire, correspond à un équilibre de Nash mixte dans le jeu dont le LCP est issu.

Il y a plusieurs manières d'implémenter et d'expliquer l'algorithme de Lemke Howson et la transformation d'un jeu sous la forme d'un LCP [McKelvey et McLennan, 1996, Von Stengel, 2002]. L'explication que nous allons donner va se baser sur une résolution et formulation utilisant des matrices.

Lorsque nous formulons le jeu sous la forme d'un LCP, la matrice M sera créée à partir des utilités des deux joueurs et q sera un vecteur colonne dont tous les éléments valent 1.

Nous avons un jeu bimatriciel sous forme normale $G = \langle S, \Omega, u \rangle$ où $S = \{1, 2\}$ et $\Omega_n = \{\omega_1, ..., \omega_{m_n}\}$ signifiant que le joueur n à m_n stratégies. Pour une table d'utilité

u_n, nous construisons une matrice d'utilité U^n avec m_n lignes et m_ν colonnes où $\nu \neq n$ tel que l'élément (i, j) de cette matrice correspondra à l'utilité du joueur n lorsque la stratégie i (correspondant l'indice d'une stratégie $\omega_n \in \Omega_n$) est jouée avec la stratégie j du joueur ν (correspondant à l'indice d'une stratégie $\omega_\nu \in \Omega_\nu$), cette valeur correspond donc à $u_n(\omega_n, \omega_\nu) = U^n_{i,j}$ et $u_\nu(\omega_n, \omega_\nu) = U^\nu_{j,i}$.

Dans un jeu bimatriciel pour des matrices d'utilité $U^1 \in \mathbb{Q}^{m_1 \times m_2}$ et $U^2 \in \mathbb{Q}^{m_2 \times m_1}$ et une stratégie mixte sous forme d'un vecteur $\xi = (\xi_1, \xi_2)$, un équilibre de Nash $\xi^* = (\xi_1^*, \xi_2^*)$ vérifie les équations suivantes :

$$\begin{aligned} \xi_1^{*T} U^1 \xi_2^* &\geqslant \xi_1^T U^1 \xi_2^*, \forall \xi_1 \\ \xi_2^{*T} U^2 \xi_1^* &\geqslant \xi_2^T U^2 \xi_1^*, \forall \xi_2 \end{aligned} \tag{1.1}$$

En définissant des vecteurs colonne de 1, e_1 et e_2 qui ont respectivement m_1 et m_2 éléments.

$$\begin{aligned} (\xi_1^{*T} U^1 \xi_2^*) e_1 &\geqslant U^1 \xi_2^* \\ (\xi_2^{*T} U^2 \xi_1^*) e_2 &\geqslant U^2 \xi_1^* \end{aligned} \tag{1.2}$$

On peut réduire ceci sous la forme d'un LCP

$$\begin{cases} x'_1, x'_2 & \geqslant & 0 \\ y'_1, y'_2 & \geqslant & 0 \\ y'_1 & = & U^1 x'_2 - e_1 \quad x'_1, y'_1 \in \mathbb{Q}^{m_1} \\ y'_2 & = & U^2 x'_1 - e_2 \quad x'_2, y'_2 \in \mathbb{Q}^{m_2} \\ x'^T_1 . y'_1 & = & 0 \\ x'^T_2 . y'_2 & = & 0 \end{cases}$$

Avec :

$$M = \begin{bmatrix} \mathbf{0} & U^1 \\ U^2 & \mathbf{0} \end{bmatrix} \; x = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \; y = \begin{bmatrix} y'_1 \\ y'_2 \end{bmatrix} \; q = \begin{bmatrix} -e_1 \\ -e_2 \end{bmatrix}$$

Les variables y' sont des variables d'écart et les variables x' correspondent à une distribution de probabilité non normalisée sur les stratégies des joueurs.

Les variables x' du LCP correspondent à des stratégies mixtes non normalisées où x'_n correspond à celle du joueur n. Toutes les variables x'_{n,ω_n} sont des variables positives ou nulles (pour des questions de notation nous considérons ici que les stratégies ω_n sont des entiers, les indices de la stratégie $1 \leqslant \omega_n \leqslant |\Omega_n|$) Les distributions de probabilités correspondant aux stratégies mixtes ξ sont obtenues en normalisant ces variables x. $\xi_n(\omega_n) = \frac{x'_{n,\omega_n}}{\sum_{i \in \Omega_n} x'_{n,i}}$ permet de définir la distribution de probabilité sur Ω_n, une stratégie mixte du joueur n. Les variables y' correspondent aux "variables d'écart" utilisées dans le LCP, elles sont propres au LCP.

Nous avons les vecteurs $x = (x'_1, x'_2)$ et $y = (y'_1, y'_2)$ qui sont tous les deux de taille $m = m_1 + m_2$. La solution d'un LCP est complémentaire et faisable. La condition de complémentarité $x^T y = 0$ est respectée pour l'ensemble des variables, ce qui veut dire que $x_i y_i = 0$, pour $1 \leqslant i \leqslant m$ (la solution est complémentaire) et les conditions $x \geqslant 0$ et $y \geqslant 0$ sont respectées (la solution est faisable).

Les solutions complémentaires et faisables du LCP sont soit un équilibre de Nash, soit une "extraneous solution" (parfois appelée solution externe, solution superflue, racine étrangère...). Une solution superflue est une solution du LCP où $x = 0$ et $y = q$, cette solution du LCP ne correspond pas à un équilibre de Nash étant donné qu'on ne peut normaliser x.

Exemple 12 (Jeu bimatriciel en LCP)**.** *Pour un jeu à 2 joueurs où le joueur* 1 *a 3 actions/stratégie pures ($m_1 = 3$) et le joueur* 2 *en a* 2 *($m_2 = 2$), ils ont respectivement une matrice d'utilité U_1 et U_2 telles que*

$$U^1 = \begin{bmatrix} 0 & 6 \\ 2 & 5 \\ 3 & 3 \end{bmatrix}, U^2 = \begin{bmatrix} 1 & 0 & 4 \\ 0 & 2 & 3 \end{bmatrix}.$$

En reprenant les vecteurs et matrices présentées précédemment nous avons donc :

$$U = \begin{bmatrix} 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 2 & 5 \\ 0 & 0 & 0 & 3 & 3 \\ 1 & 0 & 4 & 0 & 0 \\ 0 & 2 & 3 & 0 & 0 \end{bmatrix} x = \begin{bmatrix} x'_{1,1} \\ x'_{1,2} \\ x'_{1,3} \\ x'_{2,1} \\ x'_{2,2} \end{bmatrix} y = \begin{bmatrix} y'_{1,1} \\ y'_{1,2} \\ y'_{1,3} \\ y'_{2,1} \\ y'_{2,2} \end{bmatrix} q = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Principe de l'algorithme : Pour obtenir un équilibre de Nash, il faut donc obtenir une solution complémentaire faisable. A partir d'une solution complémentaire faisable (la solution externe) une succession de pivots sera effectuée afin de passer par une suite de solutions faisables presque-complémentaires jusqu'à obtenir une nouvelle solution complémentaire faisable. Cette solution correspond à un équilibre de Nash. Cet algorithme est très lié au problème *End of line* (il a d'ailleurs inspiré ce problème définissant la classe de complexité PPAD).

Base : Une solution est associée à une base, une base est un ensemble de variables x_i correspondant à une solution faisable. Pour des questions de simplification de notation et d'explication, l'ensemble des variables d'une base sera un ensemble de variables x_i et y_i.

Si une variable $x_i > 0$ ou $y_i > 0$ alors elle est dans la base, si une variable $x_i = 0$ ou $y_i = 0$ alors elle est hors base.

Pivot : Un pivot fait référence à l'échange d'une variable hors de la base avec une variable dans la base. La représentation dans le cas linéaire pouvant se faire sous forme d'une matrice à deux dimensions, l'utilisation du terme pivot vient de la méthode de manipulation de matrice. Cela revient donc à faire en sorte que l'une des équations $x_i = 0$ ou $y_i = 0$ devienne une inéquation et que l'une des inéquations $x_j > 0$ ou $y_j > 0$ devienne une équation.

Presque-complémentaire : Une solution est presque-complémentaire lorsque la condition de complémentarité n'est pas respectée par, au plus, un couple (x_i, y_i), c'est-à-dire que pour une solution presque-complémentaire, les variables x_i et y_i sont toutes deux dans la base (donc $x_i > 0$ et $y_i > 0$).

L'algorithme de Lemke-Howson est correct car il a été démontré que, pour un jeu *non-dégénéré* (détaillé dans le chapitre 5), un unique pivot est possible à chaque étape. Un chemin, partant de la solution superflue est donc parcouru. Comme le nombre de noeuds du chemin est fini (chacun correspond à un sous-ensemble de variables de base et hors base), l'algorithme se termine lorsque l'autre extrémité du chemin, qui est une solution complémentaire, est atteinte.

Il y a plusieurs méthodes applicables pour faire fonctionner l'algorithme en pratique (géométrique avec des labels, matricielle), l'explication suivante s'inspire du fonctionnement dans le cas matriciel en tentant de rester à un niveau "intuitif" (en reprenant l'approche présentée dans [McKelvey et McLennan, 1996]).

Étape 1 : Choisir une solution complémentaire et faisable comme point de départ. En général, la solution superflue est choisie donc $x_i = 0, \forall x_i \in x$, ainsi toutes les variables x_i sont hors base et toutes les variables y_i sont dans la base. On passe à l'étape 2.

Étape 2 : Choisir une des variables x_i qui va entrer dans la base. Lorsque l'on part de la solution externe, cette variable peut être n'importe quelle variable $x_i \in x$ étant donné que toute les variables x sont hors base. Un pivot est effectué avec la variable permettant de maintenir la solution faisable. Le pivot va consister à faire augmenter la valeur de la variable que l'on veut faire entrer dans la base jusqu'à ce qu'une des variables actuellement dans la base devienne nulle (qu'elle sorte de la base par conséquent). Passer à l'étape 3.

Étape 3 : S'il y a un couple $(x_i, y_i) \in B$ (dans la base) alors la solution de la base B est presque-complémentaire, nous passons à l'étape 4.

Si il n'existe pas de couple $(x_i, y_i) \in B$ alors la solution (x, y) de la base B est complémentaire et x correspond à une solution du jeu (un équilibre mixte non normalisé).

Étape 4 : Pour la dernière variable qui est sortie de la base lors du dernier pivot, l'autre variable du couple (x_i, y_i) est la prochaine variable à entrer dans la base. Par exemple, si la variable y_i a été sortie de la base lors du dernier pivot, la variable x_i entrera dans la base. Un pivot est effectué avec la variable permettant de maintenir la solution faisable. Si une variable x_i entre de la base elle augmente jusqu'à ce qu'une des variables x_j ou y_j qui est influencée par sa valeur devienne nulle (avec x_j ou y_j dans la base). Le même principe s'applique pour les variables y_i Nous retournons à l'étape 3.

Après avoir trouvé une solution faisable et complémentaire, il est possible de calculer la stratégie mixte équivalente (dans le cas présenté, normaliser les variables de x est suffisant).

Exemple 13 (Exemple informelle de l'exécution de l'algorithme)**.** *En reprenant l'exemple décrit précédemment, nous avons les matrices et vecteurs suivant :*

$$-U = \begin{bmatrix} 0 & 0 & 0 & 0 & -6 \\ 0 & 0 & 0 & -2 & -5 \\ 0 & 0 & 0 & -3 & -3 \\ -1 & 0 & -4 & 0 & 0 \\ 0 & -2 & -3 & 0 & 0 \end{bmatrix} \quad q = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Donc avec $y = -Ux + q$ et $y \geqslant 0$ nous avons les équations suivantes :

$$\begin{array}{c} \\ y_{1,1} \\ y_{1,2} \\ y_{1,3} \\ y_{2,1} \\ y_{2,2} \end{array} \begin{array}{c} \begin{array}{ccccc} x_{1,1} & x_{1,2} & x_{1,3} & x_{2,1} & x_{2,2} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & -6 \\ 0 & 0 & 0 & -2 & -5 \\ 0 & 0 & 0 & -3 & -3 \\ -1 & 0 & -4 & 0 & 0 \\ 0 & -2 & -3 & 0 & 0 \end{bmatrix} \end{array} + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \geqslant \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Ce qui donne les équations suivantes pour les y :

$$\begin{aligned} y_{1,1} &= -6x_{2,2} + 1 \geqslant 0 \\ y_{1,2} &= -2x_{2,1} - 5x_{2,2} + 1 \geqslant 0 \\ y_{1,3} &= -3x_{2,1} - 3x_{2,2} + 1 \geqslant 0 \\ y_{2,1} &= -x_{1,1} - 4x_{1,3} + 1 \geqslant 0 \\ y_{2,2} &= -2x_{1,2} - 3x_{1,3} + 1 \geqslant 0 \end{aligned}$$

Nous partons de la solution externe, qui correspond à $x = 0, y = q$, qui se vérifie facilement avec le système d'équation. Nous allons noter b l'ensemble des variables dans la base, qui est actuellement $b = (y_{1,1}, y_{1,2}, y_{1,3}, y_{2,1}, y_{2,2})$. On choisit donc une variable à entrer dans la base, étant donné que nous sommes à la solution externe (qui est complémentaire), nous avons le choix entre toutes les variables hors de la base, à savoir toutes les variables de x. Nous choisissons par exemple de faire entrer la variable $x_{2,2}$. Pour faire entrer cette variable, nous allons augmenter sa valeur jusqu'à qu'une des variables actuellement dans la base devienne nulle.

$$\begin{aligned} y_{1,1} &= -6x_{2,2} + 1 \geqslant 0 \\ y_{1,2} &= -5x_{2,2} + 1 \geqslant 0 \\ y_{1,3} &= -3x_{2,2} + 1 \geqslant 0 \end{aligned} \tag{1.3}$$

Ici la première variable à devenir nulle lorsque $x_{2,2}$ augmente est $y_{1,1}$ avec $x_{2,2} = \frac{1}{6}$. La nouvelle base est donc $b = (x_{2,2}, y_{1,2}, y_{1,3}, y_{2,1}, y_{2,2})$ avec $x = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{6} \end{bmatrix}$ et $y = \begin{bmatrix} 0 & \frac{1}{6} & \frac{1}{2} & 1 & 1 \end{bmatrix}$

La variable $y_{1,1}$ venant de sortir de la base, nous allons faire entrer la variable correspondante $x_{1,1}$.

$$\begin{aligned} y_{2,1} &= -x_{1,1} + 1 \geqslant 0 \\ y_{2,2} &= 1 \end{aligned}$$

Ici étant donné que seule $y_{2,1}$ est influencée par $x_{1,1}$, ce sera la variable qui va devenir nulle lorsque $x_{1,1} = 1$ La nouvelle base est $b = (x_{1,1}x_{2,2}, y_{1,2}, y_{1,3}, y_{2,2})$ avec $x = \begin{bmatrix} 1 & 0 & 0 & 0 & \frac{1}{6} \end{bmatrix}$ et $y = \begin{bmatrix} 0 & \frac{1}{6} & \frac{1}{2} & 0 & 1 \end{bmatrix}$

La variable $x_{2,1}$ est la prochaine à entrer dans la base :

$$\begin{aligned} y_{1,1} &= -6x_{2,2} + 1 \geqslant 0 \\ y_{1,2} &= -2x_{2,1} - 5x_{2,2} + 1 \geqslant 0 \\ y_{1,3} &= -3x_{2,1} - 3x_{2,2} + 1 \geqslant 0 \end{aligned} \tag{1.4}$$

Ici il est possible de trouver la valeur de $x_{2,1}$ facilement car dans le cas actuel, il n'y a pas besoin de changer la valeur de $x_{2,2}$ (nous serions contraints en temps normal de la diminuer). Lorsque $x_{2,1} = \frac{1}{12}$ la valeur de $y_{1,2}$ devient nulle donc cette dernière sort de la base. Noter que dans certain cas il est possible que "l'augmentation" d'une variable n'annule pas une variable y mais une variable x.

La nouvelle base est $b = (x_{1,1}, x_{2,1}, x_{2,2}, y_{1,3}, y_{2,2})$ avec $x = \begin{bmatrix} 1 & 0 & 0 & \frac{1}{12} & \frac{1}{6} \end{bmatrix}$ et $y = \begin{bmatrix} 0 & 0 & \frac{3}{4} & 0 & 1 \end{bmatrix}$

La prochaine variable à entrer dans la base est $x_{1,2}$

$$\begin{aligned} y_{2,1} &= -x_{1,1} + 1 \geqslant 0 \\ y_{2,2} &= -2x_{1,2} + 1 \geqslant 0 \end{aligned}$$

Ici la valeur de $x_{1,1}$ *ne changera pas car* $x_{1,2}$ *est seulement dans le calcul de* $y_{2,2}$ *qui sera la variable qui deviendra nulle lorsque* $x_{1,2} = \frac{1}{2}$. *La nouvelle base est* $b = (x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}, y_{1,3})$ *avec* $x = \begin{bmatrix} 1 & \frac{1}{2} & 0 & \frac{1}{12} & \frac{1}{6} \end{bmatrix}$ *et* $y = \begin{bmatrix} 0 & 0 & \frac{3}{4} & 0 & 0 \end{bmatrix}$. *Il n'y a pas de couple* $(x_{n,i}, y_{n,i})$ *dans b donc la solution est complémentaire.*

Le vecteur $x = \begin{bmatrix} 1 & \frac{1}{2} & 0 & \frac{1}{12} & \frac{1}{6} \end{bmatrix}$ *correspond aux deux vecteurs* $x_1 = \begin{bmatrix} 1 & \frac{1}{2} & 0 \end{bmatrix}, x_2 = \begin{bmatrix} \frac{1}{12} & \frac{1}{6} \end{bmatrix}$ *qui après normalisation nous donnent les stratégies mixtes suivantes :* $\xi_1 = (\frac{2}{3}, \frac{1}{3}, 0), \xi_2 = (\frac{1}{3}, \frac{2}{3})$

Une version matricielle de l'algorithme est présentée en dans l'annexe A.

Algorithme de Wilson

En 1971, Wilson a étendu théoriquement l'algorithme de Lemke Howson aux jeux avec plus de 2 joueurs (jeux à N joueurs). Il a démontré qu'il était possible de définir le problème de calcul d'un équilibre de Nash mixte dans un jeu à N joueurs sous la forme d'un système d'équations et d'inéquations, qui sera plus tard appelé problème de complémentarité polynomiale (PCP) [Gowda, 2017]. La méthode définie théoriquement par Wilson consiste à effectuer un parcours de chemin en effectuant des pivots entre points presque complémentaires. L'une des difficultés, par rapport au cas à 2 joueurs, est qu'un pivot nécessite de résoudre un système d'équations polynomiales, au lieu de linéaires.

L'algorithme de Wilson n'est qu'"esquissé", et présenté de manière théorique, comme une succession de parcours d'arcs définis eux aussi par des systèmes d'équations polynomiales.

Dans cette thèse j'en propose une description originale, algébrique et combinatoire, et j'en propose la première implémentation (chapitre 5).

Simplicial subdivision

La méthode "simplicial subdivision" définie par [van der Laan *et al.*, 1987] est basée sur une approximation de l'espace des stratégies jointes mixtes par un "squelette" formé de sommets et d'arêtes (rectilignes). Un algorithme de parcours de chemin (comme Lemke Howson) est appliqué sur ce squelette, et converge vers un équilibre de Nash approché. Plus la grille est fine, plus on s'approche d'un équilibre de Nash (et nous convergeons vers un équilibre de Nash si l'espace entre les points de la grille tend vers 0). Notez que le nombre de points augmente exponentiellement quand l'espace entre les points se réduit.

1.4.2 Méthode à base d'équations polynomiales

Il existe de nombreux algorithmes de calcul d'équilibres de Nash. Certains utilisables seulement pour des jeux ayant des propriétés particulières, nous allons seulement en mentionner quelques uns.

Énumération de supports L'énumération de support, définie par [Porter *et al.*, 2008] est une méthode permettant de calculer un équilibre de Nash mixte exact dans un jeu avec N joueurs. Cette méthode va consister à énumérer chacun des supports de stratégie possibles dans un ordre de taille croissante en commençant par les stratégies pures. Le support d'une stratégie mixte d'un joueur est l'ensemble de ses stratégies pures de probabilités non nulles. Pour chaque support, on vérifie tout d'abord qu'il n'existe pas de stratégie dominée si l'on retire les stratégies qui ne seront jamais jouées. S'il existe une stratégie dominée, cela signifie que le jeu équivalent a déjà été exploré par un support précédemment évalué. Le support actuel est ignoré. S'il n'existe pas de stratégie dominée alors pour le support actuel nous vérifions s'il n'existe pas de stratégie mixte correspondant à un équilibre de Nash. Lorsqu'un équilibre est trouvé, l'algorithme est arrêté, cependant, il est possible d'utiliser cette méthode pour énumérer tous les équilibres de Nash d'un jeu.

Nous reparlerons plus en détails de l'énumération de support dans la section 5.3 du chapitre 5.

Nous expliquons ici brièvement comment les supports sont énumérés dans le cas d'un jeu à 3 joueurs (et 2 actions chacun).

Exemple 14 (Énumération des supports)**.** *Pour un jeu avec 3 joueurs où chaque joueur a 2 actions, les premiers supports énumérés seront ceux où tous les joueurs jouent une seule de leur stratégie (3 stratégies sont jouées), donc toutes les stratégies jointes pures possibles, à savoir :*

ω_1	ω_2	ω_3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Ensuite, nous énumérons les supports où un joueur joue deux de ses stratégies pures (probabilités non nulles) et les deux autres en jouent une seule (au total 4 *stratégies sont jouées).*

ω_1	ω_2	ω_3
$(0,1)$	0	0
$(0,1)$	0	1
$(0,1)$	1	0
$(0,1)$	1	1
0	$(0,1)$	0
0	$(0,1)$	1
1	$(0,1)$	0
1	$(0,1)$	1
0	0	$(0,1)$
0	1	$(0,1)$
1	0	$(0,1)$
1	1	$(0,1)$

S'il n'y a toujours aucun équilibre trouvé nous passons aux supports où il y a 5 actions jouées. Ici cela signifie que deux joueurs jouent deux de leurs stratégies pures tandis que le troisième joueur en joue une seule, comme représenté dans le tableau suivant :

ω_1	ω_2	ω_3
$(0,1)$	$(0,1)$	0
$(0,1)$	$(0,1)$	1
$(0,1)$	0	$(0,1)$
$(0,1)$	1	$(0,1)$
0	$(0,1)$	$(0,1)$
1	$(0,1)$	$(0,1)$

Notez que si nous avions 3 actions par joueurs il y aurait d'autres possibilité qui seraient énumérée après celles décrites précédemment car il y a des supports possibles qui seraient moins équilibrés. Les nouveaux cas possibles sont qu'un joueur joue trois actions et que les deux autres en jouent une seule. Ces supports plus déséquilibré sont examinées après les support équilibrés.

Supposant que nous n'avons toujours rien trouvé, nous passons maintenant à un support de taille 6. Chaque joueur jouera ses deux actions, ce qui donnera forcement une stratégie mixte correspondant à un équilibre de Nash mixte.

ω_1	ω_2	ω_3
$(0,1)$	$(0,1)$	$(0,1)$

Équations Polynomiales Il existe des algorithmes utilisant des systèmes d'équations et d'inéquations polynomiales pour calculer un équilibre de Nash approché [Lipton et Markakis, 2004], l'équilibre de Nash correspondra à la solution d'un système d'équations polynomiales. La méthode proposée par [Datta, 2010] propose de formuler le problème sous la forme de système d'équations polynomiales qui seront résolues en utilisant les bases de Gröbner et une méthode homotopique.

1.4.3 Méthode à base d'optimisation et d'homotopie

Principe de l'homotopie : A partir d'un problème de départ, nous souhaitons trouver $f(\vec{x}) = \vec{0}$. Nous prenons un problème "facile" à résoudre avec $g(\vec{x}) = \vec{0}$ tel qu'une solution $\vec{x}^*$ est évidente à trouver. L'homotopie va consister à résoudre $h_\alpha(\vec{x}) = \vec{0}$ pour un coefficient $\alpha \in [0,1]$ où $h_\alpha(\vec{x}) = (1-\alpha)g(\vec{x}) + \alpha f(\vec{x})$ donc $h_0(\vec{x}) = g(\vec{x})$ et $h_1(\vec{x}) = f(\vec{x})$. L'idée va être d'augmenter progressivement α en partant de 0 vers 1. Cela nous donnera un "arc" de solutions intermédiaires $\{\vec{x}^*_\alpha\}_{\alpha\in[0,1]}$. Le but est de parcourir cet arc, et calculer $\vec{x}^*_{\alpha+d\alpha}$ à partir de $\vec{x}^*_\alpha$, par recherche locale. Le problème est que l'arc défini par l'homotopie peut être "discontinu", tous les $h_\alpha(\vec{x}) = \vec{0}$ ont une solution, mais s'il y a un $\vec{x}^*_{\alpha+d\alpha}$ qui est très différent de $\vec{x}^*_\alpha$ alors la recherche locale ne fonctionne pas.

Homotopie Il existe plusieurs méthode utilisant l'homotopie [Herings et van den Elzen, 2002, Govindan et Wilson, 2003, Govindan et Wilson, 2004, Herings et Peeters, 2010].

La méthode Global Newton permet de calculer un équilibre de Nash mixte [Govindan et Wilson, 2003]. Cette méthode utilise l'homotopie. Elle consiste à prendre un

problème ne pouvant pas être résolu en l'état et à le déformer en un problème facile à résoudre pour ensuite passer progressivement par une séquence de problèmes se rapprochant de plus en plus du problème d'origine. Dans le cas d'un jeu à N joueurs ceci reviendra à reformuler le jeu sous la forme d'un jeu plus facile a résoudre et trouver l'équilibre de Nash de ce jeu facile. La recherche de l'équilibre est formulée sous la forme d'un système d'équations et d'inéquations.

La méthode proposée, par [Govindan et Wilson, 2004] est liée à celle présentée dans [Govindan et Wilson, 2003]. Comme la méthode précédente, elle va utiliser l'homotopie, la différence va intervenir sur les jeux à résoudre. Comme pour l'autre algorithme il va s'agir de résoudre itérativement des jeux simples en se rapprochant progressivement du jeu original à résoudre. Cependant un jeu original sera "approché" sous la forme d'un jeu polymatriciel (les jeux polymatriciels seront présentés dans le chapitre 2), permettant d'utiliser les propriétés des opérations de Lemke Howson. Cette méthode peut être utilisée conjointement avec la méthode global Newton, qui est plus fiable, pour effectuer un "démarrage" plus rapide de l'algorithme.

Optimisation/Minimisation de fonction [Sandholm *et al.*, 2005] montrent une formulation possible du problème de recherche d'un équilibre de Nash mixte des jeux bimatriciels sous la forme d'un MIP(mixed interger program), un programme linéaire où certaines des variables doivent être des entiers.

[Boryczka et Juszczuk, 2013] s'intéressent à l'utilisation d'un algorithme à évolution différentielle pour calculer un équilibre de Nash approché dans un jeu à plusieurs joueurs.

Il est proposé par [Berg et Sandholm, 2017] un algorithme permettant de calculer un équilibre de Nash mixte approché dans les jeux à N joueurs. Cette méthode consistera à utiliser un arbre de recherche pour lequel l'espace de recherche sera décomposé en un ensemble de "régions" qui seront explorées jusqu'à ce qu'un équilibre approché soit trouvé. Chaque région correspond à un ensemble de stratégies mixtes définies selon une fonction. En utilisant un oracle, nous déterminons s'il est impossible qu'un équilibre se trouve dans une région donnée. S'il est impossible de trouver un équilibre, la région est exclue, s'il est possible d'en trouver un la région est re-divisée en un nouvel ensemble de régions, division qui peut se faire de différentes façons.

1.5 Conclusion

Ce chapitre a présenté différents types de jeux et de concepts associés à ces jeux ; les jeux à information complète avec leur forme normale et leur forme extensive. Différents concepts de solutions utilisées pour résoudre les jeux ont été présentés, les stratégies dominées, les équilibres de Nash purs et mixtes, les équilibres corrélés...

Dans le chapitre suivant, des représentations compactes des jeux à information complète seront présentées, les jeux succincts.

Chapitre 2

Jeux Succincts

2.1 Introduction

L'une des difficultés de modélisation en théorie des jeux intervient lorsqu'il s'agit de représenter des jeux impliquant un grand nombre de joueurs. La représentation la plus basique, les jeux sous forme normale, demande le stockage d'une table multidimensionnelle des utilités des joueurs associées à chacune des stratégies jointes possibles. L'espace nécessaire pour représenter le jeu augmentera de manière exponentielle en fonction du nombre de joueurs. Plus la taille d'un jeu augmente, plus il est difficile de la représenter en un espace raisonnable et, par conséquent, plus il devient difficile de chercher des solutions (comme un équilibre de Nash mixte).

Par exemple, la complexité du problème de recherche d'un équilibre pur est linéaire en la taille de représentation donc exponentielle en le nombre de joueurs. Pour le problème de recherche d'un équilibre de Nash mixte, celle-ci est exponentielle en la taille de représentation du jeu.

Les jeux sous forme normale partent du principe qu'il y a des interactions entre tous les joueurs. Cependant, lorsque l'on modélise des situations réelles, cette hypothèse n'est pas toujours vérifiée. L'utilité d'un joueur peut dépendre seulement d'un sous-ensemble des autres joueurs. Il est important de pouvoir représenter ces jeux de manière compacte, en se basant sur des modèles graphiques notamment.

Comme mentionné dans [Kearns *et al.*, 2001], une représentation par graphe peut permettre de modéliser les interactions locales entre les joueurs. C'est le cas lorsque les agents ont des interactions dépendantes de caractéristiques physiques, par exemple des problèmes où les profits des vendeurs dépendent de la présence d'autres vendeurs dans des régions environnantes. Ou bien des relations au niveau d'une organisation telle que dans une entreprise où des employés dépendent de superviseurs, qui eux-mêmes dépendent de managers, en grimpant les échelons jusqu'au PDG. Ou encore la modélisation d'un réseau informatique où le choix de chaque joueur a une influence sur l'utilité de ses voisins, comme lorsqu'il s'agit d'effectuer de la répartition de charge entre plusieurs ordinateurs/serveurs.

Exemple 15 (Road Game)**.** *Prenons un exemple inspiré du jeu Road game [Vickrey et Koller, 2002]. Chaque joueur possède un terrain le long d'une route et souhaite choisir quelque chose à construire sur ce terrain. On suppose ici qu'il a le choix entre 2 actions, construire un magasin ou une résidence. La satisfaction d'un joueur dépend de :*

- *Son choix.*
- *Le choix des joueurs sur les terrains adjacents.*

— *Le choix du joueur sur le terrain en face de lui.*

La figure 2.1 représente un exemple d'un Road game à 6 joueurs.

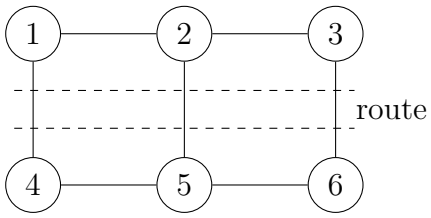


FIGURE 2.1 – Road Game

Dans ce jeu, l'utilité d'un joueur dépend des stratégies d'au plus trois autres joueurs. La forme normale équivalente demanderait 6 *tables avec* 2^6 *(*$|\Omega_n|^N$*) entrées, ce qui n'est pas encore très coûteux cependant si nous prenions un Road Game avec une centaine d'agents la représentation sous forme normale ne serait plus envisageable (100 tables avec* 2^{100} *entrées).*

Il existe plusieurs types de jeux succincts permettant de représenter de manière compacte un jeu tel que celui présenté dans l'exemple 15. Les jeux polymatriciels, graphiques et hypergraphiques sont des jeux succincts utilisant un modèle graphique pour représenter les interactions entre les différents joueurs.

2.2 Jeux graphiques

Les jeux graphiques ont été conçus afin de représenter succinctement les jeux où l'utilité d'un joueur dépend seulement d'un sous-ensemble des stratégies des autres joueurs [Kearns *et al.*, 2001].

Définition 17 (Jeu graphique)**.** *Un jeu graphique est caractérisé par un tuple* $GG = \langle S, \Omega, E, u \rangle$ *où*

- $S = \{1, ..., N\}$ *est l'ensemble des joueurs participant au jeu.*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des stratégies jointes.* Ω_n *est l'ensemble des stratégies du joueur* n*.*
- $E = \{e_1, .., e_N\}$ *est un ensemble de* N *sous-ensembles de* S*.* e_n *est un ensemble contenant* n *et représente l'ensemble des joueurs dont les stratégies influencent l'utilité du joueur* n*.*
- $u = (u_n : \Omega_{e_n} \rightarrow \mathbb{R})_{n \in S}$ *est l'ensemble des* N *tables d'utilité (de dimension strictement inférieure à* N*, en général).* u_n *est l'utilité du joueur* n*, dépendant des stratégies de ses voisins*

L'ensemble des joueurs jouant avec le joueur n (mais excluant ce dernier) se note S_n. L'utilité d'un joueur n dépendra de l'ensemble d'action jointe Ω_{e_n} tel que $\Omega_{e_n} = \Omega_n \times \Omega_{S_n}$.

Exemple 16 (Road Game)**.** *Comme mentionné précédemment, dans un Road game un joueur aura au plus 3 voisins donc* $max_{n \in S}|e_n| = 4$*. Avec* m *le nombre d'actions de chaque joueur, la taille de représentation du jeu sera* $N \times m^4$ *au lieu de* $N \times m^N$ *pour une représentation sous forme normale.*

— $N = 6$ *avec* $S = \{1, 2, 3, 4, 5, 6\}$ *l'ensemble des joueurs*

— $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3 \times \Omega_4 \times \Omega_5 \times \Omega_6$ *avec* $\Omega_n = \{M, R\}$ *(où* M *correspond à la construction d'un magasin et* R *d'une résidence)*

— $E = \{e_1 = (1,2,4), e_2 = (1,2,3,5), e_3 = (2,3,6), e_4 = (1,4,5), e_5 = (2,4,5,6), e_6 = (3,5,6)\}$

— $u = (u_1, u_2, u_3, u_4, u_5, u_6)$. *Les tables d'utilité* u_1,u_2 *et* u_3 *sont décrites respectivement dans les figures 2.3,2.5 et 2.4.*

La représentation graphique de ce jeu, rappelée ici, est la même que pour la figure 2.1

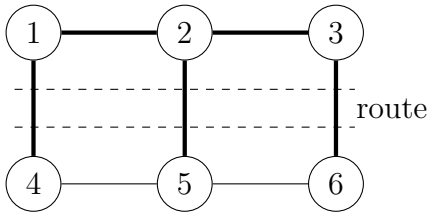


FIGURE 2.2 – Road Game

ω_1	ω_2	ω_4	u_1
M	M	M	0
M	M	R	1
M	R	M	1
M	R	R	2
R	M	M	2
R	M	R	1
R	R	M	1
R	R	R	1

FIGURE 2.3 – Utilités du joueur 1

ω_2	ω_3	ω_6	u_3
M	M	M	0
M	M	R	1
M	R	M	2
M	R	R	1
R	M	M	1
R	M	R	2
R	R	M	1
R	R	R	1

FIGURE 2.4 – Utilités du joueur 3

ω_1	ω_2	ω_3	ω_5	u_2
M	M	M	M	0
M	M	M	R	1
M	M	R	M	1
M	M	R	R	2
M	R	M	M	3
M	R	M	R	2
M	R	R	M	2
M	R	R	R	1
R	M	M	M	1
R	M	M	R	2
R	M	R	M	2
R	M	R	R	3
R	R	M	M	2
R	R	M	R	1
R	R	R	M	1
R	R	R	R	1

FIGURE 2.5 – Utilités du joueur 2

2.3 Jeux polymatriciels

Les jeux polymatriciels ont été conçus à la fin des années 60 [Yanovskaya, 1968] comme moyen pour représenter des jeux à plusieurs joueurs où les interactions entre les joueurs se font par paires. Un jeu polymatriciel est décrit par :

Définition 18 (Jeu polymatriciel)**.** *Un jeu polymatriciel est un tuple* $PG = \langle S, \Omega, E, u \rangle$*, où :*

- $S = \{1, ..., N\}$ *est l'ensemble des* N *joueurs.*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des stratégies jointes.* Ω_n *est l'ensemble des stratégies pures du joueur* n*.*
- E *un ensemble de paires d'éléments de* S*.* $\langle S, E \rangle$ *est un graphe non dirigé.*
- $u = \{u_\nu^{(n,n')} : \Omega_n \times \Omega_{n'} \to \mathbb{R}\}_{(n,n') \in E, \nu \in (n,n')}$ *est l'ensemble des tables d'utilités locales.* $u_\nu^{(n,n')}$ *est l'utilité du joueur* ν *pour le jeu local* (n, n')*.*

On peut représenter les interactions entre les joueurs via un graphe $\langle S, E \rangle$ où chaque joueur correspondant à un noeud, et E l'ensemble des paires de joueurs constituant un jeu local. Une arête $e = (n, n')$ avec $n \neq n'$, correspond à un jeu local $G^{(n,n')} = \langle \{n, n'\}, \Omega_n \times \Omega_{n'}, (u_n^{(n,n')}, u_{n'}^{(n,n')}) \rangle$. Les jeux locaux sont des jeux sous forme normale entre deux joueurs (jeux bimatriciels). Pour un jeu entre n et n', l'utilité définie par ce jeu local est notée $u_\nu^{(n,n')}(\omega_n, \omega_{n'})$ pour $\nu \in \{n, n'\}$.

Dans un jeu polymatriciel, l'action que joue un joueur dans un jeu local sera la même que dans tous les autres jeux locaux auxquels il participe. Les joueurs choisissent une stratégie et récupèrent les utilités de chaque jeu auquel ils participent de manière "simultanée" et additive.

La fonction d'utilité globale d'un joueur est donc la somme des utilités obtenues dans tous les jeux locaux auxquels il participe.

Définition 19 (Utilité globale)**.** *L'utilité d'un joueur* n *dans un jeu polymatriciel* $PG = \langle S, \Omega, E, u \rangle$*, pour une stratégie jointe* ω*, est :*

$$u_n(\omega) = \sum_{n', (n,n') \in E} u_n^{(n,n')}(\omega_n, \omega_{n'})$$

En prenant un jeu avec N joueurs où chaque joueur a m action, dans le pire des cas, un jeu polymatriciel demandera une taille de représentation $O((N^2 - N) \times m^2)$. Alors que la forme normale du jeu équivalent demanderait une représentation d'espace $O(N \times m^N)$.

Exemple 17 (Road Game)**.** *Prenons l'exemple d'un Road game similaire à celui vu dans l'exemple 15 mais avec 3 joueurs :*

- $N = 3$ *où* $\{1, 2, 3\}$ *est l'ensemble des joueurs*
- $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3$ *où* $\Omega_n = \{M, R\}$ *où* M *: construire un magasin,* R *: construire une résidence*
- $E = \{(1, 2), (2, 3)\}$
- $u = (u_1^{(1,2)}, u_2^{(1,2)}, u_2^{(2,3)}, u_3^{(2,3)})$*. Les tables d'utilité locales sont présentées dans les figures 2.7 et 2.8. Les utilités globale sont donc :*
  - $u_1(\omega_1, \omega_2, \omega_3) = u_1^{(1,2)}(\omega_1, \omega_2)$*. Si* $\omega = (R, M, R)$ *alors* $u_1(R, M, R) = 2$
  - $u_2(\omega_1, \omega_2, \omega_3) = u_2^{(1,2)}(\omega_1, \omega_2) + u_2^{(2,3)}(\omega_2, \omega_3)$*. Si* $\omega = (R, M, R)$ *alors* $u_2(R, M, R) = 1 + 1 = 2$
  - $u_3(\omega_1, \omega_2, \omega_3) = u_3^{(2,3)}(\omega_2, \omega_3)$*. Si* $\omega = (R, M, R)$ *alors* $u_3(R, M, R) = 1$

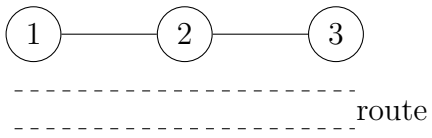


FIGURE 2.6 – Road Game

ω_1	ω_2	u_1	u_2
M	M	0	0
M	R	1	1
R	M	2	1
R	R	0	0

FIGURE 2.7 – Jeu entre le joueur 1 et 2

ω_2	ω_3	u_2	u_3
M	M	0	0
M	R	1	1
R	M	1	2
R	R	0	0

FIGURE 2.8 – Jeu entre le joueur 2 et 3

Sous la forme normale équivalente, ce jeu à 3 joueurs aurait la table suivante :

ω_1	ω_2	ω_3	u_1	u_2	u_3
M	M	M	0	0	0
M	M	R	0	1	1
M	R	M	1	2	2
M	R	R	1	1	0
R	M	M	2	1	0
R	M	R	2	2	1
R	R	M	0	1	2
R	R	R	0	0	0

FIGURE 2.9 – Représentation sous la forme normale du jeu polymatriciel

2.4 Jeux hypergraphiques

Les jeux hypergraphiques forment une généralisation des jeux polymatriciels et graphiques [Papadimitriou et Roughgarden, 2008]. Dans un jeu hypergrahique les joueurs jouent simultanément dans plusieurs jeux locaux comprenant chacun un sous-ensemble de joueurs (de taille potentiellement supérieure à 2). Leurs utilités sont comme dans les jeux polymatriciels, l'addition des utilités obtenues dans les jeux locaux auxquels ils participent.

Définition 20 (Jeu hypergraphique)**.** *Un jeu hypergraphique est un tuple* $G = \langle S, \Omega, E, u \rangle$ *où :*

- $S = \{1, ..., N\}$ *est l'ensemble des joueurs.*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des actions jointes avec* Ω_n*l'ensemble des actions du joueur* n*.*
- E *est un ensemble de sous ensembles de* S *correspondant à un ensemble d'hyperarêtes (ou à un ensemble de jeux locaux).*
- $u = (u_n^e : \Omega_e \to \mathbb{R})_{e \in E, n \in e}$ *est un ensemble de fonctions d'utilités.* $u_n^e(\omega_e)$ *est l'utilité du joueur* n *dans le jeu local* e *lorsque la stratégie* ω_e *est jouée.*

Un jeu hypergraphique peut être représenté par un hypergraphe $(\{1, ..., N\}, E)$ où $\{1, ..., N\}$ est l'ensemble des noeuds correspondant à chacun des joueurs et E est

l'ensemble des hyperarêtes caractérisant les interactions entre les joueurs. Chaque jeu local est un tuple $G_e = \langle e, \Omega_e, u^e \rangle$

Dans un jeu hypergraphique, comme dans un jeu polymatriciel, l'utilité globale d'un joueur est la somme des utilités locales obtenues dans tous les jeux auxquels il participe.

Définition 21 (Utilité globale)**.** *L'utilité d'un joueur n dans un jeu hypergraphique $G = \langle \{1, ..., N\}, \Omega, E, u \rangle$ est :*

$$u_n(\omega) = \sum_{e \in E, n \in e} u_n^e(\omega_e).$$

Proposition 3. *Un jeu polymatriciel est un jeu hypergraphique dans lequel $max_{e \in E}|e| = 2$.*

Un jeu graphique est un jeu hypergraphique dans lequel $|E| = (e_1, ..., e_N)$ et où, pour chaque $n \in S$, il existe un jeu e où $u_\nu^e(\omega_e) = 0, \forall \omega, \forall \nu \in e_n \backslash \{n\}$.

Exemple 18 (Road Game)**.** *Prenons un jeu similaire au Road game mais où les influences des choix ne se font plus au niveau de voisins sur une route mais de voisins dans une ville. Chaque joueur se trouvant au voisinage d'une autre ville jouera aussi dans cette dernière.*

Prenons un jeux hypergraphique à cinq joueurs tel que $\{1, 2, 3, 4, 5\}$ où les joueurs ont tous 2 actions. L'hypergraphe correspondant est représenté dans la figure 2.10

- $S = \{1, 2, 3, 4, 5\}$ *est l'ensemble des joueurs*
- $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3 \times \Omega_4 \times \Omega_5$ *est l'ensemble des actions jointes avec* $\Omega_n = \{M, R\}$.
- $E = (e_1, e_2) = (\{1, 2, 3\}, \{3, 4, 5\})$ *est l'ensemble des hyperarêtes correspondant à chacun des jeux locaux.*
- $u = \{u_1^{e_1}, u_2^{e_1}, u_3^{e_1}, u_3^{e_2}, u_4^{e_2}, u_5^{e_2}\}$ *sont les tables d'utilités de chacun des jeux locaux.*

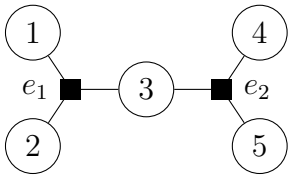


FIGURE 2.10 – Représentation hypergraphique

ω_1	ω_2	ω_3	u_1	u_2	u_3
M	M	M	0	0	0
M	M	R	1	1	2
M	R	M	1	2	1
M	R	R	2	1	1
R	M	M	2	1	1
R	M	R	1	2	1
R	R	M	1	1	2
R	R	R	0	0	0

FIGURE 2.11 – Jeu entre les joueurs $\{1, 2, 3\}$

ω_3	ω_4	ω_5	u_3	u_4	u_5
M	M	M	0	0	0
M	M	R	1	1	2
M	R	M	1	2	1
M	R	R	2	1	1
R	M	M	2	1	1
R	M	R	1	2	1
R	R	M	1	1	2
R	R	R	0	0	0

FIGURE 2.12 – Jeu entre les joueurs $\{3, 4, 5\}$

Si nous devions représenter ce même jeu sous sa forme normale nous aurions 5 *tables contenant chacune* 32 *éléments (pour un total de* 160*) alors qu'ici nous avons* 2 *tables de* 24 *éléments (pour un total de* 48*).*

Si les utilités du joueur 3 étaient représentées sous forme normale, on utiliserait une table de 32 éléments.

2.5 Autres représentations

Il existe plusieurs autres type de représentations succinctes.

2.5.1 Représentations succinctes à base de graphes

Les "action graph games" (AGG) sont des jeux qui représentent les interactions entre les actions des joueurs sous la forme d'un graphe. Les AGG permettent de représenter les jeux de manière compacte en fonction des actions du jeu, ainsi que représenter des jeux anonymes ou bien ayant des actions indépendante entre elles [Bhat et Leyton-Brown, 2004]. A la différence des jeux présentés précédemment, pour la représentation graphique du jeu, les noeuds ne représentent pas les joueurs, mais les actions. L'ensemble des noeuds correspond à l'ensemble des stratégies distinctes de tous les joueurs de telle manière que pour un ensemble $\Omega = \Omega_1 \times ... \times \Omega_N$, l'ensemble des noeuds est $\bar{\Omega} = \cup_{n \in S} \Omega_n$. L'ensemble des arêtes dirigées sert a caractériser l'influence qu'a une action sur les autres, elle peut pointer vers elle même lorsqu'elle peut s'influencer (plusieurs joueur peuvent jouer la même action). Des extensions des AGG [Jiang *et al.*, 2011] ont été proposé, qui comprennent des noeuds fonctions et permettent de généraliser la représentation par AGG.

Un autre exemple sont les Multi-Agent Influence Diagrams (MAID) qui vont proposer de représenter un jeu sous la forme d'un diagramme d'influence [Koller et Milch, 2001]. La différence entre un diagramme d'influence classique est que les noeuds de décision et les fonctions de récompenses sont réparties entre les différent joueurs.

2.5.2 Jeux anonymes

Dans les jeux dits "anonymes", les utilités des joueurs ne dépendent pas directement des actions jouées par les autres joueurs mais plutôt du nombre de fois que chacune des actions est jouée, indépendamment de l'identité des joueurs jouant ces actions. Cette particularité permet une représentation plus succincte des utilités. Dans ces jeux les joueurs ont tous les même actions.

Une représentation plus succincte qu'avec la forme normale de ces jeux (qui demande $N \times m^N$ nombres au total) est possible.

Dans un jeu anonyme (Anonymous game) [Nisan, 2007], lorsqu'il s'agit de calculer l'utilité d'un joueur selon une stratégie, il est seulement nécessaire d'évaluer le nombre de joueurs jouant chacune des stratégies. Pour N joueur avec m stratégie, la représentation demandera d'avoir $N \times m\binom{N+m-2}{m-1}$ nombres.

Exemple 19 (Représentation d'un jeu anonyme)**.** *Dans un jeu anonyme, les utilités des joueurs dépendent du nombres de joueurs jouant chaque action. Pour un jeu avec 3 joueurs et 2 actions par joueur, nous avons la représentation qui suit :*

		Nombre d'action joué par les autres joueurs		
		#0=2 #1=0	*#0=1 #1=1*	*#0=0 #1=2*
Joueur 1	*0*	*8*	*2*	*4*
	1	*6*	*2*	*7*

		Nombre d'action joué par les autres joueurs		
		#0=2 #1=0	*#0=1 #1=1*	*#0=0 #1=2*
Joueur 2	*0*	*8*	*9*	*1*
	1	*1*	*2*	*0*

		Nombre d'action joué par les autres joueurs		
		#0=2 #1=0	*#0=1 #1=1*	*#0=0 #1=2*
Joueur 3	*0*	*2*	*5*	*4*
	1	*3*	*1*	*6*

Ici si le joueur 3 *joue l'action* 0 *et que les autres joueurs jouent* $(0, 1)$ *ou* $(1, 0)$*, cela signifie que l'action* 0 *et* 1 *sont chacune jouées une fois par les autres joueurs donc l'utilité du joueur* 3 *sera* 5*.*

Nous parlerons d'un jeu symétrique [Nisan, 2007] lorsque nous avons un jeu anonymes pour lequel les tables d'utilité des joueurs sont identique. Pour N joueur avec m stratégie, la taille de représentation nécessaire sera de $m\binom{N+m-2}{m-1}$ nombres (qui est plus petite que la forme normale qui demandera $N \times m^N$), lorsque $N \mapsto \infty$ (et m est borné).

Les jeux de congestion [Rosenthal, 1973] sont un autre exemple de jeux anonymes. Ce sont des jeux à N joueurs avec M ressources où chaque action d'un joueur spécifie un sous ensemble de ressources. L'utilité d'un joueur dépendra des ressources qu'il choisit et du nombre de joueurs choisissant les même ressources.

Les AGG décrits précédemment généralisent les jeux anonymes.

2.5.3 Représentation compacte des tables d'utilité

Il existe d'autres représentations succinctes qui représentent les tables d'utilité de manière compacte.

Par exemple, dans un Boolean games [Harrenstein *et al.*, 2001] chaque joueur contrôle un ensemble de variables propositionnelles et son utilité est une formule propositionnelle (les utilités qui sont donc binaires).

2.6 Complexité des jeux succincts à base de graphe

Les représentations succinctes graphiques, polymatricielles et hypergraphiques présentées précédemment peuvent permettre de gagner une place exponentielle lorsqu'il s'agit de représenter un jeu, par rapport à sa forme normale. On peut se demander s'il est possible de résoudre des problèmes sur ces jeux en un temps polynomial par rapport à la taille de ces représentations.

Le problème de recherche de l'existence d'un équilibre de Nash pur pour les jeux hypergraphiques est un problème NP-complet [Gottlob *et al.*, 2005] en général, même pour les jeux avec un voisinage au plus à 3.

Lorsque nous prenons des cas particuliers, comme des jeux hypergraphiques acycliques avec des hyperarêtes de tailles bornées, le problème peut devenir PTIME selon les caractéristiques du graphe, notamment par rapport à la taille du voisinage [Gottlob *et al.*, 2005, Jiang et Safari, 2010].

Certains types de jeux succincts ont des propriétés particulières. Il est possible de classer les problèmes de recherche dans d'autres classes de complexité. Par exemple, les "congestion games" possèdent un équilibre de Nash pur, par conséquent ce type de problème appartient à TFNP, à la sous classe PLS pour être précis [Fabrikant *et al.*, 2004]. Il a été démontré que dans les jeux succincts (comprenant les jeux hypergraphiques, graphiques et polymatriciels) le *problème NASH* est un problème qui est PPAD complet comme dans les jeux sous forme normale [Daskalakis *et al.*, 2006].

2.7 Résolution des jeux succincts

Comme les jeux succincts peuvent être transformés en jeux sous forme normale, les algorithmes et méthodes adaptées aux jeux à forme normale sont utilisables pour résoudre tous ces jeux. Cependant ce type d'approche est exponentielle en espace, puisqu'il faut reconstruire les tables d'utilités globales.

En fonction du type de représentation succincte, il existe plusieurs algorithmes permettant de calculer plus directement les équilibres de Nash mixtes ou purs, de manière exacte ou approchée.

2.7.1 Algorithme de Howson pour les jeux polymatriciels

En se basant sur l'algorithme de Lemke Howson [Lemke et Howson, 1964] et son extension théorique aux jeux à N joueurs [Wilson, 1971], [Howson, 1972] propose une méthode de parcours de chemin utilisant une formulation du problème de recherche d'un équilibre de Nash mixte dans un jeu polymatriciel sous la forme d'un problème de complémentarité linéaire.

Pour un jeu polymatriciel $PG = \langle S, \Omega, E, u \rangle$, comme dans le cas bimatriciel, on représentera les fonctions d'utilité par des matrices. L'utilité du joueur n lorsqu'il joue avec un joueur n' est représentée dans une matrice $U^{nn'}$ de taille $m_n \times m_{n'}$. La cellule de coordonnées (i, j) correspondra à l'utilité de $u_n^{(n,n')}(i, j)$, avec $i \in \Omega_n$ et $j \in \Omega_{n'}$.

Construction d'un problème de complémentarité linéaire

A partir des matrices d'utilité $U^{n,n'}$, les matrices de désutilité positives $A^{n,n'} = -(U^{n,n'} - k_n E_{[m_n, m_{n'}]})$ sont créées, pour E une matrice de 1 (les dimensions de E dépendent de la taille des autres matrices utilisées) et pour k_n un entier tel que $U^{nn'} - k_n E_{[m_n, m_{n'}]} < 0$. Ayant des désutilités, le joueur ne cherchera plus à maximiser son utilité mais à minimiser ses désutilités.

La "valeur" d'un jeu pour le joueur n selon une stratégie mixte est :

$$v^n = (\xi^n)^T \sum_{n' \neq n} A^{nn'} \xi^{n'}$$

Pour e un vecteur colonne de 1, nous complétons les notations en utilisant :

$$y^n = \sum_{n' \neq n} A^{nn'} \xi^{n'} - v^i e$$

$$u^n = e^T \xi^n - 1$$

Où :

$$\xi^n \geqslant 0,\ y^n \geqslant 0,\ (\xi^n)^T y^n = 0, \quad 1 \leqslant n \leqslant N$$

$$v^n \geqslant 0,\ u^n \geqslant 0,\ (v^n)u^n = 0, \quad 1 \leqslant n \leqslant N$$

Ces équations définissent ce qui est équivalent à un problème de complémentarité linéaire [Howson, 1972] :

$$z \geqslant 0, w \geqslant 0,\ w = q + Mz,\ z^T w = 0$$

où

$$M = \begin{bmatrix} \mathbf{0} & A^{12} & ... & A^{1N} & -e & \mathbf{0} & ... & \mathbf{0} \\ A^{21} & \mathbf{0} & ... & A^{2N} & 0 & -e & ... & \mathbf{0} \\ ... & ... & ... & ... & ... & ... & ... & ... \\ A^{N1} & A^{N2} & ... & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & ... & -e \\ \mathbf{-1} & e^T & \mathbf{0} & ... & \mathbf{0} & \mathbf{0} & ... & ... & \mathbf{0} \\ \mathbf{-1} & \mathbf{0} & e^T & ... & \mathbf{0} & \mathbf{0} & ... & ... & \mathbf{0} \\ ... & ... & ... & ... & ... & ... & ... & ... & ... \\ \mathbf{-1} & \mathbf{0} & ... & e^T & \mathbf{0} & ... & ... & ... & \mathbf{0} \end{bmatrix} \quad z = \begin{bmatrix} \xi^1 \\ ... \\ \xi^N \\ v^1 \\ ... \\ v^N \end{bmatrix} \quad w = \begin{bmatrix} y^1 \\ ... \\ y^N \\ u^1 \\ ... \\ u^N \end{bmatrix} \quad q = \begin{bmatrix} \mathbf{0} \\ \mathbf{-1} \end{bmatrix}$$

Dans le cadre polymatriciel (et dans le cas avec N joueurs qui sera présenté plus en détail dans le chapitre 5), l'exécution de l'algorithme se fera différemment de la version bimatricielle.

Le principe de l'algorithme reste le même, il va s'agir de parcourir un ensemble de solutions presque complémentaires jusqu'à trouver une solution complémentaire. Cependant, il faut trouver un point presque complémentaire pour le parcours de chemin. En effet, dans un jeu à 2 joueurs, lorsqu'on part de la solution externe, on fait entrer une variable x du premier joueur dans la base (ce joueur a donc une stratégie mixte "légitime" [1]), puis, au prochain pivot, une variable du second joueur va entrer dans la base. Donc, dès le premier pivot, l'algorithme maintiendra des stratégies légitimes. Lorsque l'on a $N > 2$ joueurs, la solution externe ne comprend aucune stratégie légitime, il restera des stratégies non légitimes après le premier pivot. L'idée de Howson (la même que dans Wilson) est de commencer par fixer des stratégies pures pour les joueurs $n > 2$ légitimes, donc, puis à résoudre le jeu résultat sur les 2 premiers joueurs. La solution complémentaire de ce jeu à deux joueurs correspond, lorsqu'on lui adjoint les valeurs de x du troisième joueur, à une solution presque complémentaire du nouveau jeu à 3 joueurs, etc...

Plus précisément, une stratégie jointe pure ω^0 est sélectionnée de manière arbitraire afin d'associer une stratégie à chaque joueur. A partir de cette sélection, un "jeu" avec 1 joueur est créé en considérant que les joueurs autres que le joueur 1 jouent leur stratégie dans ω^0. Il est donc possible de calculer facilement la meilleure réponse, la stratégie à jouer permettant d'avoir une solution complémentaire. Cette solution permet d'avoir un point presque complémentaire pour le jeu à 2 joueurs où tous les joueurs sauf 1 et 2 jouent leur stratégie dans ω^0. Un parcours successif de points presque complémentaires est effectué à chaque niveau k pour passer à un niveau $k+1$ lorsqu'un point complémentaire est trouvé.

En reprenant les matrices décrites précédemment, nous décrivons le problème de complémentarité linéaire sous la forme d'un schéma de Tucker comme décrit dans la figure 2.13.

1. Des stratégies non uniformément nulles pour aucun des joueurs.

	1	ξ^1	ξ^2	...	ξ^N	v^1	v^2	...	v^N
y^1	**0**	**0**	A^{12}	...	A^{1N}	$-e$	**0**	...	**0**
y^2	**0**	A^{21}	**0**	...	A^{2N}	**0**	$-e$	...	**0**
.	.	.	.	.	.	.	.	.	.
y^N	**0**	A^{N1}	A^{N2}	..	**0**	**0**	**0**	...	$-e$
u^1	**-1**	e^T	**0**	...	**0**	**0**	...	...	**0**
u^2	**-1**	**0**	e^T	...	**0**	.			.
..	.	.	.		.	.			.
u^N	**-1**	**0**	**0**	..	e^T	**0**	...	...	**0**

FIGURE 2.13 – Schéma de Tucker d'un jeu polymatriciel

Le schéma de Tucker nous donne un point initial faisable du LCP qui est la solution "externe" : Toutes les variables ξ sont nulles, car hors base (en colonne). Cela ne définit évidemment pas un équilibre de Nash mais un point de départ de l'algorithme (comme pour les jeux bimatriciels).

Exemple 20 (Jeu polymatriciel à trois joueurs : Partie 1)**.** *Prenons un jeu polymatriciel à 3 joueurs dont le graphe est complet, les tables d'utilité sont représentées dans la figure2.14.*

		2	
		A	B
1	A	$2,0$	$0,1$
	B	$3,5$	$4,4$

		3	
		A	B
1	A	$3,4$	$5,1$
	B	$1,1$	$4,2$

		3	
		A	B
2	A	$3,6$	$2,5$
	B	$6,0$	$7,3$

FIGURE 2.14 – Matrice des jeux locaux pour un jeu polymatriciel à 3 joueurs

En utilisant ces utilités nous pouvons calculer les désutilités de chacun des joueurs, désutilités qui sont représentées dans la figure 2.15.

		2	
		A	B
1	A	$4,8$	$6,7$
	B	$3,3$	$2,4$

		3	
		A	B
1	A	$3,3$	$1,6$
	B	$5,6$	$2,4$

		3	
		A	B
2	A	$5,1$	$6,2$
	B	$2,7$	$1,4$

FIGURE 2.15 – Matrice de désutilité des jeux locaux

Utilisant les tables de désutilité, nous construisons le schéma de Tucker, représenté dans la figure 2.16.

	1	ξ_1^1	ξ_2^1	ξ_1^2	ξ_2^2	ξ_1^3	ξ_2^3	v^1	v^2	v^3
y_1^1	0	0	0	4	6	3	1	-1	0	0
y_2^1	0	0	0	3	2	5	2	-1	0	0
y_1^2	0	8	3	0	0	5	6	0	-1	0
y_2^2	0	7	4	0	0	2	1	0	-1	0
y_1^3	0	3	6	1	7	0	0	0	0	-1
y_2^3	0	6	5	2	4	0	0	0	0	-1
u^1	-1	1	1	0	0	0	0	0	0	0
u^2	-1	0	0	1	1	0	0	0	0	0
u^3	-1	0	0	0	0	1	1	0	0	0

FIGURE 2.16 – Schéma de Tucker du jeu

Algorithme de Howson

En reprenant les notions de base et hors base utilisées pour Lemke Howson dans le cas bimatriciel, les variables $z = \begin{bmatrix}\xi & v\end{bmatrix}$ sont hors base et les variables associées aux lignes $w = \begin{bmatrix}y & u\end{bmatrix}$ sont dans la base. Il y a une colonne 1 qui correspond à q si l'on veut obtenir $w = Mz + q$. Ainsi en utilisant le principe de l'algorithme de Lemke Howson, nous allons faire pivoter les variables successivement jusqu'à obtenir une matrice qui soit une matrice complémentaire, c'est-à-dire qu'aucun couple (ξ_i^n, y_i^n) ou (u^n, v^n) n'est entièrement dans la base et la matrice est "faisable" (cela signifie que la première colonne est positive ou nulle).

Si la matrice est complémentaire et faisable, nous avons une solution du LCP, qui peut être traduite en un équilibre de Nash. Toutes les variables ξ_i^n se trouvant dans la base (lignes) vont prendre les valeurs de la première colonne (la colonne $1/q$), toutes les autres prennent 0 comme valeur.

Étape 1 : Associons à chaque joueur une stratégie et notons $\omega^0 = \{\omega_1^0, \omega_1^0, ..., \omega_N^0\}$, la stratégie jointe pure où un joueur n joue sa stratégie ω_n^0 . Cette action jointe produit l'ensemble de colonnes $\xi_{\omega^0} = \{\xi^1_{\omega_1^0}, ..., \xi^N_{\omega_N^0}\}$ qui correspond à toutes les variables de ξ pour lesquelles les actions associées se trouvent dans l'action jointe initiale ω^0. L'ensemble de colonnes $\xi_{-\omega^0}$ correspond à toutes les variables de ξ qui n'ont pas d'actions associées à l'action jointe ω^0.

En reprenant le schéma de Tucker, nous le décomposons sous la forme des sous matrices qui suivantes :

	ξ_{ω^0}	$q, \xi_{-\omega^0}, v$
u	M_{11}	M_{12}
Y	M_{21}	M_{22}

Après avoir effectué un pivot par bloc (toutes les ligne de u pivotent avec les colonnes ξ_{ω^0}) nous avons :

	u	$q, \xi_{-\omega^0}, v$
ξ_{ω^0}	$M_{11}^P = M_{11}^{-1}$	$M_{12}^P = -M_{11}^{-1}M_{12}$
Y	$M_{21}^P = M_{21}M_{11}^{-1}$	$M_{22}^P = M_{22} - M_{21}M_{11}^{-1}M_{12}$

Étant donné que la sous matrice M_{11} est une matrice identité nous pouvons écrire :

	u	$q, \xi_{-\omega^0}, v$
ξ_{ω^0}	$M_{11}^P = M_{11}$	$M_{12}^P = -M_{12}$
Y	$M_{21}^P = M_{21}$	$M_{22}^P = M_{22} - M_{21}M_{12}$

Ceci a pour but de calculer la matrice initiale, correspondant à l'action jointe ω^0. C'est un point presque complémentaire lorsque l'on effectue des pivots au niveau $k = 1$.

Exemple 21 (Jeu polymatriciel à trois joueurs : Partie 2)**.** *En reprenant le jeu de l'exemple 20, nous allons maintenant appliquer la première étape.*

Nous sommes actuellement à l'étape 1*, nous devons faire un pivot par bloc. Avant de faire le pivot par bloc il faut choisir une stratégie initiale* ω^0*. Ici nous choisissons* $\omega^0 = (1, 1, 1)$*, ce qui nous donne après pivot, le schéma de Tucker de la figure 2.17.*

	u^1	u^2	u^3	1	ξ_2^1	ξ_2^2	ξ_2^3	v^1	v^2	v^3
ξ_1^1	1	0	0	1	-1	0	0	0	0	0
ξ_1^2	0	1	0	1	0	-1	0	0	0	0
ξ_1^3	0	0	1	1	0	0	-1	0	0	0
y_1^1	0	4	3	7	0	2	-2	-1	0	0
y_2^1	0	3	5	8	0	-1	-3	-1	0	0
y_1^2	8	0	5	13	-5	0	1	0	-1	0
y_2^2	7	0	2	9	-3	0	-1	0	-1	0
y_1^3	3	1	0	4	3	6	0	0	0	-1
y_2^3	6	2	0	8	-1	2	0	0	0	-1

FIGURE 2.17 – Schéma de Tucker obtenu après le pivot par bloc

Étape 2 : Pour le niveau k actuel, nous effectuons un pivot pour faire rentrer la variable v^k dans la base tout en maintenant la colonne q positive. Lors de ce pivot la variable sortie sera toujours une variable de y^k car les variables y_i^k sont les seuls qui permettent un pivot maintenant la colonne q positive.

Étape 3 : Si $\xi^k_{\omega_k^0} y^k_{\omega_k^0} = 0$ (c'est a dire si $\xi^k_{\omega_k^0}$ est dans la base alors $y^k_{\omega_k^0}$ doit être hors base et si $y^k_{\omega_k^0}$ est dans la base alors $\xi^k_{\omega_k^0}$ doit être hors base) alors le point actuel est complémentaire pour le niveau k donc nous passons à l'étape 4.

En maintenant la colonne q positive et la condition $\sum_{n=1}^{k} (\sigma^n)^T y^n = \xi^k_{\omega_k^0} y^k_{\omega_k^0} = 0$ nous effectuons des pivots jusqu'à obtenir $\xi^k_{\omega_k^0} y^k_{\omega_k^0} = 0$ ou $v^k = 0$.

La variable que l'on entre dans la base lors d'un pivot sera la variable "jumelle" de celle que nous venons de sortir de la base lors du dernier pivot. Par exemple supposons que nous venons de sortir la variable y_i^n de la base et que notre point n'est pas complémentaire, alors la variable que nous devons entrer dans la base sera la variable correspondante/jumelle ξ_i^n. De manière similaire, si nous venions de sortir la variable ξ_i^n de la base alors ce serait y_i^n que nous devrions faire entrer dans la base.

Après avoir effectué un pivot, si la variable sortie est v^k on passe à l'étape 5 sinon cela signifie que nous avons sorti une variable ξ_i^n ou y_i^n, donc nous recommençons l'étape 3.

Étape 4 : Si $k < N$ alors $k = k + 1$ et l'algorithme retourne à l'étape 2.

Si nous sommes au niveau $k = N$ alors cela signifie que nous avons un équilibre de Nash. Au niveau N, les conditions de complémentarité sont maintenues pour tous les joueurs, donc nous avons un point qui est complémentaire et la colonne q reste positive. C'est donc une solution faisable et complémentaire du LCP, correspondant donc à un équilibre de Nash du jeu initial. Pour chaque variable ξ_i^n, si cette variable est hors base elle a pour valeur 0, si elle est dans la base elle prend la valeur de la ligne qui lui est associée pour la colonne q.

Étape 5 : L'algorithme descend d'un niveau tel que $k = k - 1$ et fait rentrer dans la base la variable $\xi^k_{\omega_k^0}$ ou $y^k_{\omega_k^0}$ qui se trouve hors base, toujours en maintenant la colonne q positive. L'algorithme retourne ensuite à l'étape 3.

Exemple 22 (Jeu polymatriciel à trois joueurs : Partie 3)**.** *Reprenons l'exemple 21 après avoir effectué le pivot par bloc de l'étape* 1*.*

Passons à l'étape 2 *et initialisons* $k = 1$. *Il faut faire rentrer la variable* v^1 *dans la base en en faisant sortir une autre.*

La variable v^1 *entre dans la base et la variable* y_1^1 *en sort*

Le pivot s'effectue comme si nous avions les équations suivantes :

$y_1^1 = 4u^2 + 3u^3 + 7 + 2\xi_2^2 - 2\xi_2^3 - v^1$ *qui devient* $v^1 = 4u^2 + 3u^3 + 7 + 2\xi_2^2 - 2\xi_2^3 - y_1^1$

On obtient donc :

$$\begin{aligned} y_2^1 &= 3u^2 + 5u^3 + 8 - \xi_2^2 - 3\xi_2^3 - v^1 \\ y_2^1 &= 3u^2 + 5u^3 + 8 - \xi_2^2 - 3\xi_2^3 - (4u^2 + 3u^3 + 7 + 2\xi_2^2 - 2\xi_2^3 - y_1^1) \\ y_2^1 &= -1u^2 + 2u^3 + 1 - 3\xi_2^2 - \xi_2^3 + y_1^1 \end{aligned}$$

Après avoir fait les pivots pour toutes les lignes nécessaires, le schéma obtenu est le suivant :

	u^1	u^2	u^3	1	ξ_2^1	ξ_2^2	ξ_2^3	y_1^1	v^2	v^3
ξ_1^1	1	0	0	1	−1	0	0	0	0	0
ξ_1^2	0	1	0	1	0	−1	0	0	0	0
ξ_1^3	0	0	1	1	0	0	−1	0	0	0
v^1	0	4	3	7	0	2	−2	−1	0	0
y_2^1	0	−1	2	1	0	−3	−1	1	0	0
y_1^2	8	0	5	13	−5	0	1	0	−1	0
y_2^2	7	0	2	9	−3	0	−1	0	−1	0
y_1^3	3	1	0	4	3	6	0	0	0	−1
y_2^3	6	2	0	8	−1	2	0	0	0	−1

Passons à l'étape 3. Nous observons qu'il y a complémentarité au niveau $k = 1$ *car* ξ_1^1 *est dans la base et* y_1^1 *est hors base ; donc* $\xi^1_{\omega_1^0} y^1_{\omega_1^0} = 0$. *Passons donc à l'étape* 4.

Étant donné que $k < 3$ *nous incrémentons* k *et avons donc maintenant* $k = 2$ *l'algorithme retourne à l'étape 2.*

La variable v^2 *entre dans la base et la variable* y_2^2 *en sort ce qui donne :*

	u^1	u^2	u^3	1	ξ_2^1	ξ_2^2	ξ_2^3	y_1^1	y_2^2	v^3
ξ_1^1	1	0	0	1	−1	0	0	0	0	0
ξ_1^2	0	1	0	1	0	−1	0	0	0	0
ξ_1^3	0	0	1	1	0	0	−1	0	0	0
v^1	0	4	3	7	0	2	−2	−1	0	0
y_2^1	0	−1	2	1	0	−3	−1	1	0	0
y_1^2	1	0	3	4	−2	0	2	0	1	0
v^2	7	0	2	9	−3	0	−1	0	−1	0
y_1^3	3	1	0	4	3	6	0	0	0	−1
y_2^3	6	2	0	8	−1	2	0	0	0	−1

La condition de complémentarité n'est pas respectée car ξ_1^2 *et* y_1^2 *sont toutes les deux dans la base. Nous devons effectuer un nouveau pivot. Étant donné que nous avons sorti* y_2^2 *de la base précédemment, nous devons faire entrer* ξ_2^2 *dans la base, ainsi nous conservons tous les produits* $\xi^k_{\omega_k^0} y^k_{\omega_k^0}$ *qui étaient nuls. La colonne* 1 *sera maintenue positive seulement si* y_2^1 *sort de la base.*

	u^1	u^2	u^3	1	ξ_2^1	y_2^1	ξ_2^3	y_1^1	y_2^2	v^3
ξ_1^1	1	0	0	1	-1	0	0	0	0	0
ξ_1^2	0	$\frac{4}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	$-\frac{1}{3}$	0	0
ξ_1^3	0	0	1	1	0	0	-1	0	0	0
v^1	0	$\frac{10}{3}$	$\frac{13}{3}$	$\frac{23}{3}$	0	$-\frac{2}{3}$	$-\frac{8}{3}$	$-\frac{1}{3}$	0	0
ξ_2^2	0	$-\frac{1}{3}$	$\frac{2}{2}$	$\frac{1}{3}$	0	$-\frac{1}{3}$	$-\frac{1}{3}$	$\frac{1}{3}$	0	0
y_1^2	1	0	3	4	-2	0	2	0	1	0
v^2	7	0	2	9	-3	0	-1	0	-1	0
y_1^3	3	-1	4	6	3	-2	-2	2	0	-1
y_2^3	6	$\frac{4}{3}$	$\frac{4}{3}$	$\frac{26}{3}$	-1	$-\frac{2}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$	0	-1

La condition de complémentarité n'est pas respectée car ξ_1^2 et y_1^2 sont toutes les deux dans la base. Donc étant donné que nous avons sorti y_2^1 nous devons faire entrer ξ_2^1 qui permet de maintenir la colonne 1 *positive.*

	u^1	u^2	u^3	1	ξ_1^1	y_2^1	ξ_2^3	y_1^1	y_2^2	v^3
ξ_2^1	1	0	0	1	-1	0	0	0	0	0
ξ_1^2	0	$\frac{4}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	$-\frac{1}{3}$	0	0
ξ_1^3	0	0	1	1	0	0	-1	0	0	0
v^1	0	$\frac{10}{3}$	$\frac{13}{3}$	$\frac{23}{3}$	0	$-\frac{2}{3}$	$-\frac{8}{3}$	$-\frac{1}{3}$	0	0
ξ_2^2	0	$-\frac{1}{3}$	$\frac{2}{2}$	$\frac{1}{3}$	0	$-\frac{1}{3}$	$-\frac{1}{3}$	$\frac{1}{3}$	0	0
y_1^2	-1	0	3	2	2	0	2	0	1	0
v^2	4	0	2	6	3	0	-1	0	-1	0
y_1^3	6	-1	4	9	-3	-2	-2	2	0	-1
y_2^3	5	$\frac{4}{3}$	$\frac{4}{3}$	$\frac{23}{3}$	1	$-\frac{2}{3}$	$-\frac{2}{3}$	$\frac{2}{3}$	0	-1

La condition de complémentarité n'est pas respectée car ξ_1^2 et y_1^2 sont toutes les deux dans la base. Donc étant donné que nous avons sorti ξ_1^1 nous devons faire entrer y_1^1, la positivité de la colonne 1 *n'est maintenue que si ξ_1^2 sort.*

	u^1	u^2	u^3	1	ξ_1^1	y_2^1	ξ_2^3	ξ_1^2	y_2^2	v^3
ξ_2^1	1	0	0	1	-1	0	0	0	0	0
y_1^1	0	4	-2	2	0	1	1	-3	0	0
ξ_1^3	0	0	1	1	0	0	-1	0	0	0
v^1	0	2	5	7	0	-1	-3	1	0	0
ξ_2^2	0	1	0	1	0	0	0	-1	0	0
y_1^2	-1	0	3	2	2	0	2	0	1	0
v^2	4	0	2	6	3	0	-1	0	-1	0
y_1^3	6	7	0	13	-3	0	0	-6	0	-1
y_2^3	5	4	0	9	1	0	0	-2	0	-1

Nous sommes complémentaire au niveau $k = 2$. Donc nous passons au niveau suivant où $k = 3$.

Pour entrer v^3 dans la base, il faut sortir y_2^3

	u^1	u^2	u^3	1	ξ_1^1	y_2^1	ξ_2^3	ξ_1^2	y_2^2	y_2^3
ξ_2^1	1	0	0	1	−1	0	0	0	0	0
y_1^1	0	4	−2	2	0	1	1	−3	0	0
ξ_1^3	0	0	1	1	0	0	−1	0	0	0
v^1	0	2	5	7	0	−1	−3	1	0	0
ξ_2^2	0	1	0	1	0	0	0	−1	0	0
y_1^2	−1	0	3	2	2	0	2	0	1	0
v^2	4	0	2	6	3	0	−1	0	−1	0
y_1^3	1	3	0	4	−4	0	0	−4	0	1
v^3	5	4	0	9	1	0	0	−2	0	−1

Nous n'avons toujours pas un schéma complémentaire car ξ_1^3 et y_1^3 sont toutes les deux dans la base.

Étant donné que nous venons de sortir y_2^3 de la base, nous devons faire entrer ξ_2^3 dans la base. Nous sommes contraints de sortir ξ_1^3 si nous voulons maintenir la colonne 1 *positive.*

	u^1	u^2	u^3	1	ξ_1^1	y_2^1	ξ_1^3	ξ_1^2	y_2^2	y_2^3
ξ_2^1	1	0	0	1	−1	0	0	0	0	0
y_1^1	0	4	−1	3	0	1	−1	−3	0	0
ξ_2^3	0	0	1	1	0	0	−1	0	0	0
v^1	0	2	2	4	0	−1	3	1	0	0
ξ_2^2	0	1	0	1	0	0	0	−1	0	0
y_1^2	−1	0	5	4	2	0	−2	0	1	0
v^2	4	0	1	5	3	0	1	0	−1	0
y_1^3	1	3	0	4	−4	0	0	−4	0	1
v^3	5	4	0	9	1	0	0	−2	0	−1

Le schéma est complémentaire au niveau 3*, le problème contient 3 joueurs, nous avons donc une solution qui correspond à un équilibre de Nash.*

Afin de récupérer les distributions de probabilité correspondantes, nous prenons les valeurs dans la colonne 1 *des variables ξ qui sont dans la base : $\xi_2^1 = 1$,$\xi_2^2 = 1$ et $\xi_2^3 = 1$ Nous avons donc $\xi^1 = (0, 1)$,$\xi^2 = (0, 1)$ et $\xi^3 = (0, 1)$ Les utilités des joueurs* 1*,* 2 *et* 3 *sont respectivement* 8*,* 11 *et* 5*, si l'on change leurs stratégies de manière unilatérale les utilités possibles seront respectivement 5, 7 et 1 : aucune amélioration possible pour aucun des joueurs ce qui confirme que nous avons bien un équilibre de Nash (pur ici).*

2.7.2 Autres méthodes pour les jeux polymatriciels

Il existe d'autres algorithmes utilisables pour la recherche d'équilibre de Nash dans un jeu polymatriciel.

Un algorithme utilisant une méthode basée sur la descente de gradient a été proposée afin de calculer un équilibre de Nash mixte approché [Deligkas *et al.*, 2015]. L'algorithme a pour but d'essayer de minimiser le regret d'un joueur entre la meilleure réponse possible et l'utilité qu'il a actuellement pour la stratégie qui est jouée. L'algorithme consiste à partir d'une stratégie arbitraire et de construire itérativement une nouvelle stratégie pour laquelle le regret maximum a été réduit.

Cet algorithme a été comparé à celui de Howson pour les jeux polymatriciels [Deligkas *et al.*, 2016], les deux algorithmes permettent d'obtenir de bons résultats pour la plupart des cas comparés. La descente de gradient permet, en général, d'obtenir une solution plus rapidement et même si elle est approchée, cette solution est de bonne qualité en général.

Il existe d'autres algorithmes s'intéressant à des cas plus particuliers de jeux polymatriciels. L'un d'entre eux s'intéresse aux problèmes de meneur suiveur et à la recherche d'équilibre pessimiste (en défaveur du meneur) et optimiste (en faveur du meneur) en utilisant la programmation linéaire [De Nittis *et al.*, 2018]. Un autre algorithme s'intéresse à la recherche d'équilibre approché dans les jeux polymatriciels représentables sous la forme d'un arbre en utilisant un algorithme de Las Vegas [Barman *et al.*, 2015].

2.7.3 Méthodes pour les Jeux graphiques

Il existe différents algorithmes applicables afin de trouver un équilibre pour un jeu graphique.

TreeNash et NashProp

Le premier algorithme développé pour la recherche d'un équilibre de Nash dans un jeu graphique est l'algorithme abstrait *TreeNash* [Kearns *et al.*, 2001]. C'est un algorithme utilisable pour les jeux graphiques dont la structure est représentable par un arbre, c'est-à-dire un graphe connexe, non orienté et acyclique.

Le principe de cet algorithme est d'effectuer ce que l'on peut comparer à de la propagation de message dans l'arbre depuis les feuilles vers la racine puis de la racine vers les feuilles.

Dans un arbre pour un noeud donné, nous considérerons tous les noeuds se trouvant entre la racine et le noeud actuel comme étant en aval. Pour ce même noeud nous considérons tous les noeuds se trouvant entre une feuille et ce noeud comme étant en amont. Chaque noeud aura un noeud en aval (enfant) et au moins un noeud en amont (parent), à l'exception des feuilles qui n'ont pas de parent et de la racine qui n'a pas d'enfant.

FIGURE 2.18 – Exemple de jeu graphique arborescent [2]

Lors de l'exécution de l'algorithme, il y a deux phases différentes : un passage vers l'aval et un passage vers l'amont.

Un noeud envoie à son enfant un tableau à 2 dimensions et à valeurs $\{0, 1\}$ qui précise si, pour une combinaison de stratégies mixtes entre l'enfant et le parent, il existe un équilibre de Nash mixte pour le parent lorsqu'il joue cette stratégie avec les

2. Pour un jeu graphique, $E = \{e_1, ..., e_N\}$ avec $n \in e_n$. S'il est représentable par un graphe G cela signifie $e_n = \{n', n' \in V_G(n)\} \cup \{n\}$ où $V_G(n)$ est l'ensemble des voisins de n dans G

autres noeuds en amont de ce dernier. Pour un parent n avec la stratégie ξ_n et son enfant ν et la stratégie ξ_ν, $T(\xi_\nu, \xi_n) = 1$ si et seulement si, il existe un équilibre de Nash dont les stratégies mixtes de n et ν sont (ξ_n, ξ_ν).

Lors de la phase "vers l'aval", l'algorithme commence par les feuilles de l'arbre (le parcours est effectué par niveau des feuilles vers la racine) et initialise la table binaire pour récupérer les meilleures réponses aux stratégies possibles de leur enfant et "transmettre" cette table à leur enfant. Lorsque le noeud n'est pas une feuille, en prenant la table binaire de chacun des parents, le noeud enfant va à son tour créer sa table binaire en tenant compte du fait qu'il n'existe un équilibre avec son enfant que s'il existe une stratégie mixte jointe de ses parents qui le permet. Cette stratégie mixte jointe sera ajoutée à une liste de "témoins" si c'est le cas. La table binaire ainsi que les témoins sont passés à l'enfant.

Une fois la racine atteinte, le passage en amont commence dans l'ordre inverse (racine vers les feuilles). Pour un noeud donné (commençant par la racine), nous choisissons n'importe quelle stratégie pour laquelle un équilibre est possible, parmi la liste des témoins. Elle est ensuite envoyée aux parents. Pour chaque noeud recevant une stratégie de son enfant, il choisit à son tour un témoin parmi ceux qui existent, associé à la stratégie qu'il vient de recevoir, en utilisant la table binaire. Une fois cette opération effectuée pour tous les noeuds, la stratégie jointe obtenue est un équilibre de Nash.

Il existe deux versions de cet algorithme. La première est *ApproximateTreeNash* [Kearns *et al.*, 2001] qui retourne un équilibre approché. Dans cette version, les joueurs vont utiliser des stratégies mixtes discrétisées et récupérer non pas les meilleures réponses dans les tables binaires mais les meilleures réponses approchées.

La seconde est *ExactTreeNash* [Kearns *et al.*, 2001] qui calcule un équilibre exact. Le tableau transmis entre les noeuds sera un tableau dans lesquels les stratégies mixtes des joueurs seront décomposées et ordonnées selon une liste d'intervalle. Un sous-ensemble binaire est associé à chacun de ces intervalles.

L'algorithme *NashProp* est une "extension" de l'algorithme de *TreeNash* aux jeux graphiques dont le graphe est cyclique (il reste utilisable lorsque le graphe est acyclique) [Ortiz et Kearns, 2003]. Comme *TreeNash*, il s'inspire de la propagation de croyance dans un réseau bayésien. Il possède deux phases qui sont similaires à celles présentées précédemment.

Durant la première phase de l'algorithme, qui s'effectue plusieurs fois, il s'agit d'échanger des tables entre les joueurs. Ces tables caractérisent la "croyance" qu'a un joueur concernant l'existence d'un équilibre pour deux stratégies. Initialement, les joueurs considèrent que toutes les combinaisons possibles permettent d'obtenir un équilibre. Les joueurs vont progressivement retirer les combinaisons qu'ils considèrent comme impossibles. Par conséquent, ils convergeront vers une solution et n'élimineront jamais un équilibre possible.

Durant la seconde phase, les joueurs vont progressivement construire un équilibre en se transmettant les tables obtenues à la fin de la première phase. De manière similaire à *TreeNash*, il s'agit d'associer au noeud une stratégie pour laquelle il existe un équilibre possible et de récupérer les témoins/stratégies mixtes jointes permettant d'avoir un équilibre de Nash.

2.7.4 Autres algorithmes sur les jeux graphiques

Depuis la proposition des jeux graphiques, il y a eu des algorithmes utilisant des méthodes différentes pour calculer des équilibres de Nash.

Par exemple, [Vickrey et Koller, 2002] proposent plusieurs algorithmes. L'un d'entre eux permet de calculer un équilibre approché avec une méthode de minimisation de fonction (hill climbing). Elle consiste à définir une fonction score mesurant la distance entre une stratégie et un équilibre, d'utiliser un algorithme de recherche locale commençant par une stratégie initiale aléatoire et l'améliorer progressivement jusqu'à maximiser localement la fonction score.

Un autre algorithme proposé dans [Vickrey et Koller, 2002] se base sur la résolution d'un CSP. Deux autres algorithmes sont aussi proposés, l'un est utilisé pour calculer un équilibre approché en cherchant un équilibre avec des algorithmes de CSP ou de minimisation de coût, l'autre consiste à décomposer le jeu en sous-jeux et à les résoudre séparément avant de les combiner.

L'algorithme *PureProp* présenté dans [Soni *et al.*, 2007] est un algorithme formulant le problème sous la forme d'un problème de satisfaction de contraintes, de manière à trouver un équilibre de Nash pur approché.

Une autre méthode utilisant des champs aléatoires de Markov permet de trouver s'il existe un équilibre de Nash pur dans un jeu graphique [Daskalakis, 2006].

Un algorithme de backtrack asynchrone de recherche d'un équilibre de Nash mixte approché dans les jeux graphiques utilisant des CSP distribués a été développé [Grubshtein et Meisels, 2012].

2.7.5 Méthodes pour les jeux hypergraphiques

Il existe peu d'algorithmes qui peuvent être utilisés pour trouver des équilibres de Nash dans les jeux hypergraphiques.

Certains de ces algorithmes s'inspirent d'algorithmes existant pour les jeux graphiques.

L'algorithme nommé Value Nash Propagation (VNP) permet de calculer un équilibre de Nash pur [Chapman *et al.*, 2010]. Cet algorithme utilise une propagation de message similaire à TreeNash. Deux versions de cet algorithme existent, l'une est une version complète qui fonctionne lorsque l'hypergraphe est un arbre (hypergraphe acyclique), l'autre est une version approchée lorsque le jeu contient un hypergraphe cyclique.

Enfin, [Wahbi et Brown, 2016] proposent un algorithme asynchrone permettant de trouver un équilibre de Nash approché transformant le jeu hypergraphique en un problème asymétrique distribué de satisfaction de contraintes.

2.8 Conclusion

Ce chapitre a présenté différents types de jeux succincts, notamment les jeux polymatriciels, graphiques et hypergraphiques. Ces représentations offrent potentiellement un gain en espace exponentiel. Nous avons décrit la complexité de description et résolution de ces jeux, puis présentée brièvement les algorithmes de résolution existant. Nous avons mis l'accent sur les algorithmes de Howson (dans ce chapitre), Lemke Howson et Wilson (dans le chapitre 1), dont nous nous sommes inspirés dans cette thèse.

Le prochain chapitre va s'intéresser aux jeux à information incomplète, aux jeux bayésiens, ainsi qu'à leurs liens avec les jeux succincts.

Chapitre 3

Jeux à information incomplète

3.1 Introduction

La théorie des jeux a pour but de représenter, expliquer et optimiser des situations où plusieurs agents, les joueurs, ont le choix entre plusieurs actions. L'utilité reçue par un joueur, qu'il souhaite maximiser, dépend de son action choisie ainsi que des actions choisies par les autres joueurs. Dans un jeu non coopératif simultané à information complète, les joueurs ne peuvent pas coordonner leurs actions mais chacun a une connaissance totale du jeu : les joueurs, les actions possibles et les utilités.

L'hypothèse d'une information complète n'est pas toujours satisfaite : les joueurs peuvent ne connaître que partiellement le jeu. Leur connaissance des utilités des autres joueurs peut, en particulier, être incomplète. Les jeux bayésiens [Harsanyi, 1967, Harsanyi, 1968b, Harsanyi, 1968a], proposés dans les années soixante, permettent de modéliser la connaissance incomplète d'un jeu et la possibilité qu'ont les joueurs de disposer d'informations supplémentaires, représentées par la notion de "type". Le "type" d'un joueur regroupe les informations sur le jeu auxquelles il a accès.

3.2 Jeux à information imparfaite/incomplète

Le problème de représentation des jeux à information incomplète a été abordé initialement avec la définition des jeux sous forme extensive [Von Neumann et Morgenstern, 1944]. Ces jeux permettent de représenter des situations où les joueurs obtiennent des informations différentes lors de la progression du jeu. Ces jeux sont des jeux à "information imparfaite".

Le terme "information incomplète" [Von Neumann et Morgenstern, 1944] fait référence à un jeu où une partie de la forme normale n'est pas précisée. L'incertitude d'un joueur sur les différents paramètres du jeu peut être modélisée via un jeu sous forme extensive avec information incomplète. Dans un jeu sous forme extensive avec information complète, un joueur à connaissance de l'intégralité de "l'histoire" du jeu pour un noeud dans lequel il se trouve (comme aux échecs par exemple). Dans un jeu sous forme extensive avec information incomplète, un joueur n'a pas forcement connaissance de "l'histoire" du jeu pour le noeud où il se trouve actuellement, il peut envisager de se trouver dans plusieurs noeuds différents (comme dans la figure 3.1).

Afin de pouvoir analyser les jeux à information incomplète, une généralisation sous forme normale a été proposée [Luce *et al.*, 1957] dans laquelle les joueurs ne connaissent pas toutes les utilités des autres joueurs. Dans [Luce *et al.*, 1957], une première formulation a été faite sous forme d'un jeu à N joueurs avec N^2 fonctions d'utilité où

chacune de ces fonctions caractérise la croyance d'un joueur n sur la fonction d'utilité d'un autre joueur n'.

Cependant cette représentation engendre quelques problèmes lorsqu'il s'agit de, par exemple, représenter la croyance d'un joueur sur la croyance d'un autre ou bien sur la "fiabilité" de la croyance d'un joueur sur sa propre utilité comparée à celle d'un autre joueur.

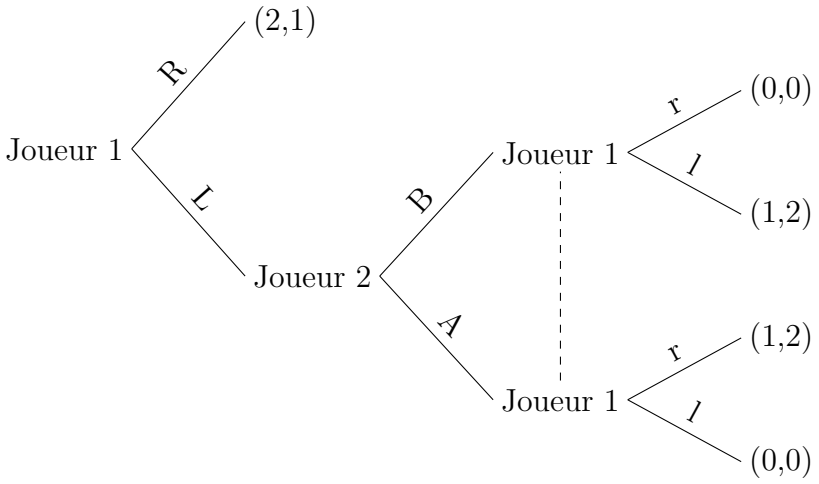


FIGURE 3.1 – Représentation d'un jeu à information incomplète où le joueur 1 n'est pas informé de l'action jouée par le joueur 2

3.3 Jeux Bayésiens

Les jeux bayésiens ont été proposés par [Harsanyi, 1967] comme moyen de représentation des jeux à information incomplète. Dans un jeu bayésien, il existe un ensemble de "mondes possibles" dans lesquels les joueurs peuvent jouer mais sans avoir une connaissance certaine de celui dans lequel ils vont jouer (chaque "monde" correspond à un jeu à information complète).

L'utilité u_n d'un joueur n ne dépend pas seulement des actions des autres joueurs mais aussi du "vrai" monde, inconnu, dans lequel il joue. Les joueurs ne savent pas exactement dans quel "vrai" monde ils jouent mais ils partagent une connaissance probabiliste dessus. De plus, avant de jouer, chaque joueur n recevra une information "privée" sur le monde réel qui associera cette information à un élément θ_n d'un ensemble Θ_n. Nous appellerons θ_n le type d'un joueur n et Θ_n l'ensemble des types possibles du joueur n. Plusieurs mondes différents peuvent avoir une information amenant vers un même type pour un joueur n. Après avoir reçu l'information, un joueur a une idée plus précise concernant le jeu dans lequel il va jouer, cependant il y a toujours une incertitude concernant le jeu réel dans lequel il jouera.

Par conséquent, chaque combinaison de types $\theta = (\theta_1, ..., \theta_N)$ (les types joints) caractérise l'ensemble des informations reçues par chacun des joueurs. Un joueur n ne connaît pas les types des autres joueurs (θ_{-n}), mais il dispose d'une connaissance incertaine sur celle-ci, modélisée par $P(\theta_{-n}|\theta_n)$.

Dans un jeu bayésien, la notion de type telle que définie par Harsanyi permet d'encapsuler toutes les informations disponibles qu'un joueur a au sujet d'un jeu. Cette notion ne comprend pas seulement l'état du monde mais aussi la croyance d'un joueur au sujet des autres joueurs et de leur "état introspectif". Ces questions sur les

liens entre les types et les états de croyance ainsi que la théorie des jeux épistémiques sont abordées dans [Brandenburger et Dekel, 1993, Fagin *et al.*, 1995, Aumann et Brandenburger, 1995, Battigalli et Bonanno, 1999, Nebel, 2000, Brandenburger, 2008, Dekel et Siniscalchi, 2015]

Nous pouvons définir formellement un jeu bayésien de la manière suivante :

Définition 22 (Jeux bayésien)**.** *Un jeu bayésien est un tuple* $BG = \langle S, \Omega, \Theta, P, u \rangle$ *pour lequel :*

- $S = \{1, ..., N\}$ *est l'ensemble des joueurs.*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des stratégies jointes.* Ω_n *est l'ensemble des stratégies du joueur* n*.*
- $\Theta = \Theta_1 \times ... \times \Theta_N$ *est l'ensemble des types joints.* Θ_n *est l'ensemble des types du joueur* n*.*
- $P : \Theta \to [0, 1]$ *est une distribution de probabilité sur les types joints.*
- $u = (u_n)$ *où* $u_n : \Omega \times \Theta \to \mathbb{R}$ *est la fonction d'utilité d'un joueur* n*.* $u_n(\omega, \theta)$ *est l'utilité du joueur* n *pour l'action jointe* ω*, dans le jeu correspondant au type* θ*.*

La distribution de probabilité a priori P est commune à tous les joueurs. L'information que les joueurs peuvent avoir sur le "vrai" monde correspond à un type $\theta \in \Theta$ mais elle n'est pas commune à tous les joueurs. Un joueur n n'a pas une connaissance complète de θ, il a seulement connaissance de θ_n. La notation $P(\theta_{-n}|\theta_n) = P(\theta|\theta_n)$ correspond à la connaissance d'un joueur n de type θ_n sur le jeu une fois qu'il a connaissance de θ_n.

Pour un ensemble de N joueurs, qui ont chacun m stratégies et t types, représenter un jeu bayésien demande une table de t^N éléments pour la probabilité jointe sur les types et une table de $N \times m^N \times t^N$ éléments pour les utilités.

Exemple 23 (Dilemme du Shérif)**.** *Prenons un exemple d'un jeu bayésien à 2 joueurs où l'un des joueurs a* 2 *types. Un shérif (joueur* 1*) fait face à un suspect armé (joueur* 2*). Le shérif et le suspect ont le choix entre 2 actions, tirer (*T*) ou ne pas tirer (*PT*). Le suspect peut être un criminel ou bien un civil, ce qui correspond à ses types, donc nous avons un jeu bayésien* $BG = \langle S = \{1, 2\}, \Omega = (T, PT) \times (T, PT), \Theta = (Sh) \times (Cri, Civ), P, u \rangle$ *où les utilités sont représentées dans la figure 3.2 et les probabilités sont* $P(Sh, Cri) = p$ *et* $P(Sh, Civ) = 1 - p$*.*

		2 : T	2 : PT
1 : T		$0, 0$	$-1, -2$
1 : PT		$-2, 2$	$1, -1$

$\theta_2 = Cri$

		2 : T	2 : PT
1 : T		$-1, -3$	$-1, -2$
1 : PT		$-2, -1$	$0, 0$

$\theta_2 = Civ$

FIGURE 3.2 – Table d'utilité en fonction du type du joueur 2

Dans un jeu bayésien, la stratégie du joueur n dépend de l'information $\theta_n \in \Theta_n$ qu'il a reçue.

Dans un jeu à information complète, un joueur joue selon une stratégie pure ou mixte. Ces notions peuvent être étendues aux jeux bayésiens.

Dans un jeu bayésien, comme dans un jeu sous forme normale, une stratégie mixte est une distribution de probabilité sur les actions des joueurs. La différence avec un jeu à information complète est qu'une stratégie mixte différente sera associée à chacun

des types de chaque joueur. Cela signifie qu'en fonction de son type, le joueur jouera selon une distribution de probabilité qui ne sera pas la même.

Une stratégie mixte est un tuple $\sigma = (\sigma_1, ..., \sigma_N)$ où σ_n est une fonction associant une distribution de probabilité sur Ω_n à chaque type Θ_n. On notera $\sigma_n(\omega_n|\theta_n)$ la probabilité qu'a le joueur n de jouer ω_n lorsqu'il est de type θ_n [1].

Une stratégie jointe pure est un tuple $\sigma = (\sigma_1, ..., \sigma_N)$ où σ_n est une fonction associant une action $\omega_n \in \Omega_n$ à chaque type $\theta_n \in \Theta_n$.

Dans le cas d'une stratégie pure, la notation $\sigma_n(\theta_n)$ désigne la stratégie jouée par n lorsqu'il reçoit l'information θ_n.

Définition 23 (Utilité espérée)**.** *Le calcul de l'utilité espérée $EU_n(\sigma|\theta_n)$ dans un jeu bayésien pour un joueur n, une stratégie mixte σ et un type θ_n est :*

$$EU_n(\sigma|\theta_n) = \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega \in \Omega} \left(\prod_{\nu=1}^{N} \sigma_\nu(\omega_\nu|\theta_\nu) \right) u_n(\omega, \theta)$$

Notons que même lorsqu'une stratégie pure est utilisée, l'utilité espérée sera calculée (espérance sur les types des autres joueurs) car l'utilité dépendra toujours de la distribution de probabilité sur les types joints.

3.4 Équilibre de Nash dans un jeu bayésien

Dans un jeu bayésien, comme dans un jeu classique, un équilibre de Nash consiste en une stratégie de laquelle aucun joueur ne peut unilatéralement dévier pour améliorer son utilité. Dans le cadre bayésien, cette condition doit être respectée pour tous les types de chaque joueur : pour un type donné, un joueur ne pourra pas améliorer son utilité espérée en changeant la stratégie associée à ce type.

Comme pour les jeux à information complète, on peut définir les meilleures réponses de la manière suivante :

Définition 24 (Meilleure réponse dans un jeu bayésien)**.** *Dans un jeu bayésien, pour une stratégie jointe σ_{-n} des joueurs différents de n, la meilleure réponse du joueur n de type θ_n à cette stratégie est une stratégie pure ω_n telle que*

$$EU_n(\omega_n, \sigma_{-n}|\theta_n) = max_{\omega'_n \in \Omega_n} EU_n(\omega'_n, \sigma_{-n}|\theta_n)$$

Il est aussi possible d'étendre la notion d'équilibre de Nash mixte aux jeux bayésiens.

Définition 25 (Équilibre de Nash mixte dans un jeu bayésien)**.** *Un équilibre de Nash mixte dans un jeu bayésien est une stratégie mixte σ^* telle que pour tous les joueurs $n \in S$ et tous les types $\theta_n \in \Theta_n$ nous avons :*

$$EU_n(\sigma_n^*, \sigma_{-n}^*|\theta_n) \geqslant EU_n(\sigma_n, \sigma_{-n}^*|\theta_n), n \in S, \theta_n \in \Theta_n \forall \sigma_n$$

Si σ^* est une stratégie pure alors l'équilibre de Nash bayésien est pur.

De manière similaire au cas à information complète nous pouvons vérifier que σ^* est un équilibre de Nash en considérant toutes les stratégies pures possibles :

1. Nous notons une stratégie mixte d'un joueur n de type θ_n en utilisant $\sigma_n(.|\theta_n)$ pour la différencier de la stratégie mixte $\xi_n(.)$ d'un joueur n dans un jeu à information complète. Cette différence sera utile lorsqu'il s'agira de présenter les transformations entre les jeux à information complète et incomplète.

Proposition 4. *σ^* est un équilibre de Nash si et seulement si :*

$$EU_n(\sigma_n^*, \sigma_{-n}^*|\theta_n) \geqslant EU_n(\omega_n, \sigma_{-n}^*|\theta_n), n \in S, \theta_n \in \Theta_n, \forall \omega_n \in \Omega_n$$

Exemple 24 (Dilemme du shérif)**.** *Reprenons l'exemple 23 avec $P(Cri) = P(Sh, Cri) = \frac{1}{2}$ et $P(Civ) = P(Sh, Civ) = \frac{1}{2}$, la stratégie bayésienne pure $\sigma_1(Sh) = S$, $\sigma_2(Cri) = S$ et $\sigma_2(Civ) = NS$.*

Avec cette stratégie, les utilités sont les suivantes $EU_1(\sigma|Sh) = -\frac{1}{2}$, $EU_2(\sigma|Cri) = 0$ et $EU_2(\sigma|Civ) = -2$

Si l'on change la stratégie des joueurs de manière unilatérale, dans le cas du joueur "Shérif", en utilisant $\sigma_1(Sh) = NS$, nous avons $EU_1(\sigma|Sh) = -1$ qui n'améliore pas son utilité. Si nous modifions maintenant celle du joueur 2 de type "Criminel" pour que $\sigma_2(Cri) = NS$ nous avons $EU_2(\sigma|Cri) = -2$ qui n'est, une fois de plus, pas une amélioration. Et finalement pour le joueur 2 de type "Civil", avec $\sigma_2(Civ) = S$ nous avons $EU_2(\sigma|Civ) = -3$ qui n'est encore une fois pas une amélioration. Donc la stratégie pure σ est bien un équilibre de Nash bayésien pur.

Dans la section suivante, nous allons montrer que tout jeu bayésien peut être ramené à un jeu sous forme normale équivalent. Il existe un équilibre de Nash mixte pour tout jeu normal, donc pour tout jeu bayésien il existe également un équilibre de Nash mixte [Harsanyi, 1968b].

3.5 Équivalence d'un jeu bayésien avec un jeu à information complète

Il est possible de transformer un jeu bayésien en un jeu sous forme normale à information complète [Harsanyi, 1967]. Cette transformation consiste à représenter les joueurs et leurs types par un ensemble de joueurs prenant la forme de couple "joueur, type". Cette formulation a initialement été décrite par Selten et Harsanyi dans un modèle appelé les "Selten games" (aussi appelé "posterior lottery model").

En utilisant cette décomposition, il est possible de reformuler le jeu bayésien sous forme d'un jeu sans incertitude (un jeu à information complète sous forme normale).

Proposition 5 (Jeu sous forme normale équivalent à un jeu Bayésien)**.** *Pour un jeu bayésien $BG = \langle S, \Omega, \Theta, P, u\rangle$ il existe un jeu sous forme normale équivalent $\hat{G} = \langle \hat{S}, \hat{\Omega}, \hat{u}\rangle$ tel que :*

- *$\hat{S} = \{(n, \theta_n)\}_{n\in S, \theta_n \in \Theta_n}$ est l'ensemble des joueurs de $\hat{G}$ où le joueur (n, θ_n) est issue du couple joueur n / type θ_n.*
- *$\hat{\Omega} = \times_{(n,\theta_n)\in\hat{S}}\Omega_n$ est l'ensemble des stratégies jointes de $\hat{G}$.*
- *$\hat{u} = \{\hat{u}_{(n,\theta_n)} : \hat{\Omega} \to \mathbb{R}\}_{(n,\theta_n)\in\hat{S}}$ est l'ensemble des fonctions d'utilité de $\hat{G}$ où $\hat{u}_{(n,\theta_n)}(\hat{\omega})$ est l'utilité du joueur (n, θ_n) lorsque $\hat{\omega}$ est jouée. Celle-ci est définie par :*

$$\hat{u}_{n,\theta_n}(\hat{\omega}) = \sum_{\theta_{-n}\in\Theta_{-n}} P(\theta_{-n}|\theta_n)u_n(\hat{\omega}(\theta), \theta)$$

où $\hat{\omega}(\theta) = (\hat{\omega}_{\nu,\theta_\nu})_{\nu\in S}$

Harsanyi a montré que cette transformation est cohérente [Harsanyi, 1967]. C'est à dire que :

- Les stratégies jointes pures/mixtes du jeu bayésien et de son jeu transformé en forme normale sont en bijection.

— Les utilités espérées de deux stratégies pures/mixtes équivalentes (dans le jeu bayésien et le jeu transformé) sont égales.

Une stratégie jointe $\hat{\omega} \in \hat{\Omega}$ du jeu transformé sera équivalente à une stratégie σ du jeu bayésien si pour tout $\theta \in \Theta$, $\hat{\omega}(\theta) = (\hat{\omega}_{\nu,\theta_\nu})_{\nu \in S}$. Nous avons donc $\hat{\omega}(\theta) \in \Omega$. Pour une action jointe $\hat{\omega} \in \hat{\Omega}$ et un type joint θ on définit $\hat{\omega}(-\theta)$ par : $\hat{\omega} = (\hat{\omega}(\theta), \hat{\omega}(-\theta))$

On écrit $\hat{\xi}$ la stratégie mixte du jeu $\hat{G}$ où $\hat{\xi}_{n,\theta_n}(\hat{\omega}_{n,\theta_n})$ est la probabilité que le joueur type (n, θ_n) joue la stratégie $\hat{\omega}_{n,\theta_n}$.

Pour une stratégie mixte $\hat{\xi}$ nous avons donc l'utilité espérée suivante :

$$\hat{EU}_{n,\theta_n}(\hat{\xi}) = \sum_{\hat{\omega} \in \hat{\Omega}} \left(\prod_{(n,\theta_n) \in \hat{S}} \hat{\xi}_{n,\theta_n}(\hat{\omega}_{n,\theta_n}) \right) \hat{u}_{n,\theta_n}(\hat{\omega})$$

Il est possible de montrer l'égalité entre les utilités des stratégies mixtes du jeu BG et celles du jeu $\hat{G}$ équivalent. Pour toute stratégie σ, il existe une stratégie équivalente $\hat{\xi}$ le définie par

$$\hat{\xi}_{(n,\theta_n)}(\hat{\omega}_{(n,\theta_n)}) = \sigma_n(\hat{\omega}_{(n,\theta_n)}|\theta_n)$$

Proposition 6 (Égalité des utilités)**.** *Pour une stratégie mixte σ dans jeu bayésien et sa stratégie équivalente $\hat{\xi}$ dans le jeu sous forme normale équivalent, nous avons :*

$$EU_n(\sigma, \theta) = \hat{EU}_{(n,\theta_n)}(\hat{\xi})$$

Cette égalité est souvent utilisée lorsqu'il s'agit de trouver un équilibre de Nash pour un jeu bayésien. En effet, l'équilibre de Nash du jeu sous forme normale équivalent peut être transformé en un équilibre de Nash bayésien du jeu initial.

Corollaire 1 (Équivalence des équilibres de Nash)**.** *σ^* est un équilibre de Nash mixte du jeu bayésien BG si et seulement si la stratégie mixte $\hat{\xi}^*$ est un équilibre de Nash mixte du jeu sous forme normale $\hat{G}$*

Après la transformation il n'est plus nécessaire de représenter les distributions de probabilité sur les types joints. Pour un jeu bayésien avec N joueurs qui ont chacun m actions et t types, afin de représenter le jeu, l'espace demandé est dorénavant $O(N \times t \times m^{N \times t})$.

Exemple 25 (Dilemme du Shérif)**.** *En reprenant l'exemple précédent, supposons que nous avons une distribution de probabilité $P(Cri) = p$ et $P(Civ) = 1 - p$. Le jeu sous forme normale équivalent est :*

— $\hat{S} = \{sh, cri, civ\}$.
— $\hat{\Omega} = \{\hat{\Omega}_{sh} \times \hat{\Omega}_{cri} \times \hat{\Omega}_{civ}\}$ *où* $\hat{\Omega}_{sh} = \hat{\Omega}_{cri} = \hat{\Omega}_{civ} = \{S, NS\}$
— $\hat{u} = \{\hat{u}_{sh}, \hat{u}_{cri}, \hat{u}_{civ}\}$ *sont représentées dans la figure 3.3*

$\hat{\omega}_{sh}$	$\hat{\omega}_{cri}$	$\hat{\omega}_{civ}$	$\hat{u}_{sh}$	$\hat{u}_{cri}$	$\hat{u}_{civ}$
S	S	S	$p-1$	0	-3
S	S	NS	$p-1$	0	-2
S	NS	S	-1	-2	-3
S	NS	NS	-1	-2	-2
NS	S	S	-2	2	-1
NS	S	NS	$-2p$	2	0
NS	NS	S	$3p-2$	-1	-1
NS	NS	NS	p	-1	0

FIGURE 3.3 – Table d'utilité du jeu sous la forme normale équivalent au jeu du dilemme du shérif

3.6 Équivalence d'un jeu bayésien bimatriciel avec un jeu polymatriciel

Peu d'algorithmes permettent de résoudre des jeux bayésiens sans les transformer en jeux sous forme normale [Ceppi *et al.*, 2009]. Mais Howson et Rosenthal ont montré qu'il était possible de transformer un jeu bayésien bimatriciel en un jeu polymatriciel à information complète équivalent [Howson et Rosenthal, 1974]. Ceci amène la possibilité d'appliquer l'algorithme [Howson, 1972] pour les résoudre. Dans le chapitre 4 nous étendrons le résultat de Howson et Rosenthal à n'importe quel type de jeu bayésien sous forme normale, en le transformant en jeu hypergraphique.

Cette transformation, de manière similaire à la transformation en forme normale, consiste à décomposer les joueurs de manière à avoir un ensemble de couples "joueur, type". Pour chaque type joint possible, un jeu local sera créé. Pour un type joint (θ_1, θ_2), il y aura un jeu local entre les joueurs $(1, \theta_1)$ et $(2, \theta_2)$.

Définition 26 (Jeu polymatriciel équivalent)**.** *Pour un jeu bayésien à 2 joueurs* $BG = \langle S = \{1,2\}, \Omega, \Theta, P, u\rangle$ *il existe un jeu polymatriciel équivalent* $\hat{PG} = \langle \hat{S}, \hat{\Omega}, \hat{E}, \hat{u}\rangle$ *défini par :*

- $\hat{S} = \{(n, \theta_n)\}_{n\in\{1,2\},\theta_n\in\Theta_n}$
- $\hat{\Omega} = \times_{\hat{n}\in\hat{S}}\Omega_{\hat{n}}$
- $\hat{E} = \{\{(1,\theta_1),(2,\theta_2)\}\}_{\theta\in\Theta}$
- $\hat{u} = \{u^{(1,\theta_1),(2,\theta_2)}_{(n,\theta_n)} : \Omega \to \mathbb{R}\}_{n\in\{1,2\}}$

Pour un jeu local dans ce jeu polymatriciel, l'utilité est obtenue en effectuant le calcul suivant

$$u^{(1,\theta_1),(2,\theta_2)}_{(n,\theta_n)}(\hat{\omega}_{(1,\theta_1),(2,\theta_2)}) = P(\theta_{-n}|\theta_n).u_n(\hat{\omega}_{(1,\theta_1),(2,\theta_2)}, \theta)$$

Étant donné que nous avons deux joueurs et que l'ensemble des jeux locaux correspond à toutes les combinaisons de types possibles entre ces deux joueurs, lorsque nous représentons le jeu polymatriciel sous la forme d'un graphe, nous avons un graphe biparti. Par exemple, si nous avons un jeu à deux joueurs où le joueur 1 a 3 types et où le joueur 2 en a 5, le graphe jeu polymatriciel est semblable à celui de la figure 3.4.

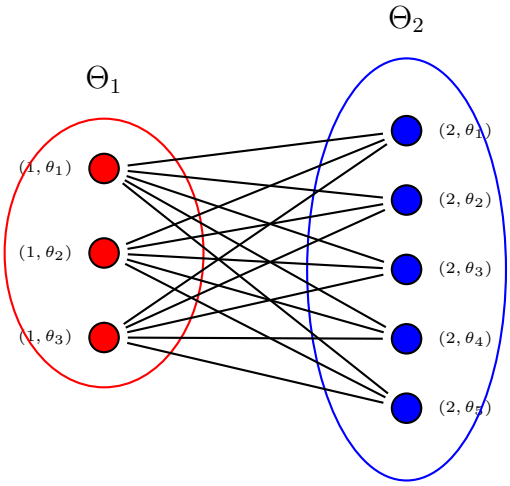


FIGURE 3.4 – Graphe du jeu polymatriciel équivalent à un jeu bayésien à 2 joueurs où le joueur 1 a 3 types et le joueur 2 a 5 types.

Pour une stratégie mixte σ dans jeu bayésien bimatriciel et sa stratégie équivalente $\hat{\xi}$ dans sa forme polymatriciel équivalente [Howson et Rosenthal, 1974], nous avons :

$$EU_n(\sigma, \theta_n) = \hat{EU}_{(n,\theta_n)}(\hat{\xi})$$

Une démonstration d'un résultat plus général que cette équivalence sera proposée dans le chapitre 4. Avec cette représentation, pour un jeu où chacun des deux joueurs a t types et m actions cela revient à avoir $2 \times t$ joueurs avec t^2 jeux locaux. Donc pour représenter ces jeux, il faut $2 \times t^2$ tables d'utilités avec m^2 entrées chacune.

Exemple 26 (Dilemme du Sherif)**.** *En reprenant une fois de plus l'exemple 23, mais cette fois en le transformant en un jeu sous forme polymatricielle, nous obtenons le jeu suivant :*

- $\hat{S} = \{Sh, Cri, Civ\}$ *l'ensemble des joueurs (en simplifiant la notation a partir de* $\{(1, Sh), (2, Cri), (2, Civ)\}$
- $\hat{\Omega} = \Omega_1 \times \Omega_2 \times \Omega_2$
- $\hat{E} = \{\{Sh, Cri\}, \{Sh, Civ\}\}$
- $\hat{u} = \{u_{Sh}^{(Sh,Cri)}, u_{Cri}^{(Sh,Cri)}, u_{Sh}^{(Sh,Civ)}, u_{Civ}^{(Sh,Civ)}\}$

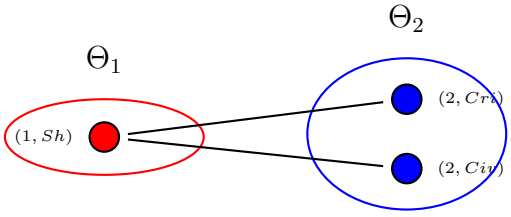


FIGURE 3.5 – Graphe du jeu polymatriciel équivalent au jeu de l'exemple 23

Cri

Sh	S	NS
S	$0, 0$	$-p, -2$
NS	$-2p, 2$	$p, -1$

Civ

Sh	S	NS
S	$p-1, -3$	$p-1, -2$
NS	$2p-2, -1$	$0, 0$

FIGURE 3.6 – Tables d'utilités du jeu polymatriciel équivalent au dilemme du shérif.

3.7 Jeux Bayésiens succincts

Comme pour les jeux à information complète, des formalismes de représentation succincte ont été proposés pour jeux bayésiens. Cependant, contrairement à leurs équivalents à information complète, il existe actuellement peu de cadres pour leur représentation succincte.

Jeux graphiques à information incomplète

Une extension des jeux graphiques acycliques (sous forme d'arbre) aux jeux bayésiens a été proposée par [Singh *et al.*, 2004]. Deux représentations des types sont proposées, l'une utilise un ensemble discret pour représenter les types de chaque joueur, l'autre propose d'utiliser un ensemble continu. Pour le premier de ces formalismes, un algorithme de propagation de message se basant sur celui défini par [Kearns *et al.*, 2001] est proposé pour calculer un équilibre de Nash bayésien approché. Un algorithme défini de manière abstraite a également été proposé, qui calcule un équilibre de Nash bayésien approché pour un espace de type continu.

Bayesian Action Graph Games

L'une des représentations existantes sont les Bayesian Action Graph Games (BAGG) [Jiang et Leyton-Brown, 2010]. Comme leur nom le sous-entend, il s'agit d'une extension bayésienne des Action Graph games décrits précédemment (sous-section 2.5).

A la différence de la forme normale bayésienne, la distribution de probabilité du jeu est représentée sous la forme d'un réseau bayésien ayant N variables $\theta_1, ..., \theta_N$. De plus on considère que pour plusieurs types différents d'un joueur ($\theta_n, \theta'_n \in \Theta_n$) nous pouvons avoir des ensembles d'actions différentes ($|\Omega_{n,\theta_n}| \neq |\Omega_{n,\theta'_n}|$).

Il est possible de représenter tout jeu bayésien sous la forme d'un BAGG en stockant le même nombre de valeurs d'utilités. De plus ce type de formulation permet de représenter des cadres où nous avons des indépendances entre les actions et les types des joueurs.

Les extensions définies pour les AGG [Jiang *et al.*, 2011], notamment les "function node" sont applicables sur les BAGG. Toutefois, aucun algorithme de calcul d'équilibre de Nash pour les BAGG n'a été proposé.

3.8 Complexité des jeux bayésiens

Il n'y a pas beaucoup de travaux s'intéressant à la question de la complexité des problèmes de recherche et de calcul des équilibres de Nash dans le cas bayésien.

Des travaux existent sur le problème de recherche de l'existence d'un équilibre de Nash pur. Ce problème a été démontré comme étant NP-complet pour les jeux bayésiens [Conitzer et Sandholm, 2008]. Cependant aucune démonstration de l'appartenance du *problème NASH* à PPAD ou à une autre classe n'existe, dans le cas bayésien.

Étant donné que pour un jeu sous forme normale, le *problème NASH* est PPAD-complet [Daskalakis *et al.*, 2009] et qu'un jeu de ce type peut être ramené à un jeu bayésien où chaque joueur dispose d'un seul type, alors dans le cas le plus facile le *problème NASH* dans un jeu bayésien est PPAD-complet ce qui signifie que ce problème est PPAD-difficile [Conitzer et Sandholm, 2008].

Dans le travail présenté dans cette thèse, nous allons démontrer formellement un nouveau résultat de complexité sur les jeux bayésiens en utilisant le nouveau modèle que nous avons défini.

3.9 Résolution des jeux bayésiens

Il existe peu d'algorithmes, pour résoudre les jeux bayésiens. En général, lorsqu'il s'agit de résoudre un jeu bayésien, la méthode utilisée consiste à prendre ce jeu et récupérer la forme normale équivalente pour ensuite utiliser un algorithme pour cette forme normale. Cependant quelques travaux se sont intéressés à des méthodes n'utilisant pas de transformation en jeu sous forme normale.

[Ceppi *et al.*, 2009] proposent une version de l'algorithme d'énumération de support pour les jeux bayésien (sans transformation en jeux sous forme normale). Dans ce même papier il est montré qu'il est aussi possible d'étendre Lemke Howson [Lemke et Howson, 1964] et la méthode mixed integer programming [Sandholm *et al.*, 2005] aux jeux bayésiens à 2 joueurs sans avoir à passer par une transformation.

Certains papiers s'intéressent à des algorithme pour des jeux bayésien particuliers comme [Cai et Wurman, 2005] avec une méthode de Monte Carlo pour les enchères séquentielles.

Pour les représentations succinctes, [Singh *et al.*, 2004] s'intéressent à la recherche d'un equilibre de Nash bayésien pour les représentations graphiques acycliques des jeux bayésiens avec un algorithme de propagation similaire à TreeNash.

3.10 Conclusion

Les jeux bayésiens, utilisés pour représenter les jeux à information incomplète ont été présentés dans ce chapitre. Ils servent à représenter les situations où les joueurs ne disposent pas de toutes les informations sur le jeu comme, par exemple, les utilités de leurs adversaires, ou l'état du jeu dans lequel ils vont jouer ...

Les liens avec les représentations des jeux à information complète (chapitres 1 et 2) ont aussi été présentées.

Dans le chapitre 4, une description de ma première contribution sera présentée : l'extension du résultat de Howson et Rosenthal au jeux bayésiens à N joueurs. Je propose également un nouveau modèle, les *jeux bayésiens hypergraphiques* et la possibilité d'étendre le résultat de Howson et Rosenthal à ces jeux.

Il existe une multitude de types de jeux différents non présentés dans ces premiers chapitres utilisés pour modéliser d'autres types de jeux. Certain de ces jeux, comme les jeux leader follower, stackelberg security game et jeux stochastiques seront mentionnés dans le chapitre 6.

Deuxième partie

Contributions

Chapitre 4

Jeux bayésiens hypergraphiques

4.1 Introduction

Comme mentionné dans le chapitre 3, lorsque l'on veut représenter un problème sous la forme d'un jeu il n'est pas toujours possible de considérer que les joueurs disposent d'une information complète du problème. Ils peuvent ne disposer que d'une information partielle. Les jeux bayésiens, présentés dans le chapitre 3, sont une réponse à ce problème de représentation.

Cependant, l'une des difficultés avec les jeux bayésiens tels que proposés par [Harsanyi, 1967] est que la demande en espace peut devenir importante. Si pour un jeu avec N joueurs, chaque joueur a t types et d actions alors il est nécessaire de stocker N tables d'utilité de taille $t^N \times d^N$.

Un second problème intervient lorsqu'il s'agit de calculer un équilibre de Nash bayésien. Il n'y a que très peu d'algorithmes permettant de trouver un équilibre pour un jeu bayésien arbitraire. Il existe des approches indirectes qui exploitent la possibilité de transformer le jeu sous son jeu équivalent sous forme normale à information complète et d'utiliser des algorithmes pour les jeux à informations complète. Il y a deux manières de transformer un jeu bayésien en un jeu sous forme normale. La première, déjà présentée dans le chapitre 3 est la représentation "type agent". Chaque type d'un joueur deviendra un joueur dans le jeu sous forme normale, il y a donc N^t joueurs avec d actions. La seconde, pas encore présentée, est la "forme normale induite". Elle va consister à considérer que chaque action d'un joueur est en fait la combinaison des actions qu'il joue pour chacun de ses types. Le jeu résultant possède maintenant N joueurs avec d^t actions pures chacun.

Cependant chacune de ces transformations est exponentielle en espace.

Bien que la complexité de la recherche d'un équilibre de Nash dans un jeu bayésien n'ait pas encore été formellement établie, il est connu que le problème appartient à la classe PPAD (voir section 1.3) car il existe toujours un équilibre et il est possible de vérifier que cette solution est bien un équilibre en temps polynomial en la taille du jeu bayésien. Les algorithmes de résolution des jeux sous forme normale sont exponentiels en la taille de représentation du jeu, donc si la transformation du jeu bayésien sous sa forme normale à information complète est utilisée, alors les algorithmes sont doublement exponentiels en la taille de représentation du jeu bayésien ($t^N \times d^N \mapsto d^{Nt}$ ou d^{t^N}).

Cependant, il existe trois classes de jeux bayésiens pour lesquels ces remarques ne s'appliquent pas totalement.

La première de ces classes est celle des jeux bayésiens à deux joueurs. Comme men-

tionné dans le chapitre 3, il est possible de transformer un jeu bayésien à 2 joueurs en un jeu polymatriciel en temps polynomial [Howson et Rosenthal, 1974], par conséquent il est possible d'appliquer un algorithme comme celui de Howson [Howson, 1972] (décrit dans la section 2.7.1).

La deuxième de ces classes est celle des jeux graphiques acycliques bayésiens [Singh *et al.*, 2004]. C'est une extension de la représentation graphique [Kearns *et al.*, 2001] aux cas bayésiens. Un algorithme de propagation de message est utilisable afin de calculer un équilibre de Nash approché.

La troisième de ces classes de jeux est celle des BAGG [Jiang et Leyton-Brown, 2010] (Bayesian Action Graph Games, mentionnés dans la section 3.7) qui sont des représentations succinctes où la connaissance des joueurs sera représentée via un réseau bayésien. Cependant cette représentation amène à une augmentation de la complexité de calcul et aucun algorithme permettant de calculer un équilibre de Nash n'a été proposé (sauf à passer par une transformation en forme normale).

Dans ce chapitre, je présente plusieurs contributions liées aux jeux bayésiens. Tout d'abord je présente une généralisation du théorème de Howson et Rosenthal aux jeux avec N joueurs, ce qui permet de proposer de nouvelles méthodes de résolution pour les jeux bayésiens avec N joueurs en passant par des représentations succinctes plutôt que par des représentations sous forme normale.

Une autre de mes contributions a été de définir une nouvelle représentation succincte des jeux à information incomplète. Pour ce faire, j'ai défini une nouvelle représentation pour les jeux bayésiens, nettement moins coûteuse en espace, en incorporant les principes des jeux hypergraphiques (et polymatriciels) dans un contexte bayésien. Ce nouveau modèle de "Jeu Bayésien Hypergraphique", permet de représenter de manière plus concise les jeux bayésiens où les interactions sont locales. Par exemple, dans un jeu de coordination où la satisfaction d'un joueur est proportionnelle au nombre de ses "voisins" prenant la même décision, où les joueurs ont des préférences qui sont incertaines. L'exemple 27 décrit un jeu de ce type. Je l'utiliserai comme illustration dans ce chapitre.

Exemple 27 (Coordination d'abonnement internet)**.** *Je vais décrire un exemple de jeu qui sera repris plusieurs fois dans ce chapitre.*

L'exemple, inspiré de [Simon et Wojtczak, 2017, Amor et al.*, 2020], est un jeu de coordination entre des joueurs cherchant à choisir un fournisseur internet. Dans cette variation, on considère que la satisfaction d'un joueur dépendra du choix de ses voisins dans un "réseau social" représenté par un graphe $\langle S, E\rangle$. Plus le nombre de voisins ayant le même fournisseur que le joueur sera élevé, plus il sera satisfait. Dans cet exemple, je considère que le composant incertain est la réception par un joueur (ou l'absence de réception) d'une publicité envoyée par l'un des fournisseurs. La réception d'une publicité augmente la préférence du joueur l'ayant reçu pour le fournisseur et lui donne l'information suivante : "Si j'ai reçu une publicité, normalement mes voisins en ont aussi reçu une".*

Considérons que chaque joueur a 2 stratégies pures, $\Omega_n = \{B, C\}$, qui correspondent respectivement au choix des fournisseurs Berizon et C-Mobile. C'est le fournisseur Berizon qui envoie des publicités aux joueurs, mais les joueurs n'ont aucune connaissance de qui a reçu les publicités. $\Theta_n = \{R, \bar{R}\}$ sont respectivement "a reçu" et "n'a pas reçu" une publicité de Berizon.

Un joueur qui n'a pas reçu de publicité (un joueur de type $\bar{R}$) préfère C-Mobile :

- *Lorsqu'il joue C, son degré de satisfaction est le nombre de ses voisins qui jouent aussi C.*

— *Lorsqu'il joue B, son degré de satisfaction est égal au nombre ses voisins jouant B multiplié par un facteur* α, $0 < \alpha < 1$.

Un joueur qui reçoit une publicité (un joueur de type R) préfère Berizon :

— *Lorsqu'il joue C, son degré de satisfaction est toujours le nombre de ses voisins qui jouent aussi C*

— *Maintenant, lorsqu'il joue B, son degré de satisfaction est égal au nombre de ses voisins jouant B multiplié par un facteur* $\beta > 1$

Dans cet exemple, l'utilité d'un joueur n dépend de son type mais ne dépend pas de celui de ses voisins.

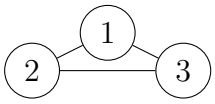


FIGURE 4.1 – Le graphe de voisinage d'un jeu de coordination complètement connexe à 3 joueurs.

Notons $nh(n)$ *l'ensemble des voisins du joueur n. La fonction d'utilité d'un joueur n est définie par :*

$$\begin{aligned} u_n(C.\omega_{-n}, \bar{R}.\theta_{-n}) &= |\{\nu \in nh(n), \omega_\nu = C\}| \\ u_n(C.\omega_{-n}, R.\theta_{-n}) &= |\{\nu \in nh(n), \omega_\nu = C\}| \\ u_n(B.\omega_{-n}, \bar{R}.\theta_{-n}) &= \alpha.|\{\nu \in nh(n), \omega_\nu = B\}| \\ u_n(B.\omega_{-n}, R.\theta_{-n}) &= \beta.|\{\nu \in nh(n), \omega_\nu = B\}| \end{aligned}$$

Dans cet exemple, la distribution de probabilité jointe sur les types est définie par : tous les joueurs reçoivent une publicité (probabilité 0.3), ou aucune publicité n'est envoyée par Berizon (probabilité 0.3), toutes les autres situations ont la même probabilité q. Donc pour $N \geqslant 2$ *joueurs,* $q = \frac{1-0.6}{2^N-2}$ *Par exemple si* $N = 2$ *:*
$p(RR) = p(\bar{R}\bar{R}) = 0.3$, $p(R\bar{R}) = p(\bar{R}R) = 0.2$ *;*
et si $N = 3$
$p(RRR) = p(\bar{R}\bar{R}\bar{R}) = 0.3$, $p(RR\bar{R}) = p(R\bar{R}\bar{R}) = \frac{0.4}{6} = \frac{1}{15}$.

Considérons une stratégie σ *qui est une stratégie pure* $\sigma_n(R) = B$, $\sigma_n(\bar{R}) = C$ *- les joueurs recevant une publicité de Berizon jouent B ceux qui n'en reçoivent pas jouent C. L'utilité espérée du joueur n pour* σ_n *est*

$$\begin{aligned} EU_n(\sigma|R) &= \Sigma_{\theta_{-n}} P(\theta_{-n}|R).\beta.|\{\nu \in nh(n), \theta_\nu = R\}|, \\ EU_n(\sigma|\bar{R}) &= \Sigma_{\theta_{-n}} P(\theta_{-n}|\bar{R}).|\{\nu \in nh(n), \theta_\nu = \bar{R}\}|. \end{aligned}$$

Dans le cas avec 3 joueurs, $P(RR|R) = \frac{P(RRR)}{P(R)} = \frac{0.3}{0.5} = 0.6$ *et* $P(R\bar{R}|R) = P(\bar{R}R|R) = \frac{P(R\bar{R}R)}{P(R)} = \frac{0.4}{6}\frac{1}{0.5} = \frac{2}{15}$.

Donc pour un graphe complet (Figure 4.1), on a :

$$\begin{aligned} EU_n(\sigma|R) &= P(RR|R).\beta.2 + P(R\bar{R}|R).\beta + P(\bar{R}R|R).\beta =, \frac{22}{15}\beta. \\ EU_n(\sigma|\bar{R}) &= P(\bar{R}\bar{R}|\bar{R}).2 + P(R\bar{R}|\bar{R}) + P(\bar{R}R|\bar{R}) = \frac{22}{15}. \end{aligned}$$

Si le joueur n change $\sigma_n(R) = B$ *en* $\sigma'_n(R) = C$, *il obtiendra* $EU_n(\sigma'_n.\sigma_{-n}|R) = \frac{22}{15} < \frac{22}{15}\beta$. *Et s'il change* $\sigma_n(\bar{R}) = C$ *en* $\sigma'_n(\bar{R}) = B$, *il aura* $EU_n(\sigma'_n.\sigma_{-n}|\bar{R}) = \frac{22}{15}\alpha < \frac{22}{15}$.

Donc σ *est un équilibre de Nash pur.*

4.2 Extension du théorème de Howson et Rosenthal aux jeux bayésiens à N joueurs

Nous nous sommes tout d'abord intéressé au théorème de Howson et Rosenthal [Howson et Rosenthal, 1974], montrant que tout jeu bayésien à 2 joueurs peut être transformé en un jeu polymatriciel. Cette transformation est décrite dans le chapitre 3. Le graphe du jeu polymatriciel est un graphe biparti où chaque groupe de noeuds correspond aux types d'un des deux joueurs du jeu bayésien.

Cette section propose une extension de ce théorème aux jeux bayésiens à N joueurs : il est possible de transformer un jeu bayésien avec N joueurs en un jeu à représentation succincte, qui sera un jeu hypergraphique. Le graphe de ce jeu est N-parti.

Définissons tout d'abord la transformation d'un jeu bayésien à N joueurs.

Définition 27 (Transformation d'un jeu bayésien à N joueurs en jeu hypergraphique)**.** *Soit $G = \langle S, \Omega, \Theta, P, u\rangle$ un jeu bayésien avec N joueurs. Le jeu hypergraphique équivalent $\tilde{G} = \langle \tilde{S}, \tilde{\Omega}, \tilde{E}, \tilde{u}\rangle$ est défini par :*

- *$\tilde{S} = \{(n, \theta_n), n \in S, \theta_n \in \Theta_n\}$ est l'ensemble des joueurs créé à partir de tous les couples (joueur, type) possibles.*
- *$\tilde{\Omega} = \bigotimes_{(n,\theta_n)} \Omega_n$ i.e. $\Omega_{(n,\theta_n)} = \Omega_n$ est l'ensemble des actions jointes.*
- *$\tilde{E} = \{\tilde{e}(\theta), \theta \in \Theta\}$ est l'ensemble des hyperarêtes où $\tilde{e}(\theta) = \{(n, \theta_n), \theta_n \in \theta, n \in S\}$*
- *$\tilde{u} = \{\tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}, \tilde{e}(\theta) \in \tilde{E}, (n, \theta_n) \in \tilde{e}(\theta)\}$ où $\tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}(\omega) = P(\theta_{-n}|\theta_n).u_n(\omega, \theta)$*

Comme pour la transformation d'un jeu bayésien sous sa forme normale à information complète et la transformation d'un jeu bayésien à 2 joueurs en un jeu polymatriciel à information complète vue précédemment, la transformation présentée ici utilise la forme "type agent", chaque type θ_n qu'un joueur n peut avoir deviendra un joueur individuel (n, θ_n).

Définition 28 (Transformation d'une stratégie)**.**
Pour une stratégie mixte σ de G, $\tilde{\sigma}$ est la stratégie de $\tilde{G}$ définie par :

$$\tilde{\sigma}_{(n,\theta_n)}(\omega_n) = \sigma_n(\omega_n|\theta_n), \forall \omega_n \in \Omega_n.$$

Proposition 7 (Equivalence avec la forme normale bayésienne)**.** *Pour tout jeu bayésien G, il existe un jeu hypergraphique (à information complète) $\tilde{G}$ tel que $\forall n \in S, \theta_n \in \Theta_n, \forall \sigma, EU_n(\sigma|\theta_n) = EU_{(n,\theta_n)}(\tilde{\sigma})$*

Preuve (Proposition 7)**.** *Soit un jeu bayesien G et $\tilde{G}$ son jeu hypergraphique équivalent comme décrit dans la définition 27. Puisque $\tilde{E} = \{e_\theta, \theta \in \Theta\}$, alors $\tilde{E}$ comprend une hyperarête par type joint.*

Fixons $n \in S, \theta_n \in \Theta_n$ et considérons un $\tilde{\sigma}$ quelconque. Alors, selon la définition d'un jeu hypergraphique $\tilde{G}$ nous avons :

$$\begin{aligned} EU_{(n,\theta_n)}(\tilde{\sigma}) &= \sum_{\tilde{\omega}\in\tilde{\Omega}} \tilde{\sigma}(\tilde{\omega}) \left(\sum_{\theta_{-n}\in\Theta_{-n}} \tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}(\tilde{\omega}) \right), \\ &= \sum_{\tilde{\omega}\in\tilde{\Omega}} \left(\prod_{(\nu,\theta_\nu)\in\tilde{S}} \tilde{\sigma}_{(\nu,\theta_\nu)}(\tilde{\omega}_{(\nu,\theta_\nu)}) \right) \left(\sum_{\theta_{-n}\in\Theta_{-n}} \tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}(\tilde{\omega}_{(1,\theta_1)}, \ldots, \tilde{\omega}_{(N,\theta_N)}) \right). \end{aligned}$$

Pour tout $\theta \in \Theta$, nous écrivons $\tilde{\omega}_{\tilde{e}(\theta)} = (\tilde{\omega}_{(1,\theta_1)}, \ldots, \tilde{\omega}_{(N,\theta_N)})$. Alors,après avoir éliminé par marginalisation, les variables libres dans $\tilde{\sigma}$:

$$EU_{(n,\theta_n)}(\tilde{\sigma}) = \sum_{\theta_{-n}} \sum_{\tilde{\omega}_{\tilde{e}(\theta)}} \left(\prod_{\nu \in S} \tilde{\sigma}_{(\nu,\theta_\nu)}(\tilde{\omega}_{(\nu,\theta_\nu)}) \right) \tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}(\tilde{\omega}_{\tilde{e}(\theta)}),$$

Selon la définition 27, nous avons :

$$\tilde{u}^{\tilde{e}(\theta)}_{(n,\theta_n)}(\tilde{\omega}_{\tilde{e}(\theta)}) = P(\theta_{-n}|\theta_n) u_n(\tilde{\omega}_{\tilde{e}(\theta)}, \theta).$$

et d'après la définition 28, en renommant les variables muettes $\tilde{\omega}_{(\nu,\theta_\nu)}$ en $\omega_\nu, \forall \nu \in S$ nous obtenons :

$$\begin{aligned} EU_{(n,\theta_n)}(\tilde{\sigma}) &= \sum_{\theta_{-n} \in \Theta_{-n}} \sum_{\omega \in \Omega} \left(\prod_{\nu \in S} \sigma_\nu(\omega_\nu|\theta_\nu) \right) P(\theta_{-n}|\theta_{-n}) u_n(\omega, \theta), \\ &= \sum_{\theta_{-n} \in \Theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega \in \Omega} \sigma(\omega|\theta) u_n(\omega, \theta), \\ &= EU_n(\sigma|\theta_n). \end{aligned}$$

Cette proposition généralise le théorème de Howson et Rosenthal aux jeux bayésiens à N joueurs. Notez que cette transformation est appliquée sur un jeu bayésien sous forme normale, le jeu hypergraphique (à information complète) obtenu possède $\Sigma_{n \in S}|\Theta_n|$ noeuds (joueurs) et $|\Theta|$ hyperarêtes (jeux locaux) de taille N.

Proposition 8 (Équivalence des équilibres de Nash)**.**
σ est un équilibre de Nash pur (resp. mixte) d'un jeu bayésien G si et seulement si $\tilde{\sigma}$ est un équilibre de Nash pur (resp. mixte) de $\tilde{G}$

Preuve (Proposition 8)**.** *C'est une conséquence directe de la proposition 7.*

Proposition 9 (Transformation en temps polynomial)**.** *Pour tout jeu bayésien G, $\tilde{G}$ peut être calculé en temps polynomial.*

Preuve (Proposition 9)**.** *Calculer $\tilde{u}$ demande de calculer $|\Omega|$ éléments des tables d'utilité $u^{\tilde{e}(\theta)}_{(n,\theta_n)}$ pour tout θ. Mais il est aussi nécessaire pour u de stocker les $u_n(\omega, \theta)$ pour tout θ et ω. Les deux représentations sont donc équivalentes en taille, ce qui prouve la proposition (les autres éléments de $\tilde{G}$ sont calculés en temps polynomial).*

Exemple 28. *Reprenons le jeu de coordination présenté précédemment pour 3 joueurs avec un graphe de voisinage complet. Ce jeu bayésien ayant 3 joueurs qui ont chacun 2 types, on construit un jeu hypergraphique avec 6 joueurs et 8 hyperarêtes (il y a 8 types joints possibles).*

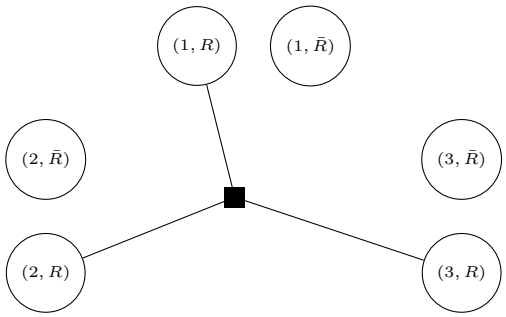


FIGURE 4.2 – Une hyperarête/jeu local du nouveau jeu hypergraphique à 6 joueurs équivalent au jeu bayésien à 3 joueurs de la figure 4.1. Cette arête correspond aux types $\theta = (R, R, R)$.

$\tilde{\omega}_{(1,R)}$	$\tilde{\omega}_{(2,R)}$	$\tilde{\omega}_{(3,R)}$	$\tilde{u}_{(1,R)}$	$\tilde{u}_{(2,R)}$	$\tilde{u}_{(3,R)}$
B	B	B	$\frac{18}{15}\beta$	$\frac{18}{15}\beta$	$\frac{18}{15}\beta$
B	B	C	$\frac{9}{15}\beta$	$\frac{9}{15}\beta$	0
B	C	B	$\frac{9}{15}\beta$	0	$\frac{9}{15}\beta$
B	C	C	0	$\frac{9}{15}$	$\frac{9}{15}$
C	B	B	0	$\frac{9}{15}\beta$	$\frac{9}{15}\beta$
C	B	C	$\frac{9}{15}$	0	$\frac{9}{15}$
C	C	B	$\frac{9}{15}$	$\frac{9}{15}$	0
C	C	C	$\frac{18}{15}$	$\frac{18}{15}$	$\frac{18}{15}$

FIGURE 4.3 – Matrice d'utilité du sous jeu issue du type joint (R, R, R) (hyperarête représentée dans la figure 4.2)

Par exemple pour le jeu local de la figure 4.2, pour le joueur $(1, R)$ *et une stratégie* $\tilde{\omega}$*, nous avons le calcul suivant :*

$$\tilde{u}_{(1,R)}^{((1,R)(2,R)(3,R))}(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,R)}, \tilde{\omega}_{(3,R)}) = P(RR|R)u_1(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,R)}, \tilde{\omega}_{(3,R)}, (RRR))$$

Étant donné que $P(RR|R) = \frac{9}{15}$ *alors nous avons :*

$$\tilde{u}_{(1,R)}^{((1,R)(2,R)(3,R))}(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,R)}, \tilde{\omega}_{(3,R)}) = \frac{9}{15}u_1(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,R)}, \tilde{\omega}_{(3,R)}, (RRR))$$

Donc si $\tilde{\omega}_{((1,R)(2,R)(3,R))} = (B, B, B)$ *alors :*

$$\tilde{u}_{(1,R)}^{((1,R)(2,R)(3,R))}(B, B, B) = \frac{9}{15} * 2\beta = \frac{18}{15}\beta$$

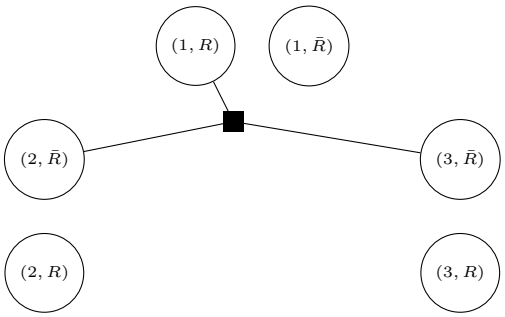


FIGURE 4.4 – Une hyperarête/jeu local du nouveau jeu hypergraphique à 6 joueurs équivalent au jeu bayésien à 3 joueurs de la figure 4.1. Cette arête correspond au type joint $\theta = (R, \bar{R}, \bar{R})$

$\tilde{\omega}_{(1,R)}$	$\tilde{\omega}_{(2,\bar{R})}$	$\tilde{\omega}_{(3,\bar{R})}$	$\tilde{u}_{(1,R)}$	$\tilde{u}_{(2,\bar{R})}$	$\tilde{u}_{(3,\bar{R})}$
B	B	B	$\frac{4}{15}\beta$	$\frac{4}{15}\alpha$	$\frac{4}{15}\alpha$
B	B	C	$\frac{2}{15}\beta$	$\frac{2}{15}\alpha$	0
B	C	B	$\frac{2}{15}\beta$	0	$\frac{2}{15}\alpha$
B	C	C	0	$\frac{2}{15}$	$\frac{2}{15}$
C	B	B	0	$\frac{2}{15}\alpha$	$\frac{2}{15}\alpha$
C	B	C	$\frac{2}{15}$	0	$\frac{2}{15}$
C	C	B	$\frac{2}{15}$	$\frac{2}{15}$	0
C	C	C	$\frac{4}{15}$	$\frac{4}{15}$	$\frac{4}{15}$

FIGURE 4.5 – Matrice d'utilité du sous jeu issue du type joint $(R, \bar{R}, \bar{R})$ (hyperarête représentée dans la figure 4.4)

Pour le jeu de la figure 4.4, étant donné que nous avons $P(\bar{R}\bar{R}|R) = \frac{2}{15}$ *alors :*

$$\tilde{u}_{(1,R)}^{((1,R)(2,\bar{R})(3,\bar{R}))}(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,\bar{R})}, \tilde{\omega}_{(3,\bar{R})}) = \frac{2}{15} u_1(\tilde{\omega}_{(1,R)}, \tilde{\omega}_{(2,\bar{R})}, \tilde{\omega}_{(3,\bar{R})}, (R\bar{R}\bar{R}))$$

4.3 Jeux bayésiens hypergraphiques

Nous décrivons maintenant un nouveau modèle, les jeux bayésiens hypergraphiques.

Définition 29 (Jeu bayésien hypergraphique)**.** *Un jeu bayésien hypergraphique ((H, B)-game) est un tuple* $G = \langle S, \Omega, \Theta, E, P, u \rangle$ *où :*

- $S = \{1, \ldots, N\}$ *est l'ensemble des* N *joueurs participant au jeu.*
- $\Omega = \Omega_1 \times ... \times \Omega_N$ *est l'ensemble des stratégies jointes.* Ω_n *est l'ensemble des stratégies du joueur* n*.*
- $\Theta = \Theta_1 \times ... \times \Theta_N$ *est l'ensemble des types joints.* Θ_n *est l'ensemble des types du joueur* n*.*
- E *est un ensemble de sous ensembles de* S *correspondant à un ensemble d'hyperarêtes (ou à un ensemble de jeux bayésiens locaux).*
- $P = \{P_e\}_{e \in E}$ *un ensemble de distributions de probabilité locales sur les types joints locaux,* $P_e : \Theta_e \mapsto [0, 1]$
- $u = \{u_n^e\}_{e \in E, n \in e}$ *est l'ensemble des tables d'utilité locales,* u_n^e *est la table d'utilité du joueur* n *dans le jeu local* e *où* $u_n^e : \Omega_e \times \Theta_e \mapsto \mathbb{R}$

L'utilité globale d'un joueur n *pour une action* ω *selon une configuration de type* θ *est définie par :*

$$u_n(\omega, \theta) = \sum_{e \in E, i \in e} u_n^e(\omega_e, \theta_e).$$

Quand $|e| \leqslant 2, \forall e \in E$*, on dit que le jeu* G *est un jeu bayésien polymatriciel*

Par la suite, $G_e = \langle e, \Omega_e, \Theta_e, P_e, u^e \rangle$ désignera le jeu local du jeu G pour l'hyperarête e. C'est un jeu bayésien avec $|e|$ joueurs.

La définition 29 capture, en tant que cas particuliers, les jeux hypergraphiques et polymatriciels à information complète lorsque $|\Theta| = 1$ et les jeux bayésiens sous forme normale lorsqu'il y a une seul hyperarête e avec $|e| = N$.

Notons que, comme pour le cas hypergraphique à information complète, s'il y a plusieurs hyperarêtes mais qu'une d'entre elles est de taille N il n'y a pas d'intérêt à représenter le jeu sous la forme d'un jeu bayésien hypergraphique.

Le condition de "connaissance commune" [1] est moins "naturelle" dans le contexte présenté ici d'une série de jeux bayésiens locaux que dans le cas des jeux bayésiens sous forme normale. En effet, il n'y a pas de raison pour laquelle le type d'un joueur devrait l'informer du type des joueurs avec lesquels il ne joue dans aucun des jeux locaux auxquels il participe. Par conséquent, on peut réduire la condition de "connaissance commune" utilisée dans les (H, B)-games à une simple condition de cohérence sur les probabilités locales. C'est-à-dire que la connaissance des joueurs est cohérente avec une distribution de probabilité a priori jointe (inconnue) sur les types :

Hypothèse 1 (Cohérences des connaissances)**.** *Il existe une distribution de probabilité jointe P sur Θ telle que :*

$$P_e(\theta_e) = \sum_{\theta_{-e}} P(\theta_e.\theta_{-e}), \forall e \in E, \theta_e \in \Theta_e.$$

Un jeu bayésien hypergraphique est donc équivalent à un jeu bayésien sous forme normale $\langle S, \Omega, \Theta, P, \{u_n\}_{n\in S}\rangle$ où u_n est donnée par la définition 29. Par conséquent, on peut utiliser la définition générale du calcul de l'utilité espérée pour une stratégie mixte dans un jeu bayésien (définition 23 vue dans le chapitre 3) :

$$EU_n(\sigma|\theta_n) = \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega\in\Omega} \left(\prod_{\nu=1}^{N} \sigma_\nu(\omega_\nu|\theta_\nu) \right) u_n(\omega, \theta)$$

Nous reprenons les notations du chapitre précédent, avec σ une stratégie mixte et σ_n la stratégie mixte du joueur n.

Notons que le calcul de l'utilité espérée EU_n du joueur n dépend de la distribution de probabilité P, qui porte sur tous les joueurs et n'est pas nécessairement totalement connue par le joueur n. En fait, seule sa marginalisation aux jeux locaux auxquels n participe est censée être connue. Néanmoins, on peut montrer le résultat suivant :

Proposition 10 (Décomposition de l'utilité espérée)**.** *Pour σ une stratégie mixte dans un jeu bayésien hypergraphique G :*

$$EU_n(\sigma|\theta_n) = \sum_{e,n\in e} EU_n^e(\sigma_e|\theta_n), \forall n \in S, \theta_n \in \Theta_n,$$

où $EU_n^e(\sigma_e|\theta_n)$ est l'utilité espérée du joueur n pour la stratégie mixte jointe locale σ_e dans le jeu local bayésien $G^e = \langle e, \Omega_e, \Theta_e, P_e, \{u_n^e, n \in e\}\rangle$.

Preuve (Proposition 10)**.** *Rappel de l'utilité espérée d'un jeu bayésien(définition 23) :*

$$EU_n(\sigma|\theta_n) = \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega\in\Omega} \left(\prod_{\nu=1}^{N} \sigma_\nu(\omega_\nu|\theta_\nu) \right) u_n(\omega, \theta)$$

Rappelons nous que, selon la définition 29, $u_n(\omega, \theta) = \sum_{e\in E, n\in e} u_n^e(\omega_e, \theta_e)$, où la somme est effectuée sur toutes les hyperarêtes contenant le noeud (joueur) n. Alors, si l'expression de $u_n(\omega, \theta)$ dans la définition de $EU_n(\sigma|\theta_n)$ est remplacée, nous obtenons :

$$EU_n(\sigma|\theta_n) = \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \cdot \sum_{\omega\in\Omega} \left(\sum_{e,n\in e} u_n^e(\omega_e, \theta_e) \right) \sigma(\omega|\theta).$$

1. C'est-à-dire que les joueurs ont une connaissance a priori sur tous les types de tous les joueurs. Elle est capturée par la distribution de probabilité P sur les types joints dans un jeu bayésien.

Alors, en utilisant la linéarité de l'opérateur d'espérance :

$$EU_n(\sigma|\theta_n) = \sum_{e,n\in e} \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega} u_n^e(\omega_e, \theta_e)\sigma(\omega|\theta).$$

et avec l'élimination par marginalisation des variables ω_{-e} dans $\sigma(\omega|\theta) = \sigma_e(\omega_e|\theta_e)\sigma_{-e}(\omega_{-e}|\theta_{-e})$:

$$EU_n(\sigma|\theta_n) = \sum_{e,n\in e} \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega_e} u_n^e(\omega_e, \theta_e)\sigma_e(\omega_e|\theta_e).$$

Finalement, en éliminant par marginalisation cette fois les variables θ_{-e} dans $P(\theta_{-n}|\theta_n) = P(\theta_{e-n}|\theta_n)P(\theta_{-e}|\theta_e)$ (qui est une égalité résultant du théorème de Bayes)[2]*, nous obtenons :*

$$\begin{aligned} EU_n(\sigma|\theta_n) &= \sum_{e,n\in e} \sum_{\theta_{e-n}} P(\theta_{e-n}|\theta_n) \sum_{\omega_e} u_n^e(\omega_e, \theta_e)\sigma_e(\omega_e|\theta_e), \\ &= \sum_{e,n\in e} EU_n^e\left(\sigma_e \mid \theta_n\right). \end{aligned}$$

Par conséquent, une connaissance des distributions de probabilité locales sur les types est suffisante pour calculer l'utilité d'une stratégie mixte.

Cette propriété peut amener à de grands gains en temps et/ou espace. Le calcul d'une probabilité jointe à partir des connaissances de marginales n'est pas un problème trivial (voir e.g. [Franc *et al.*, 2010]). De plus, la représentation des probabilités jointes, même via un réseau bayésien, peut demander un espace exponentiel.

Exemple 29. *L'exemple du jeu à 3 joueurs présenté précédemment peut facilement être représenté sous la forme d'un jeu bayésien polymatriciel $G = \langle S, \Omega, \Theta, E, P, u\rangle$ où :*

- *(S, E) est le graphe de voisinage de la figure 4.1*
- *Les fonctions d'utilité locales sont définies par (voir figure 4.6) :*
  - *$u_n^{n,\nu}(B.C, \theta_n.\theta_\nu) = u_n^{n,\nu}(C.B, \theta_n.\theta_\nu) = 0$,*
  - *$u_n^{n,\nu}(B.B, \theta_n.\theta_\nu) = \beta$ si $\theta_n = R$, sinon α ($\theta_n = \bar{R}$) ,*
  - *$u_n^{n,\nu}(C.C, \theta_n.\theta_\nu) = 1$.*
- *Les fonctions de probabilité locales sont définies par :*
  - *$P_{n,\nu}(R.R) = P_{n,\nu}(\bar{R}.\bar{R}) = 0.3 + \frac{0.4}{6} = \frac{11}{30}$*
  - *$P_{n,\nu}(R.\bar{R}) = P_{n,\nu}(\bar{R}.R) = 2.\frac{0.4}{6} = \frac{2}{15}$*

		R		$\bar{R}$	
		B	C	B	C
R	B	β, β	$0, 0$	β, α	$0, 0$
	C	$0, 0$	$1, 1$	$0, 0$	$1, 1$

		B	C	B	C
$\bar{R}$	B	α, β	$0, 0$	α, α	$0, 0$
	C	$0, 0$	$1, 1$	$0, 0$	$1, 1$

FIGURE 4.6 – Tables d'utilités par paires du jeu bayésien polymatriciel.

2. Ici, $P(\theta_{e-n}|\theta_n)$ est la probabilité conditionnelle d'avoir le type joint θ_{e-n} (type des joueurs de e autres que n), sachant le type θ_n du joueur n. Et $P(\theta_{-e}|\theta_e)$ est la probabilité conditionnelle du type joint θ_{-e}, type de tous les joueurs non inclus dans e, sachant le type joint local θ_e du jeu e .

En considérant la même stratégie pure σ mentionnée précédemment (le joueur joue B s'il reçoit une publicité sinon il joue C), pour le jeu à 3 joueurs avec le graphe complet, on a :

$$\begin{aligned}
EU_n^{n,\nu}(\sigma_{n,\nu}|R) &= P(R|R).u_n(B.B, R.R) + P(\bar{R}|R).u_n(B.C, R.\bar{R}) = \beta.P(R|R) = \beta.\frac{11}{15} \\
EU_n^{n,\nu}(\sigma_{n,\nu}|\bar{R}) &= P(R|\bar{R}).u_n(C.B, R.\bar{R}) + P(\bar{R}|\bar{R}).u_n(C.C, R.\bar{R}) = P(\bar{R}|\bar{R}) = \frac{11}{15} \\
EU_n(\sigma|R) &= \Sigma_{\nu \neq n} EU_n^{n,\nu}(\sigma_{n,\nu}|R) = \beta.\frac{22}{15}. \\
EU_n(\sigma|\bar{R}) &= \frac{22}{15}.
\end{aligned}$$

Il est possible de vérifier que cette stratégie est un équilibre de Nash puisque changer unilatéralement les stratégies des joueurs de n'importe quel type diminuera leurs utilités à $\frac{8}{15}$ (lorsque $\sigma_n(R) = C$) ou à $\alpha\frac{8}{15}$ (lorsque $\sigma_n(\bar{R}) = B$).

4.4 Complexité spatiale

Bien sûr, tous les jeux bayésiens n'impliquent pas exclusivement des interactions locales. Mais c'est le cas pour certains et une représentation polymatricielle ou hypergraphique du jeu, quand elle est possible, est moins coûteuse que sa représentation sous forme normale. La première raison est que, contrairement aux jeux sous forme normale, les (H, B)-games ne demandent pas une description globale de la table de probabilités jointes P. La seconde raison est que le coût de la représentation des fonctions d'utilité ne sera jamais supérieur et sera, en général, inférieur quand une représentation hypergraphique est utilisée. C'est un résultat connu [Papadimitriou et Roughgarden, 2008] pour les jeux à information complète et cela reste vrai pour les (H, B)-games. En notant ρ l'arité maximale d'une hyperarête d'un (H, B)-game, nous avons :

Proposition 11 (Taille de représentation)**.** *La complexité spatiale de la représentation d'un (H, B)-game d'arité maximale ρ est $O\left(|E|.\rho.t^\rho.d^\rho\right)$ [3]. Plus particulièrement, elle est de l'ordre de $O\left(|E|.t^2.d^2\right)$ pour les jeux bayésiens polymatriciels.*

Preuve (Proposition 11)**.** *Pour toute hyperarête $e \in E$, nous avons besoin d'une table d'utilité locale u_i^e pour tout $i \in e$.*

Ces tables ont $O\left(d^\rho t^\rho\right)$ éléments chacun, ce qui amène au résultat.

Le complexité spatiale d'un jeu bayésien polymatriciel en résulte lorsque $\rho = 2$.

Cette complexité spatiale doit être comparée à la taille du jeu sous forme normale bayésien équivalent au (H, B)-game. La forme normale implique une distribution de probabilité de taille $O\left(t^N\right)$ et N fonctions d'utilité de taille $O\left(d^N \cdot t^N\right)$. Donc, lorsque ρ est bornée par une constante (e.g. pour un jeu bayésien polymatriciel, $\rho = 2$) la représentation graphique est exponentiellement moins coûteuse que la représentation sous forme normale.

Maintenant, montrons que la complexité spatiale de la représentation dépend des fonctions d'utilité, et non pas de la distribution de probabilité. Plus précisément, elle dépend du degré de décomposabilité des fonctions d'utilité.

Définition 30. *Une fonction f est k-décomposable si et seulement s'il existe un ensemble F de fonctions d'arités inférieures ou égales à k telles que $f = \Sigma_{f' \in F} f'$.*

3. Où t et d sont le nombre de type et d'action par joueur.

Posons la définition suivante :

Définition 31 (Jeu bayésien k-décomposable)**.**
Un jeu bayésien $\langle S, \Omega, \Theta, P, u\rangle$ est k décomposable si et seulement si il existe un ensemble E de sous ensembles de $S = \{1, ..., N\}$ tel que :
— $|e| \leqslant k, \forall e \in E$
— $u_n(\omega, \theta) = \sum_{e \in E, n \in e} u_n^e(\omega_e, \theta_e)$ *où* $u_n^e : \Omega_e \times \Theta_e \mapsto \mathbb{R}, \forall e \in E, n \in e$

Proposition 12 (Représentation équivalente en (H, B)-game)**.**
Tout jeu bayésien sous forme normale k-décomposable admet une représentation équivalente à un (H, B)-game, dont les arêtes sont de taille bornée par k.

Preuve (Proposition 12)**.** *Si la définition 23 est appliquée à un jeu bayesien sous forme normale k-décomposable, nous obtenons pour tout (σ, n, θ_n) :*

$$\begin{aligned} EU_n(\sigma|\theta_n) &= \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega} u_n(\omega, \theta)\sigma(\omega|\theta). \\ &= \sum_{e \in E, n \in e} \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega \in \Omega} \sigma(\omega|\theta) u_n^e(\omega_e, \theta_e), \\ &= \sum_{e \in E, i \in e} \sum_{\theta_{e-n}} P(\theta_{e-n}|\theta_n) \sum_{\omega_e \in \Omega_e} \sigma(\omega_e|\theta_e) u_n^e(\omega_e, \theta_e), \\ &= \sum_{e \in E, n \in e} EU_n^e(\sigma_e|\theta_n), \end{aligned}$$

dans le (H, B)-game obtenu à partir du jeu bayesien k-décomposable en définissant $P(\theta_e) = \sum_{\theta_{-e}} P(\theta), \forall e \in E, \theta_e \in \Theta_e$.

A partir de la proposition 10, nous avons donc ce (H, B)-game qui est une représentation équivalente du jeu bayésien k-décomposable (définissant les mêmes utilités sur les stratégies mixtes). De plus, évidemment $|e| \leqslant k, \forall e \in E$.

Notons que le résultat ne nécessite pas que la distribution de probabilité jointe soit décomposable. Par exemple, la fonction d'utilité du jeu de coordination décrit dans l'exemple 27 est 2 décomposable, mais la distribution de probabilité jointe ne peut pas être récupérée à partir de l'ensemble des marginales, et sa représentation est exponentielle. Néanmoins, comme montré précédemment, le problème peut être représenté par un jeu polymatriciel bayésien, de taille exponentiellement inférieure à celle du jeu sous sa forme bayésienne originale.

4.4.1 Travaux liés

Peu de modèles ont été proposés pour les jeux bayésiens succincts. [Singh *et al.*, 2004] considèrent les problèmes pour lesquelles les fonctions d'utilité sont décomposables et où les "jeux locaux" forment un arbre (un graphe acyclique). Cependant, la distribution de probabilité n'est pas représentée de manière compacte. Les *Bayesian Action-Graph Games (BAGG)* [Jiang et Leyton-Brown, 2010] généralisent les jeux bayésiens, les jeux hypergraphiques et certains types de jeux anonymes (où seul le nombre de joueurs jouant une action a une influence sur l'utilité d'un joueur). Dans les BAGGs, les fonctions d'utilité sont présentées dans le modèle des AGG [Jiang *et al.*, 2011] qui offrent une représentation plus générale que les jeux hypergaphiques. Elles peuvent être exponentiellement plus compactes dans certains cas (e.g. jeux anonymes). D'un

autre côté, la distribution de probabilité jointe sur les types est modélisée sous la forme d'un réseau bayésien alors que dans un (H, B)-game, seules les marginales pertinentes sont représentées. La représentation d'un réseau bayésien peut être nettement plus coûteuse que via les marginales d'une distribution (dans le jeu de coordination, par exemple). De plus, le calcul de l'utilité espérée pour une stratégie demande d'effectuer une tâche d'inférence des probabilités marginales du réseau bayésien, ce qui prend, un temps exponentiel dans le cas général. Ce calcul est polynomial dans les (H, B)-games, quand ρ est borné. Dans un jeu bayésien polymatriciel, par exemple, le calcul de $EU^n(\sigma|\theta_n)$ implique au plus $N-1$ calculs de $EU_i(\sigma|\theta_i)$, qui ont une complexité de $O(d^2.t)$ chacun.

4.5 Équivalence entre les jeux à information complète et incomplète

Nous avons vu dans la section précédente qu'un jeu hypergraphique peut être exponentiellement plus succinct que son équivalent sous forme normale bayésienne. Nous montrons maintenant que le gain en concision obtenu n'amène aucune augmentation en complexité de calcul théorique. Pour faire ceci, nous montrons que tout (H, B)-game peut être transformé en son équivalent hypergraphique à information complète en temps polynomial. Cela demande d'étendre le théorème de Howson et Rosenthal, cette fois aux jeux bayésiens hypergraphiques.

Définissons la transformation.

Définition 32 (Transformation d'un (H, B)-game)**.** *Pour un jeu bayésien hypergraphique $G = \langle S, \Omega, \Theta, E, P, u\rangle$, $\tilde{G} = \langle \tilde{S}, \tilde{\Omega}, \tilde{E}, \tilde{u}\rangle$ est le jeu hypergraphique défini par :*

- $\tilde{S} = \{(n, \theta_n), n \in S, \theta_n \in \Theta_n\}$ *est l'ensemble des joueurs. Il y a un joueur (n, θ_n) pour chaque type θ_n du joueur n*
- $\tilde{\Omega} = \bigotimes_{(n,\theta_n)} \Omega_n$ *i.e.* $\Omega_{(n,\theta_n)} = \Omega_i$ *est l'ensemble des actions jointes.*
- $\tilde{E} = \{h(e, \theta_e), e \in E, \theta_e \in \Theta_e\}$ *où* $h(e, \theta_e) = \{(n, \theta_n), n \in e, \theta_n = (\theta_e)_n\}$ *est l'ensemble des jeux locaux. Pour chaque type joint θ_e d'un des jeux locaux du (H, B) d'origine, il existe un nouveau jeu local $\tilde{\theta}_e$ entre les joueurs (n, θ_n) pour lesquels $\theta_n \in \tilde{\theta}_e$*
- $\tilde{u} = \{\tilde{u}^{h(e,\theta_e)}_{(n,\theta_n)}, \tilde{\theta}_e \in \tilde{E}, (n, \theta_n) \in \tilde{\theta}_e\}$ *où* $\tilde{u}^{h(e,\theta_e)}_{(n,\theta_n)}(\omega_e) = p(\theta_{e-\{n\}}|\theta_n).u^e_n(\omega_e, \theta_e)$

En résumé, l'idée est de considérer que chaque type possible θ_n d'un joueur n devient un joueur (jouant $\sigma_n(.|\theta_n)$). θ_n joue dans les jeux locaux comprenant toutes les combinaisons possibles des types θ_ν où ν est un joueur autre que n qui appartient à un jeu local de n. Les stratégies de G et celles de $\tilde{G}$ sont en bijection et l'utilité de (n, θ_n) dans $\tilde{G}$ est égale à l'utilité espéré du joueur n de type θ_n dans G (la probabilité est intégrée dans la nouvelle utilité qui est en fait une utilité espérée).

Exemple 30. *Nous allons reprendre une fois de plus l'exemple de la sélection d'un fournisseur internet. Le jeux bayésien polymatriciel à 3 joueurs devient un jeu polymatriciel à information complète à 6 joueurs représenté dans la figure 4.7. Chaque arête $\big((n, \theta_n), (\nu, \theta_\nu)\big)$ correspond à un jeu bimatriciel. Les utilités correspondantes sont décrites dans les tables de la figure 4.8*

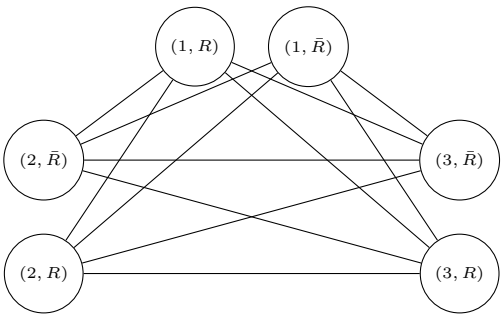


FIGURE 4.7 – Jeu polymatriciel à 6 joueurs issu de la transformation du jeu bayésien polymatriciel à 3 joueurs de la figure 4.1.

		R		$\bar{R}$	
		B	C	B	C
R	B	$\beta\frac{11}{15}, \beta\frac{11}{15}$	$0, 0$	$\beta\frac{4}{15}, \alpha\frac{4}{15}$	$0, 0$
	C	$0, 0$	$\frac{11}{15}, \frac{11}{15}$	$0, 0$	$\frac{4}{15}, \frac{4}{15}$

		B	C	B	C
$\bar{R}$	B	$\alpha\frac{4}{15}, \beta\frac{4}{15}$	$0, 0$	$\alpha\frac{11}{15}, \alpha\frac{11}{15}$	$0, 0$
	C	$0, 0$	$\frac{4}{15}, \frac{4}{15}$	$0, 0$	$\frac{11}{15}, \frac{11}{15}$

FIGURE 4.8 – Table d'utilité par paire du jeu bayésien polymatriciel.

La transformation d'une stratégie mixte du jeu bayésien hypergraphique G vers une stratégie mixte dans le jeu hypergraphique $\tilde{G}$ équivalent s'effectue comme dans la définition 28 (lorsque l'on a un jeu bayésien et son équivalent hypergraphique), c'est-à-dire $\tilde{\sigma}_{(n,\theta_n)}(\omega_n) = \sigma_n(\omega_n|\theta_n), \forall \omega_n \in \Omega_n$.

Rappelons nous que, comme montré dans la section 4.2, un jeu bayésien à N joueurs est transformable en un jeu hypergraphique à information complète. Un jeu bayésien hypergraphique peut être décrit comme un jeu hypergraphique où tous les jeux locaux sont bayésiens. Donc pour un jeu (H, B)-game donné, lorsque nous effectuons une transformation cela revient à appliquer une transformation sur tous les jeux locaux de ce (H, B)-game. Le jeu hypergraphique résultant maintient les utilités espérées des stratégies mixtes :

Proposition 13 ((H, B)-game équivalent)**.** *Si G est un (H, B)-game et $\tilde{G}$ est défini comme dans la définition 32, alors :*

$$EU_n(\sigma|\theta_n) = EU_{(n,\theta_n)}(\tilde{\sigma}), \forall n \in S, \theta_n \in \Theta_n, \forall \sigma$$

Preuve (Proposition 13 (D'un (H, B)-game vers un jeu hypergraphique))**.** *Prenons G un (H, B)-game où $n \in S, \theta_n \in \Theta_n$ sont fixés, et σ est une stratégie mixte arbitraire de G. $\tilde{G}$ est le jeu hypergraphique issue de la transformation, comme définie dans la définition 32 et $\tilde{\sigma}$ est la stratégie mixte de $\tilde{G}$, définie à partir de σ comme dans la définition 28.*

$\tilde{G}$, défini à partir de σ comme dans la définition 28.

Notez qu'à partir de la proposition 10, nous avons :

$$EU_n(\sigma|\theta_n) = \sum_{e \in E, n \in e} EU_n^e(\sigma_e|\theta_n),$$

où $EU_n^e(\sigma_e|\theta_n)$ est l'utilité espérée de la stratégie mixte σ restreinte aux joueurs dans $e \in E$ du jeu bayésien G^e.

Maintenant, je prends $\tilde{\sigma}_e$ une stratégie $\tilde{\sigma}$ restreinte à ses composants (ν, θ_ν) où $\nu \in e$. Donc, en appliquant la proposition 7 au jeu G^e, nous obtenons :

$$EU_n^e(\sigma_e|\theta_n) = EU^e_{(n,\theta_n)}(\tilde{\sigma}_e).$$

Mais, pour tout jeu hypergraphique G' (qui est un cas particulier du (H, B)-game), appliquer la proposition 10 implique que :

$$EU_n(\sigma) = \sum_{e \in E, n \in e} EU_n^e(\sigma_e).$$

Appliquant ceci à $\tilde{G}$, nous obtenons :

$$\begin{aligned} EU_{(n,\theta_n)}(\tilde{\sigma}) &= \sum_{e \in E, n \in e} EU^e_{(n,\theta_n)}(\tilde{\sigma}_e), \\ &= \sum_{e \in E, n \in e} EU_n^e(\sigma_e|\theta_n) = EU_n(\sigma|\theta_n). \end{aligned}$$

Par conséquent, les équilibres de $\tilde{G}$ sont en bijection avec ceux de G (comme ce qui a été démontré dans la section 4.2 lorsque l'on passe d'un jeu bayésien à un jeu hypergraphique) ; formellement :

Proposition 14 (Équivalence des équilibres de Nash)**.**
σ est un équilibre de Nash pur (resp. mixte) d'un (H, B)-game G si et seulement si $\tilde{\sigma}$ est un équilibre de Nash pur (resp. mixte) de $\tilde{G}$

Preuve (Proposition 14)**.** *C'est une conséquence directe de la proposition 13.*

Finalement, il est facile de vérifier que la transformation décrite ci dessus se fait en temps polynomial. Ceci a, bien évidemment, une influence sur la complexité de la recherche des équilibres.

Proposition 15 (Transformation en temps polynomial)**.** *Pour tout (H, B)-game G, $\tilde{G}$ peut être calculé en temps polynomial (en la taille du jeu).*

Preuve (Proposition 15)**.** *Notez que calculer $\tilde{u}$ demande de calculer les $|\Omega_e|$ éléments des tables d'utilité $u^{(e,\theta_e)}_{(n,\theta_n)}$, pour chaque paires (e, θ_e). Mais d'un autre côté, u demande de stocker les $u_n^e(\omega_e, \theta_e)$ pour tout e, θ_e, ω_e. Chaque représentation est équivalente en taille, ce qui prouve la proposition (les autres éléments de $\tilde{G}$ sont calculés en temps polynomial).*

4.6 Complexité de la recherche d'équilibre

En utilisant, les équivalences qui ont été démontrées dans la section précédente, il est possible de démontrer quelques résultats de complexité pour les problèmes de recherche d'équilibre mixte dans les jeux bayésiens hypergraphiques. En utilisant ces résultats, il est aussi possible de confirmer un résultat pour la complexité de recherche des équilibres dans les jeux bayésiens sous forme normale, qui n'ont pas été démontrés jusqu'à présent.

4.6.1 Équilibre de Nash mixte

Le problème de recherche d'un équilibre de Nash mixte dans un jeu polymatriciel ou ϵ-approché dans un jeu hypergraphique classique est PPAD-complet [Daskalakis *et al.*, 2006]. La recherche d'un équilibre de Nash mixte ou ϵ-approché dans un jeu bayésien polymatriciel/hypergraphique n'est pas plus facile car un jeu hypergraphique est (H, B)-game particulier.

Définition 33 (Équilibre de Nash ϵ-approché)**.** *Une stratégie mixte ξ^* est un équilibre de Nash ϵ-approché si et seulement si pour toutes les stratégies pures ω_n de tous les joueurs n nous avons :*

$$EU_n(\xi_n^*.\xi_{-n}^*) \geqslant EU_n(\omega_n.\xi_{-n}^*) - \epsilon, n \in S, \forall \omega_n \in \Omega_n$$

L'existence d'un équilibre de Nash mixte dans un (H, B)-game est garantie car tout (H, B)-game G est équivalent, sémantiquement, à un jeu bayésien sous forme normale $\langle S, \Omega, \Theta, P, \{u_n\}\rangle$ (où u_n est donné par la définition 29). Le problème n'est, en réalité, pas plus difficile ; c'est une conséquence directe de proposition 15 :

Proposition 16 (Calcul d'un équilibre de Nash mixte dans un (H, B)-game)**.** *Le problème de recherche d'une équilibre de Nash mixte ϵ-approché dans un jeu bayésien hypergraphique est* PPAD*-complet. La* PPAD*-complétude vaut également pour la recherche des équilibres de Nash mixtes dans les jeux bayésiens polymatriciels.*

Preuve (Proposition 16)**.** *Dans [Daskalakis* et al.*, 2006], il est établi que la recherche d'un équilibre de Nash mixte ϵ-approché dans un jeu hypergraphique (parmi d'autres types de jeux succincts) est* PPAD*-complet. La proposition 15 décrit une transformation en temps polynomial d'un (H, B)-game en un jeu hypergraphique equivalent. Par conséquent, on peut en déduire que le problème de recherche d'un équilibre de Nash mixte ϵ-approché pour un (H, B)-game appartient à* PPAD*. D'un autre côté, tout jeu hypergraphique peut être représenté par un (H, B)-game dégénéré (sans incertitude). Cela prouve la complétude de la résolution des (H, B)-games pour la classe PPAD.*

Notez que la taille de $\tilde{G}$ est comparable à celle de G. Représenter G demande de stocker $|E|.t^\rho.(\rho.d^\rho + 1)$ nombres. De l'autre côté, $\tilde{G}$ contient $|E|.t^\rho$ jeux sous forme normale avec $\rho.t$ tables de d^ρ nombres, i.e. $|E|.t^{\rho+1}.\rho.d^\rho$ nombres. La proposition 16 implique que résoudre un (H, B)-game n'est pas plus difficile que résoudre un jeu hypergraphique à information complète de taille similaire.

Revenons maintenant rapidement sur les jeux bayésiens en forme normale. Bien évidemment trouver un équilibre de Nash mixte (ϵ-approché) dans jeu bayésien sous forme normale est au moins aussi difficile que dans un jeu sous forme normale à information complète classique ; ce problème est donc PPAD-difficile. Mais notez que les jeux bayésiens sous forme normale sont des cas particuliers des (H, B)-games (où il y a une seule hyperarête). Puisque la transformation en temps polynomial s'applique aussi aux jeux bayésiens sous forme normale, on peut affirmer que :

Corollaire 2 (Calcul d'équilibre de Nash mixte dans un jeu bayésien SFN)**.** *Le problème de calcul d'un équilibre de Nash mixte ϵ-approché dans un jeu bayésien sous forme normale est* PPAD*-complet.*

Preuve (Corollaire 2)**.** *Bien sûr, il est connu que trouver un équilibre de Nash mixte ϵ-approché dans un jeu Bayésien est un problème* PPAD*-difficile, puisque les jeux*

bayésiens sont une extension des jeux sous forme normale. Mais, puisque les jeux bayésiens sont aussi une forme particulière des (H, B)-games avec un seul jeu local, il est possible d'utiliser la transformation en temps polynomial de la proposition 15 afin de les transformer en jeux hypergraphiques. Et puisque trouver un équilibre de Nash mixte ϵ-approché dans un jeu hypergraphique est un problème qui appartient à la classe PPAD*, alors c'est aussi le cas pour les jeux bayésiens.*

Pour les jeux bayésiens, seule la NP complétude du problème d'existence d'un équilibre de Nash *pur* est connue [Conitzer et Sandholm, 2008]. Le résultat démontré précédemment sur les équilibres de Nash mixtes permet donc de compléter le tableau.

4.6.2 Équilibre de Nash pur

En ce qui concerne les équilibres de Nash purs, une conséquence de la proposition 13 est que le problème de décision de l'existence d'un équilibre de Nash pur dans un (H, B)-game appartient à NP. D'un autre côté, il est connu que le problème n'est pas "facile" : il est montré dans [Conitzer et Sandholm, 2008] que décider de l'existence d'un équilibre de Nash pur dans un jeu bayésien à 2 joueurs est NP-complet. Donc :

Proposition 17 (Existence d'un équilibre de Nash pur)**.**
Déterminer s'il existe un équilibre de Nash pur dans un (H, B)-game est un problème NP-complet.

Preuve (Proposition 17)**.** *La NP complétude du problème d'existence d'un équilibre de Nash pur est connue pour les jeux graphiques à information complète [Gottlob* et al.*, 2005]. Il est donc aussi NP-difficile pour les jeux hypergraphiques, qui généralisent les jeux graphiques. Par conséquent, déterminer s'il existe un équilibre de Nash pur pour un (H, B)-game est un problème NP-difficile. D'un autre côté, calculer l'utilité d'une stratégie jointe pure pour un joueur demande un temps polynomial (en la taille du problème) dans les (H, B)-games. Le nombre de stratégies pures pour chaque joueur est borné par $max_{n \in S} |\Omega_n|^{|\Theta_n|}$, qui n'est pas plus élevé que l'espace nécessaire à la représentation (qui demande de stocker les tables $u_n^e(\omega_e, \theta_e)$). Par conséquent, vérifier si une stratégie jointe pure donnée pour un (H, B)-game est un équilibre de Nash, demande seulement un temps polynomial en la taille d'expression du jeu. Donc, l'appartenance à NP du problème d'existence d'un équilibre de Nash pur est assurée.*

En résumé, dans un jeu bayésien sous forme normale comme dans un jeu sous forme normale à information complète, ou dans un jeu bayésien hypergraphique, ou dans un jeu hypergraphique à information complète, nous confirmons que "the game world is flat" comme écrit par [Daskalakis *et al.*, 2006]. Le possible gain exponentiel en concision offerte par les jeux bayésiens hypergraphiques comparé à la forme normale des jeux bayésiens n'amène pas d'augmentation de la complexité théorique du calcul des équilibres.

4.7 Conclusion et Perspectives

Dans ce chapitre, un nouveau formalisme de représentation succincte des jeux à information incomplète, les jeux bayésiens hypergraphiques, a été proposé. Le théorème de Howson et Rosenthal, montrant qu'un jeu bayésien à 2 joueurs peut être transformé en un jeu polymatriciel à information complète équivalent, a été étendu aux jeux

bayésiens avec N joueurs ainsi qu'aux jeux bayésiens hypergraphiques. Ces derniers peuvent être transformés en jeux hypergraphiques à information complète. Il a aussi été démontré que ce nouveau formalisme, ainsi que les transformations, n'amène pas d'augmentation de la complexité de recherche d'un équilibre de Nash (mixte ou pur).

Il peut être intéressant d'étendre le cadre des jeux bayésien hypergraphique à d'autre type de jeu. Par exemple, sur des jeux séquentiels, plus particulièrement les "Partially Observed Stochastic Games (POSG)" [Hansen *et al.*, 2004], un cadre qui généralise les processus décisionnel de Markov partiellement observable [Kaelbling *et al.*, 1998] et le cadre des jeux (sous forme normale).

Dans le prochain chapitre, une description d'un nouvel algorithme de calcul d'équilibre de Nash mixte dans les jeux sous forme normale (à information complète) avec plus de 2 joueurs sera présenté.

Chapitre 5

Algorithme de Wilson

5.1 Introduction

La méthode de calcul d'un équilibre de Nash mixte dans un jeu à 2 joueurs, par une approche de parcours de chemin, définie par [Lemke et Howson, 1964] a été étendue par [Wilson, 1971]. Ce dernier a décrit théoriquement une nouvelle méthode de parcours de chemin "géométrique" permettant de calculer un équilibre de Nash mixte dans un jeu à N joueurs (qui comprend les jeux avec deux joueurs mais, à partir de maintenant, lorsque j'utiliserai ce terme, je sous-entendrai des jeux avec plus de deux joueurs). Cette méthode théorique définie par Wilson a été adaptée pour la création de l'algorithme de parcours de chemin pour les jeux polymatriciels, que j'ai décrit précédemment, l'algorithme de Howson [Howson, 1972]. Aucune implémentation de la méthode de Wilson n'a été proposée pour les jeux à N joueurs non polymatriciels. Les outils mathématiques et informatiques disponibles à l'époque ne permettaient pas de construire une méthode implémentable. Aussi, Wilson n'a proposé qu'une description mathématique (incomplète) de son approche.

Wilson a démontré qu'il est possible de représenter le problème de recherche d'un équilibre de Nash mixte comme un Problème de Complémentarité Polynomiale (PCP, Polynomial Complementary Problem en anglais) [1]. Je décrirai cette représentation dans la section 5.2. La méthode d'énumération de support [Porter *et al.*, 2008] présentée brièvement dans le chapitre 1 permet de calculer un équilibre de Nash mixte. Je montre dans la section 5.3 que cette méthode permet de résoudre un PCP en explorant de manière exhaustive tous les supports des stratégies possibles, en éliminant les stratégies dominées et en résolvant des systèmes d'équations polynomiales afin de calculer un équilibre de Nash via la solution d'un PCP.

Dans la section 5.4, je présente le coeur de ma contribution, une version "combinatoire" originale de l'algorithme de parcours de chemin de Wilson pour résoudre un PCP. Cet algorithme sera détaillé dans la section 5.5. Dans la section 5.6, je montre qu'il est possible d'étendre la méthode aux jeux dégénérés et qu'il est possible de tirer bénéfice de la représentation succincte d'un jeu pour construire son PCP et potentiellement résoudre ce PCP.

L'implémentation python des algorithmes présentés dans ce chapitre et les résultats expérimentaux seront décrits dans le chapitre 7.

1. Notez que ce terme n'est pas utilisé par Wilson et n'a été proposé que plus tard par [Gowda, 2017].

5.2 Calcul d'équilibre de Nash et Problème de Complementarité Polynomiale

[Wilson, 1971] a montré que pour tout jeu à N joueurs $\langle S = \{1, ..., N\}, \Omega, u\rangle$, le problème de recherche d'un équilibre de Nash mixte peut être ramené à un problème de complémentarité (polynomiale), que nous allons définir.

Tout d'abord, on définit un jeu équivalent $\Gamma^N = \langle S, \Omega, a\rangle$ du jeu $\langle S, \Omega, u\rangle$, dans lequel les joueurs n'ont pas de tables d'utilité mais des tables de "désutilité". La désutilité d'une action jointe ω pour le joueur n se définit de la manière suivante :

$$a_n(\omega) = 1 + max_{\omega' \in \Omega} u_n(\omega') - u_n(\omega), \forall \omega \in \Omega, \forall n \in S$$

Par conséquent, chaque joueur aura une désutilité minimale de 1. Dans un équilibre de Nash, un joueur cherche à minimiser sa désutilité.

Exemple 31 (Calcul des désutilités)**.** *Considérons un jeu à 3 joueurs avec la table d'utilités suivante :*

ω_1	ω_2	ω_3	u_1	u_2	u_3
0	0	0	6	0	4
0	0	1	7	3	7
0	1	0	0	7	4
0	1	1	5	4	0
1	0	0	1	6	3
1	0	1	4	5	2
1	1	0	2	1	6
1	1	1	3	2	1

L'utilité maximale de chacun des 3 joueurs est la même, 7*. Par conséquent, la désutilité de chaque joueur est définie par :* $a_n(\omega) = 8 - u_n(\omega)$*. Le résultat est :*

ω_1	ω_2	ω_3	a_1	a_2	a_3
0	0	0	2	8	4
0	0	1	1	5	1
0	1	0	8	1	4
0	1	1	3	4	8
1	0	0	7	2	5
1	0	1	4	3	6
1	1	0	6	7	2
1	1	1	5	6	7

Par définition, une stratégie mixte ξ est un équilibre de Nash mixte si et seulement si les équations et inéquations suivantes sont respectées :

$$\begin{cases} A^n(\xi) & \leqslant & A_i^n(\xi^{-n}), \forall n \in S, i \in \Omega_n, \\ A^n(\xi) & = & A_i^n(\xi^{-n}), \forall n \in S, i \in \Omega_n, \text{ tq } \xi_i^n > 0. \end{cases} \tag{5.1}$$

où $A^n(\xi)$ est la désutilité espérée du joueur n pour ξ et $A_i^n(\xi^{-n})$ est la désutilité

espérée du joueur n pour sa stratégie i lorsque les autres joueurs jouent selon ξ^{-n}.

$$A_i^n(\xi^{-n}) =_{def} \sum_{\substack{\omega\in\Omega \\ \omega_n=i}} a_n(\omega)\prod_{\nu\neq n}\xi^\nu_{\omega_\nu}, \forall n\in S, i\in\Omega_n \tag{5.2}$$

$$\text{et } A^n(\xi) =_{def} \sum_{i\in\Omega_n} A_i^n(\xi^{-n})\xi_i^n, \forall n\in S \tag{5.3}$$

Une stratégie mixte ξ est un équilibre de Nash mixte s'il n'est pas possible de changer unilatéralement la stratégie d'un joueur et de diminuer sa désutilité.

Wilson propose un changement de variables, plutôt que d'utiliser une distribution de probabilité ξ. Une distribution de probabilité non normalisée x sera utilisée (vérifiant simplement $x_i^n \geqslant 0, \forall n\in S, \forall i\in\Omega_n$). Les stratégies jointes mixtes peuvent être représentées indirectement par une liste de vecteurs $x=(x^n)_{n\in S}$ de coordonnées $(x_i^n)_{i\in\Omega_n}\in(\mathbb{R}^+)^D$ où $D=\sum_{n\in S}|\Omega_n|$. Une distribution de probabilité unique est obtenue en normalisant chaque x^n. Cette distribution correspondra, comme on le verra plus tard, à une stratégie mixte ξ^n : $\xi_i^n=\frac{x_i^n}{\sum_{j\in\Omega_n}x_j^n}$.

Utilisant un ensemble de variables (x_i^n) et la définition d'un équilibre de Nash mixte, il est possible de définir un problème de complémentarité polynomiale "équivalent" :

Définition 34 (Problème de complémentarité polynomiale)**.** *Le PCP correspondant à un jeu $\Gamma^N=(S,\Omega,a)$ est le système d'équations et d'inéquations sur les variables $(x_i^n)_{n\in S,i\in\Omega_n}\in(\mathbb{R}^+)^D$:*

$$\forall(n,i)\in I_N, \left\{\begin{array}{rcll} x_i^n & \geqslant & 0 & \\ A_i^n(x^{-n}) & \geqslant & 1 & (\mathcal{S}_N) \\ x_i^n\cdot(A_i^n(x^{-n})-1) & = & 0 & \end{array}\right.$$

où $I_N=_{def}\{(n,i), n\in S, i\in\Omega_n\}$.

En reprenant les équations 5.2 et 5.3, les "désutilités espérées" des joueurs peuvent être écrites sous la forme de polynômes multilinéaires :

$$A_i^n(x^{-n}) =_{def} \sum_{\substack{\omega\in\Omega \\ \omega_n=i}} a_n(\omega)\prod_{\nu\neq n}x^\nu_{\omega_\nu}, \forall n\in S, i\in\Omega_n \tag{5.4}$$

$$\text{et } A^n(x) =_{def} \sum_{i\in\Omega_n} A_i^n(x^{-n})x_i^n, \forall n\in S \tag{5.5}$$

On note $\mathcal{D}^N$ l'ensemble des points $x\in(\mathbb{R}^+)^D$ qui respectent l'ensemble des inéquations du PCP : $\mathcal{D}^N=\left\{x\in(\mathbb{R}^+)^D, x_i^n\geqslant 0,\ A_i^n(x^{-n})\geqslant 1, \forall(n,i)\in I_N\right\}$. Un point $x\in\mathcal{D}^N$ sera appelé un *point faisable*.

Un problème $\mathcal{S}_N$ est appelé un problème de complémentarité polynomiale puisque l'on cherche une solution x non négative pour laquelle pour toutes les paires $(n,i)\in I_N$, au moins l'une des deux conditions suivantes est satisfaite : soit $x_i^n=0$ soit $A_i^n(x^{-n})=1$. Un point des coordonnées x respectant ces conditions pour toutes les paires sera appelé *point complémentaire*. x^n représente la stratégie mixte non normalisée du joueur n (donc $x_i^n=0$ si et seulement si $\xi_i^n=0$), $A_i^n(x^{-n})$ est la désutilité (à un facteur multiplicatif près) du joueur n appliquant la stratégie i lorsque les autres joueurs applique ξ^{-n}. Les égalités dans $\mathcal{S}_N$ modélisent le concept de "meilleure réponse" dans un jeu : si i n'est pas une meilleure réponse pour le joueur n (i.e $A_i^n(x^{-n})>1$), sa probabilité doit être à 0 (donc x_i^n aussi). De manière similaire, si $A_i^n(x^{-n})=1$, la

stratégie i minimise la désutilité espérée du joueur n. Donc cette stratégie pure sera une meilleure réponse à la stratégie ξ^{-n}, $(x_i^n > 0)$, elle sera de probabilité non nulle.

Dans le cas où il n'y a que deux joueurs ($N = 2$), il est possible de vérifier que $\forall(n,i) \in I_N$, $A_i^n(x^{-n}) = 1$ est une équation linéaire. Dans ce cas, le problème $\mathcal{S}_2$ est un *Problème de Complémentarité Linéaire* [Lemke et Howson, 1964].

[Wilson, 1971] montre l'équivalence entre un équilibre de Nash d'un jeu et une solution correspondante du PCP de ce même jeu.

Proposition 18 (Équivalence NE/PCP [Wilson, 1971])**.** *Soit Γ^N un jeu à N joueurs et $\mathcal{S}_N$ sa transformation en PCP. Il y a une bijection entre les équilibres de Nash de Γ^N et les solutions de $\mathcal{S}_N$:*

1. *Si x est une solution de $\mathcal{S}_N$, alors ξ défini par $\xi_i^n = \frac{x_i^n}{\sum_{j\in\Omega_n} x_j^n}$, $\forall(n,i) \in I_N$ est un équilibre de Nash de Γ^N.*
2. *Si ξ est un équilibre de Nash mixte de Γ^N, x défini par*

$$x_i^n = \left(\frac{\prod_{\nu\neq n} A^\nu(\xi)}{A^n(\xi)^{N-2}}\right)^{\frac{-1}{N-1}} \xi_i^n, \forall(n,i) \in I_N \text{ est une solution de } \mathcal{S}_N.$$

Preuve (Proposition 18)**.** *Voir [Wilson, 1971]*

Exemple 32 (Création des polynomes)**.** *En reprenant l'exemple précédent :*

ω_1	ω_2	ω_3	u_1	u_2	u_3	a_1	a_2	a_3
0	0	0	6	0	4	2	8	4
0	0	1	7	3	7	1	5	1
0	1	0	0	7	4	8	1	4
0	1	1	5	4	0	3	4	8
1	0	0	1	6	3	7	2	5
1	0	1	4	5	2	4	3	6
1	1	0	2	1	6	6	7	2
1	1	1	3	2	1	5	6	7

(5.6)

Dans ce jeu, les 3 joueurs ont chacun 2 stratégies. Il y a donc un ensemble de 6 couples $I_N =_{def} \{(1,0),(1,1),(2,0),(2,1),(3,0),(3,1)\}$. Le PCP correspondant aura donc 6 variables $(x_0^1, x_1^1, x_0^2, x_1^2, x_0^3, x_1^3,)$ et 6 polynômes A_i^n, 2 par joueur.

Les polynômes sont construits selon les formules suivantes :

$$A_i^1\left(x^2,x^3\right) = \sum_{\omega_2,\omega_3} a_1(i,\omega_2,\omega_3) x_{\omega_2}^2 x_{\omega_3}^3 \tag{5.7}$$

$$A_i^2\left(x^1,x^3\right) = \sum_{\omega_1,\omega_3} a_2(\omega_1,i,\omega_3) x_{\omega_1}^1 x_{\omega_3}^3 \tag{5.8}$$

$$A_i^3\left(x^1,x^2\right) = \sum_{\omega_1,\omega_2} a_3(\omega_1,\omega_2,i) x_{\omega_1}^1 x_{\omega_2}^2 \tag{5.9}$$

En développant :

$$\begin{aligned}
A_0^1\left(x^2,x^3\right) &= 2x_0^2x_0^3 + x_0^2x_1^3 + 8x_1^2x_0^3 + 3x_1^2x_1^3\\
A_1^1\left(x^2,x^3\right) &= 7x_0^2x_0^3 + 4x_0^2x_1^3 + 6x_1^2x_0^3 + 5x_1^2x_1^3\\
A_0^2\left(x^1,x^3\right) &= 8x_0^1x_0^3 + 5x_0^1x_1^3 + 2x_1^1x_0^3 + 3x_1^1x_1^3\\
A_1^2\left(x^1,x^3\right) &= x_0^1x_0^3 + 4x_0^1x_1^3 + 7x_1^1x_0^3 + 6x_1^1x_1^3\\
A_0^3\left(x^1,x^2\right) &= 4x_0^1x_0^2 + 4x_0^1x_1^2 + 5x_1^1x_0^2 + 2x_1^1x_1^2\\
A_1^3\left(x^1,x^2\right) &= x_0^1x_0^2 + 8x_0^1x_1^2 + 6x_1^1x_0^2 + 7x_1^1x_1^2
\end{aligned}$$

Rappelons qu'une solution $x \in (\mathbb{R}^+)^D$ d'un PCP est appelée un *point complémentaire*. Un point $x \in \mathbb{R}^D$ est appelé un *point (n,i)-presque complémentaire* s'il satisfait toutes les contraintes de $\mathcal{S}_N$ avec la *possible exception* de l'équation $x_i^n(A_i^n(x^{-n})-1)=0$.

La recherche d'un équilibre de Nash mixte d'un jeu Γ^N revient à la recherche d'une solution d'un PCP $\mathcal{S}_N$, soit un point faisable et complémentaire.

Exemple 33 (Point faisable complémentaire)**.** *En reprenant l'exemple précédent, $I_N = \{(1,0),(1,1),(2,0),(2,1),(3,0),(3,1)\}$ et le PCP correpondant :*

$$\left\{\begin{array}{rcl} x_0^1, x_1^1, x_0^2, x_1^2, x_0^3, x_1^3 & \geqslant & 0 \\ A_0^1(x^2,x^3), A_1^1(x^2,x^3) & \geqslant & 1 \\ A_0^2(x^1,x^3), A_1^2(x^1,x^3) & \geqslant & 1 \\ A_0^3(x^1,x^2), A_1^3(x^1,x^2) & \geqslant & 1 \\ x_0^1\cdot(2x_0^2x_0^3+x_0^2x_1^3+8x_1^2x_0^3+3x_1^2x_1^3-1) & = & 0 \\ x_1^1\cdot(7x_0^2x_0^3+4x_0^2x_1^3+6x_1^2x_0^3+5x_1^2x_1^3-1) & = & 0 \\ x_0^2\cdot(8x_0^1x_0^3+5x_0^1x_1^3+2x_1^1x_0^3+3x_1^1x_1^3-1) & = & 0 \\ x_1^2\cdot(x_0^1x_0^3+4x_0^1x_1^3+7x_1^1x_0^3+6x_1^1x_1^3-1) & = & 0 \\ x_0^3\cdot(4x_0^1x_0^2+4x_0^1x_1^2+5x_1^1x_0^2+2x_1^1x_1^2-1) & = & 0 \\ x_1^3\cdot(x_0^1x_0^2+8x_0^1x_1^2+6x_1^1x_0^2+7x_1^1x_1^2-1) & = & 0 \end{array}\right. \quad (\mathcal{S}_N)$$

Considérons le point x :

— $x^1 = (x_0^1, x_1^1) = (\frac{1}{6}\sqrt{\frac{55}{21}}, \frac{1}{6}\sqrt{\frac{77}{15}})$

— $x^2 = (x_0^2, x_1^2) = (\frac{3}{2}\sqrt{\frac{3}{385}}, \frac{3}{4}\sqrt{\frac{15}{77}})$

— $x^3 = (x_0^3, x_1^3) = (\frac{1}{3}\sqrt{\frac{35}{33}}, 0)$

On vérifie que les valeurs des polynômes précédents pour ce point sont :

$$\begin{array}{rcl} A_0^1\left(x^2,x^3\right)-1 & = & 0 \\ A_1^1\left(x^2,x^3\right)-1 & = & 0 \\ A_0^2\left(x^1,x^3\right)-1 & = & 0 \\ A_1^2\left(x^1,x^3\right)-1 & = & 0 \\ A_0^3\left(x^1,x^2\right)-1 & = & 0 \\ A_1^3\left(x^1,x^2\right)-1 & = & 0.925 > 0 \end{array}$$

Par ailleurs, $\forall (n,i) \in I_N$ soit $x_i^n = 0$, soit $A_i^n(x^{-n}) = 1$ (ici, $x_1^3 = 0$, en particulier). Comme $x \in \mathcal{D}^N$, x est une solution du PCP. Si nous normalisons x, nous obtenons :

— $\xi^1 = (\frac{5}{12}, \frac{7}{12})$

— $\xi^2 = (\frac{2}{7}, \frac{5}{7})$

— $\xi^3 = (1,0)$

Qui correspond à un équilibre de Nash mixte du jeu initial.

5.3 Résolution d'un PCP par énumération de supports

La méthode d'énumération de supports décrite dans [Porter *et al.*, 2008] pour calculer un équilibre de Nash mixte semble, a priori, très éloignée de l'algorithme de Wilson que nous décrirons par la suite. Néanmoins, nous montrons ici qu'elle peut aussi être décrite simplement en termes de résolution d'un PCP.

Un support est, par définition, un sous ensemble $W \subseteq I_N$ tel que $\forall (n,i) \notin W$, $x_i^n = 0$ ($\xi_i^n = 0$), et $\forall (n,i) \in W, x_i^n > 0$ (dans le cas non dégénéré) donc $\xi_i^n > 0$. On retrouve la définition classique du support d'une stratégie mixte jointe : $W = W_1 \cup W_2 \cup ... \cup W_N$ où $W_n = \{(n,i) \in W\}_{i \in \Omega_n}$, W_n est le support de la stratégie ξ^n.

L'approche de [Porter *et al.*, 2008] consiste à énumérer tous les ensembles de supports joints $W \subseteq I_N$ tels que $W \cap \{(n,i)\}_{i \in \Omega_n} \neq \varnothing, \forall n \in S$ et à résoudre un système $\mathcal{S}^{\overline{W},W}$ (cette notation sera justifiée dans la section suivante) pour chaque ensemble de supports. Résoudre un PCP $\mathcal{S}_N$ est équivalent à trouver $W \subseteq I_N$ et $x \in \mathcal{D}^N$ tels que

$$\forall n \in S, \exists i \text{ tq } (n,i) \in W \text{ et } \left\{ \begin{array}{rl} x_i^n = 0, & \forall (n,i) \in \overline{W}, \\ A_i^n(x^{-n}) = 1, & \forall (n,i) \in W \end{array} \right. \quad (\mathcal{S}^{\overline{W},W})$$

Il résulte de la section précédente qu'une solution x d'un système $\mathcal{S}^{\overline{W},W}$, s'il en existe une, correspondra à un équilibre de Nash ξ.

Il y a exactement $\prod_{i=1}^{N} \left(2^{|\Omega_i|} - 1\right)$ supports possibles $W \subseteq I_N$ et tous ne correspondent pas à un système avec une solution. Essayer de résoudre tous les systèmes $\mathcal{S}^{\overline{W},W}$ pour tous les W n'est pas une bonne idée. Il est suggéré par [Porter *et al.*, 2008] d'énumérer les supports par ordre de taille de W croissante (ainsi si $W \subseteq W'$ alors W' sera exploré après W) et de tenter de résoudre successivement les systèmes $\mathcal{S}^{\overline{W},W}$ correspondants.

Exemple 34 (Ordre d'énumération)**.** *En prenant un jeu à trois joueurs où chaque joueur a deux actions, il existe 27 supports différents.*

Dans un premiers temps, les supports énumérés seront ceux où les joueurs ont chacun un support de taille 1 (taille totale du support : 3), cela revient à énumérer les supports des 8 stratégies jointes pures.

Ensuite, la taille du support est augmentée (elle passe à 4). Ici, ce seront les supports joints où 1 joueur a une stratégie de support de taille 2 *et les 2 autres de taille* 1 *qui seront énumérés, ce qui fait un total de 12 supports différents.*

Une fois de plus la taille du support est augmentée (elle passe à 5), il y a deux joueurs avec des stratégies support de taille 2 *et un de taille* 1*, ce qui fait un total de 6 supports joints à énumérer.*

Puis finalement, on passe au support de taille 6. Les deux premiers et deux derniers supports joints W énumérés peuvent être les suivants (l'ordre sur les supports n'est pas unique) :

- $W_0 = \{(1,0),(2,0),(3,0)\}$
- $W_1 = \{(1,0),(2,0),(3,1)\}$
- $W_{25} = \{((1,0),(2,0),(2,1),(3,0),(3,1)\}$
- $W_{26} = \{(1,0),(1,1),(2,0),(2,1),(3,0),(3,1)\}$

Quand un système $\mathcal{S}^{\overline{W},W}$ avec une solution est trouvé, l'algorithme s'arrête et retourne la stratégie mixte correspondant à la solution.

Afin de limiter le nombre de systèmes à résoudre, avant d'essayer de résoudre un système $\mathcal{S}^{\overline{W},W}$, nous regardons s'il y a une stratégie dans W, dominée dans le sous jeu défini en ne conservant que les stratégies de W. S'il existe au moins une stratégie dominée alors cela signifie que le support W peut être réduit en un support W'' tel que $W'' \subseteq W$ (un équilibre de Nash ne comprend pas de stratégies dominées dans son support). Compte tenu de l'ordre dans lequel les supports sont explorés, le support W'' a déjà été exploré. Par conséquent, si l'algorithme n'a pas trouvé de solution pour $\mathcal{S}^{\overline{W''},W''}$ alors il n'en trouvera pas pour $\mathcal{S}^{\overline{W},W}$. Donc nous pouvons passer ce système sans le résoudre.

Cette méthode garantit de trouver un équilibre de Nash (ou bien un continuum d'équilibres si le jeu est "dégénéré").

En général, l'énumération de support est efficace, surtout lorsqu'il existe des équilibres avec des supports de petite taille. Cependant pour les jeux où il n'existe pas d'équilibre avec un support de petite taille, il faut potentiellement explorer un grand nombre de supports et résoudre un grand nombre de système d'équations.

Exemple 35 (Jeu à trois joueurs)**.** *Reprenons le jeu présenté précédemment :*

ω_1	ω_2	ω_3	u_1	u_2	u_3	a_1	a_2	a_3
0	0	0	6	0	4	2	8	4
0	0	1	7	3	7	1	5	1
0	1	0	0	7	4	8	1	3
0	1	1	5	4	0	3	4	8
1	0	0	1	6	3	7	2	5
1	0	1	4	5	2	4	3	6
1	1	0	2	1	6	6	7	2
1	1	1	3	2	1	5	6	7

(5.10)

Ce jeu n'admet pas d'équilibre pur. Donc, lors de l'énumération, les systèmes basés sur des supports de taille 3 n'admettent pas de solution. On passe ensuite à l'énumération des supports de taille 4 (un total de 12 supports, 4 par "distribution" possible), en commençant par $W = \{(1,0),(1,1),(2,0),(3,0)\}$*. Pour ce premier support on obtient le sous-jeu suivant :*

ω_1	ω_2	ω_3	u_1	u_2	u_3
0	0	0	6	0	4
1	0	0	1	6	3

(5.11)

Il est facile de voir que la stratégie 1 *du joueur* 1 *est dominée par la stratégie* 0*. Lorsque l'on cherche un équilibre de Nash elle peut être supprimée mais cela signifie que l'on cherche à résoudre* $\mathcal{S}^{\overline{W'},W'}$ *avec* $W' = \{(1,0),(2,0),(3,0)\}$ *que l'on a déjà rencontré, puisque l'on résout les systèmes* $\mathcal{S}^{\overline{W},W}$ *dans un ordre compatible avec l'inclusion des* W*. Pas besoin de chercher à résoudre* $\mathcal{S}^{\overline{W'},W'}$ *et nous pouvons passer au support suivant* $W = \{(1,0),(1,1),(2,0),(3,1)\}$*.*

ω_1	ω_2	ω_3	u_1	u_2	u_3
0	0	1	7	3	7
1	0	1	4	5	2

(5.12)

La stratégie 1 *du joueur* 1 *est dominée donc on poursuit l'énumération. On observe que pour tous les supports de taille 4 il existe une stratégie dominée. Donc nous passons aux supports de taille 5. Le premier sous jeu correspondant au support* $\{(1,0),(1,1),(2,0),(2,1),(3,0)\}$ *est :*

ω_1	ω_2	ω_3	u_1	u_2	u_3
0	0	0	6	0	4
0	1	0	0	7	4
1	0	0	1	6	3
1	1	0	2	1	6

(5.13)

Aucune stratégie des joueurs 1 *et* 2 *n'est dominée. Par conséquent, il nous faut résoudre le système* $\mathcal{S}^{\overline{W},W}$ *avec* $W = \{(1,0),(1,1),(2,0),(2,1),(3,0)\}$ *et* $\overline{W} = \{(3,1)\}$ *:*

$$\left\{\begin{array}{rcl} x_0^1, x_1^1, x_0^2, x_1^2, x_0^3 & \geqslant & 0 \\ x_1^3 & = & 0 \\ A_0^1\left(x^2, x^3\right), A_1^1\left(x^2, x^3\right) & = & 1 \\ A_0^2\left(x^1, x^3\right), A_1^2\left(x^1, x^3\right) & = & 1 \\ A_0^3\left(x^1, x^2\right) & = & 1 \\ A_1^3\left(x^1, x^2\right) & \geqslant & 1 \\ x_0^1 \cdot (2x_0^2x_0^3 + x_0^2x_1^3 + 8x_1^2x_0^3 + 3x_1^2x_1^3 - 1) & = & 0 \\ x_1^1 \cdot (7x_0^2x_0^3 + 4x_0^2x_1^3 + 6x_1^2x_0^3 + 5x_1^2x_1^3 - 1) & = & 0 \\ x_0^2 \cdot (8x_0^1x_0^3 + 5x_0^1x_1^3 + 2x_1^1x_0^3 + 3x_1^1x_1^3 - 1) & = & 0 \\ x_1^2 \cdot (x_0^1x_0^3 + 4x_0^1x_1^3 + 7x_1^1x_0^3 + 6x_1^1x_1^3 - 1) & = & 0 \\ x_0^3 \cdot (4x_0^1x_0^2 + 4x_0^1x_1^2 + 5x_1^1x_0^2 + 2x_1^1x_1^2 - 1) & = & 0 \\ x_1^3 \cdot (x_0^1x_0^2 + 8x_0^1x_1^2 + 6x_1^1x_0^2 + 7x_1^1x_1^2 - 1) & = & 0 \end{array}\right. \quad \left(\mathcal{S}^{\overline{W},W}\right)$$

Il existe une solution à ce système, qui correspond à la stratégie mixte calculée dans l'exemple 33, $\xi = \{(\frac{5}{12}, \frac{7}{12}), (\frac{2}{7}, \frac{5}{7}), (1, 0)\}$

L'algorithme de Wilson [Wilson, 1971] peut être vu de manière similaire, comme la résolution de systèmes successifs $\mathcal{S}^{Z,W}$ avec $Z \neq \overline{W}$ et un ordre particulier d'exploration des paires (Z, W).

5.4 Algorithme de Wilson

[Wilson, 1971], reprenant l'approche de [Lemke et Howson, 1964] utilisée pour résoudre les problèmes de complémentarité linaire, a proposé une approche mathématique de parcours de chemin pour résoudre des PCP dits "non dégénérés". La description de Wilson laisse certaines étapes de l'algorithme indéfinies et ne gère pas les jeux dégénérés. Dans cette section, je vais proposer une réécriture originale et fonctionnelle de l'approche de Wilson.

L'approche proposée est basée sur les définitions de noeuds, d'arc et de chemins (presque complémentaires), en tant qu'ensembles d'équations multilinéaires. La sous section 5.4.1 introduit la notion de PCP , la sous section 5.4.2 décrit les noeuds et la sous section 5.4.3 décrit les arcs et le chemin fini permettant d'obtenir une solution. La description de cette section est "conceptuelle". Dans la section 5.5, nous proposons une description plus algorithmique.

5.4.1 PCP non dégénéré

Soit un PCP $\mathcal{S}_N$. Pour tout $x \in \mathcal{D}^N$ (donc tous x *faisable*), on écrit $Z(x) = \{(n, i) \in I_N, x_i^n = 0\}$ et $W(x) = \{(n, i) \in I_N, A_i^n(x^{-n}) = 1\}$. Donc, par définition, $x \in \mathcal{D}^N$ est une solution de $\mathcal{S}_N$ (un point faisable et complémentaire) si et seulement si $Z(x) \cup W(x) = I_N$ et $Z(x) \cap W(x) = \varnothing$.

Nous cherchons à trouver $x \in \mathcal{D}^N$, solution de $\mathcal{S}_N$. Avec l'énumération de supports, nous passions par une résolution de systèmes $\mathcal{S}^{\overline{W},W}$, ici je vais utiliser des systèmes $\mathcal{S}^{Z,W}$ qui sont définis par :

$$\left\{\begin{array}{rl} & x \in \mathcal{D}^N, \\ x_i^n = 0, & \forall (n, i) \in Z, \\ A_i^n(x^{-n}) = 1, & \forall (n, i) \in W \end{array}\right. \quad (\mathcal{S}^{Z,W})$$

Dans le reste de cette section, nous partirons du principe que nous avons un PCP $\mathcal{S}_N$, non dégénéré. Un *PCP non dégénéré* est défini par :

Définition 35 (PCP non dégénéré)**.** *Le PCP $\mathcal{S}_N$ est non dégénéré si et seulement si les conditions suivantes sont respectées :*

1. *Aucun point de $\mathcal{D}^N$ ne satisfait plus de $|I_N|$ équations.*
2. *Il n'existe pas deux points distincts satisfaisant le même ensemble de $|I_N|$ équations.*

En décrivant ceci mathématiquement, la condition 1 est équivalente à

$$(1) : \forall x \in \mathcal{D}^N, |Z(x)| + |W(x)| \leqslant D = |I_N|$$

et la condition 2 est équivalente à :,

$$(2) : \left.\begin{array}{rcl} Z(x) = Z(y) & = & Z \\ W(x) = W(y) & = & W \\ |Z| + |W| & = & |I_N| \end{array}\right\} \Rightarrow x = y, \forall x, y \in \mathcal{D}^N \tag{5.14}$$

Intuitivement, un PCP est non dégénéré si aucune des contraintes polynomiales n'est redondante (il y a $|I_N|$ équations indépendantes dans un espace de dimension $|I_N|$). Pour l'instant, nous supposons la non dégénérescence des PCP considérés.

Nous nommons l'ensemble des solutions x d'un système $\mathcal{S}^{Z,W}$ comme un noeud $\rho(Z, W)$. Notez que dans le cas non dégénéré l'ensemble $\rho(Z, W)$ est un singleton ou est vide. Lorsqu'il est dégénéré, l'ensemble $\rho(Z, W)$ peut comprendre une solution correspondant à plusieurs noeud différents ou un continuum de solutions.

5.4.2 Noeuds complémentaires, presque complémentaires et initiaux

Soit un PCP $\mathcal{S}_N$ et I_N, l'ensemble des paires (n, i) associé. On définit, $\forall k \in \{1, ..., N\}$, $I_k = \{(n, i), n \leqslant k, i \in \Omega_n\}$.

Pour un PCP $\mathcal{S}_N$, supposé non dégénéré, et une stratégie jointe arbitraire $\omega^0 \in \Omega$, une solution correspondra à un noeud $\rho(Z, W)$ complémentaire non vide.

Définition 36 (Noeud complémentaire)**.** *Un noeud $\rho(Z, W)$ est un noeud ω^0-complémentaire au niveau $k \in \{1, ..., N\}$ si et seulement si*

1. *$Z \cap W \cap I_k = \varnothing$ et $I_k \subseteq (Z \cup W)$*
2. *$|W \cap (I_{k'} \backslash I_{k'-1})| = 1, \forall k' \in \{k + 1, ..., N\}$*
3. *$(I_{k'} \backslash I_{k'-1}) \backslash Z = \{(k', \omega^0_{k'})\}, \forall k' \in \{k + 1, ..., N\}$*
4. *$Z \cap (I_k \backslash I_{k-1}) \neq \varnothing, \forall k \in \{1, ..., N\}$*

En d'autre terme pour un noeud $\rho(Z, W)$ ω^0-complémentaire au niveau k :

— Condition (1) : Chaque couple (n, i) de I_k est présent soit dans Z, soit dans W (mais pas dans les deux).

— Condition (2) : Il y a exactement un couple $(k', i) \in (I_{k'} \backslash I_{k'-1})$ présent dans W, pour tous les niveaux $k' > k$.

— Condition (3) : Z contient tous les couples (k', i) pour tous les niveaux $k' > k$, à l'exception des couples $(k', \omega^0_{k'})$.

— Condition (4) : Z ne contient pas tous les couples (n, i) des joueurs $n \leqslant k$ (pas de stratégie de support vide).

Un noeud ω^0-complémentaire au niveau k définit un équilibre de Nash dans le sous jeu de Γ^N où les stratégies des joueurs d'indices $k' > k$ sont fixées à $\omega^0_{k'}$.

Il est également possible de définir un noeud ω^0-presque complémentaire au niveau k :

Définition 37 (Noeud presque-complémentaire)**.** *Un noeud $\rho(Z,W)$ est un noeud ω^0-presque complémentaire au niveau $k \in \{1,...,N\}$ si et seulement si*

1. $|Z \cap W \cap I_k| \leqslant 1$ *et* $|Z \cap I_k| + |W \cap I_k| = |I_k|$
2. $|W \cap (I_{k'} \backslash I_{k'-1})| = 1, \forall k' \in \{k+1,...,N\}$
3. $(I_{k'} \backslash I_{k'-1}) \backslash Z = \{(k', \omega^0_{k'})\}, \forall k' \in \{k+1,...,N\}$
4. $Z \cap (I_k \backslash I_{k-1}) \neq \varnothing, \forall k \in \{1,...,N\}$

La seule différence entre un noeud ω^0-presque complémentaire au niveau k et un noeud complémentaire est qu'il peut exister un (au plus) couple $(n,i) \in Z \cap W \cap I_k$. Dans ce cas, un autre couple (n',i') de I_k est absent à la fois de Z et W. Notez que pour un noeud ω^0-presque complémentaire au niveau k, le couple absent à la fois de Z et W sera le couple (k, ω^0_k). Les conditions (2), (3) et (4) sont inchangées.

Pour faire le lien avec les systèmes $\mathcal{S}^{Z,W}$, il est possible de décrire la chose de la manière suivante : Pour un jeu Γ^N et son PCP $\mathcal{S}_N$, si un noeud est ω^0-presque complémentaire au niveau k cela signifie que la solution est complémentaire pour tous les couples (n,i) où $n \in \{1,...,k\}$ à l'exception du couple (k, ω^0_k). Supposons que nous ayons un noeud $\rho(Z,W)$ ω^0-presque complémentaire au niveau k pour lequel, parmi tous les couples $(k,j) \in Z$ avec $j \in \Omega_k$, seul $(k, \omega^0_k) \notin Z$, alors cela signifie que $x^k_{\omega^0_k} > 0$ et $x^k_j = 0, j \neq \omega^0_k, j \in \Omega_k$. En termes de distribution de probabilité dans le jeu associé au PCP, cela revient à dire que le joueur k joue la stratégie pure ω^0_k (pour une question de simplicité considérons que $x^k_{\omega^0_k} = 1$). Donc pour ce noeud $\rho(Z,W)$ ω^0-presque complémentaire au niveau k, cela signifie qu'à l'exception du joueur k les conditions de complémentarité $x^n_j(A^n_j(x^{-n}) - 1) = 0$ sont satisfaites pour tout $(n,j) \in I_{k-1}$.

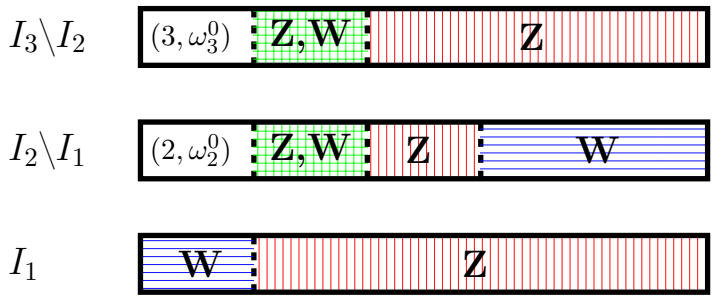


FIGURE 5.1 – Noeud $\rho(Z,W)$ ω^0-presque complémentaire au niveau 2

La figure 5.1 représente, pour un jeu à 3 joueurs, un noeud $\rho(Z,W)$ ω^0-presque complémentaire au niveau 2. Si l'on observe les couples du joueurs 2 (couples de $I_2 \backslash I_1$), nous pouvons voir qu'ils appartiennent soit à l'ensemble Z, soit à W, à l'exception du couple $(2, \omega^0_2)$ qui est absent de Z et de W. De plus, nous avons un couple qui est présent à la fois dans Z et dans W

Nous allons définir un troisième type de noeuds, les noeud initiaux. Ce sont des noeuds à la fois ω^0-presque complémentaires au niveau k et ω^0-complémentaires au niveau $k-1$.

Définition 38 (Noeud initial)**.** *Un noeud $\rho(Z,W)$ ω^0-presque complémentaire au niveau k est un noeud initial au niveau $k \in \{2,...,N\}$ si et seulement si, il satisfait la définition 37, ainsi que :*

1. $(I_k \backslash I_{k-1}) \backslash Z = \{(k, \omega^0_k)\}$
2. $|W \cap (I_k \backslash I_{k-1})| = 1$

Notez que si le couple de niveau k présent dans W est (k, ω_k^0) alors le noeud est initial et également complémentaire au niveau k.

Nous allons montrer qu'un équilibre de Nash peut être trouvé en résolvant une séquence déterministe de systèmes $\mathcal{S}^{Z,W}$ dont les solutions sont des noeuds $\rho(Z,W)$ ω^0-presque complémentaires (à différents niveaux), pour une stratégie jointe ω^0 arbitraire fixée.

Afin de décrire cette méthode, il est plus simple de décrire la séquence en partant de la fin, c'est à dire d'une solution correspondant à un noeud complémentaire au niveau N, jusqu'à atteindre un noeud complémentaire au niveau 1. Lors de l'application de l'algorithme, les noeuds presque complémentaires seront parcourus dans l'autre sens, c'est à dire en partant d'un noeud complémentaire au niveau 1 jusqu'à atteindre un noeud complémentaire au niveau N. Un noeud complémentaire au niveau 1 est facile à calculer, c'est pourquoi il est utilisé comme point de départ de l'algorithme.

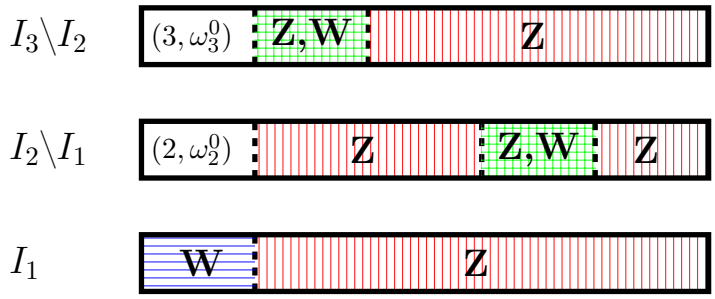


FIGURE 5.2 – Noeud $\rho(Z,W)$ ω^0-presque complémentaire initial au niveau 2 et complémentaire au niveau 1

La figure 5.2 représente pour un jeu à 3 joueurs, un exemple de noeud $\rho(Z,W)$ ω^0-presque complémentaire initial au niveau 2 et complémentaire au niveau 1. On peut voir que tous les couples du niveau 1 (couples de I_1), se trouvent soit dans Z soit dans W, mais pas dans les deux à la fois. Lorsque nous sommes au niveau 2, les couples de $I_2 \backslash I_1$ se trouvent tous dans Z à l'exception du couple $(2, \omega_2^0)$ qui est absent de Z et W. L'un des couples du joueur 2 est présent à la fois dans Z et W.

5.4.3 Arcs et chemins presque complémentaires

Comme mentionné précédemment, l'idée de l'algorithme est de parcourir une séquence de noeuds $\rho(Z,W)$ ω^0-presque complémentaires.

Deux noeuds ω^0-presque complémentaires successifs seront liés par ce que nous appellerons un arc ω^0-presque complémentaire. Supposant qu'un noeud $\rho(Z,W)$ ω^0-presque complémentaire ne soit pas vide, ce noeud sera un singleton, solution du système $\mathcal{S}^{Z,W}$. De la même manière, un arc $\gamma(Z,W)$ est un ensemble de solutions d'un système $\mathcal{S}^{Z,W}$ défini de la manière suivante :

Définition 39 (Arc presque-complémentaire)**.** *$\gamma(Z,W)$, ensemble de solutions d'un système $\mathcal{S}^{Z,W}$, est un arc ω^0-presque complémentaire au niveau $k \in \{1,...,N\}$ si et seulement si :*

1. $Z \cap W \cap I_k = \varnothing$
2. $|W \cap (I_{k'} \backslash I_{k'-1})| = 1, \forall k' \in \{k+1,...,N\}$
3. $I_k \backslash (Z \cup W) = \{(k, \omega_k^0)\}$

4. $Z \cap (I_k \backslash I_{k-1}) \neq \varnothing, \forall k \in \{1, ..., N\}$

Intuitivement, en comparant avec la définition 37 (noeud presque complémentaire), on voit que l'on peut passer d'un noeud $\rho(Z, W)$ ω^0-presque complémentaire au niveau k à un arc $\gamma(Z', W')$ ω^0-presque complémentaire au niveau k en "retirant" un couple (n, i) de Z ou de W. Nous formaliserons cette intuition plus loin.

Remarquons que si un arc $\gamma(Z, W)$ ω^0-presque complémentaire au niveau k est non vide et que le jeu est non dégénéré, $\gamma(Z, W)$ est inclus dans l'ensemble des solutions d'un système à $D-1$ équations sur D variables. La non dégénérescence implique que l'ensemble est de dimension 1 et peut être paramétré par un seul paramètre à valeur réelle. D'où l'appellation "arc".

Les points aux extrémités de $\gamma(Z, W)$, appartenant aux limites du domaine $\mathcal{D}^N$, sont des noeuds presque complémentaires. Un arc aura normalement deux points (noeuds) à ses extrémités (il est borné) ou bien un seul (il est non borné dans ce cas).

L'approche de Wilson consiste à suivre un chemin unidimensionnel dans $\mathcal{D}^N$, en traversant des arcs et des noeuds, jusqu'à éventuellement arriver à un noeud complémentaire. Cette approche se base sur la proposition suivante :

Proposition 19 (Arcs au voisinage d'un noeud)**.** *Soient $\mathcal{S}_N$, un PCP non-dégénéré, $\omega^0 \in \Omega$ et $k \in \{1, ..., N\}$: Tout noeud ω^0-presque complémentaire au niveau k a deux arcs ω^0-presque complémentaires bornés pour voisins. Si le noeud est complémentaire, il a un seul arc borné pour voisin.*

Preuve (Proposition 19)**.** *Pour toute paire $Z, W \subseteq I_N$ telle que $(\mathcal{S}^{Z,W})$ définit un noeud ω^0-presque complémentaire au niveau k, $\rho(Z, W)$, qui n'est ni complémentaire ni vide, les deux arcs ω^0-presque complémentaire qui "partent" de ce noeud sont $\gamma(Z \backslash W, W)$ et $\gamma(Z, W \backslash Z)$, ensembles de points de $\mathcal{D}^N$ satisfaisant respectivement les systèmes d'équations $\left(\mathcal{S}^{Z\backslash W,W}\right)$ et $\left(\mathcal{S}^{Z,W\backslash Z}\right)$. En effet, toute solution de $(\mathcal{S}^{Z,W})$ (un noeud) est aussi une solution de $\left(\mathcal{S}^{Z\backslash W,W}\right)$ et $\left(\mathcal{S}^{Z,W\backslash Z}\right)$. La non dégénérescence fait que cette solution n'appartient à aucun autre arc ω^0-presque complémentaire au niveau k. Par ailleurs, on vérifie que $\gamma(Z \backslash W, W)$ et $\gamma(Z, W \backslash Z)$ satisfont la définition 39.*

Si $(\mathcal{S}^{Z,W})$ désigne un noeud ω^0-complémentaire au niveau k, $\rho(Z, W)$, $Z \cap W \cap I_k = \varnothing$ et $I_k \subseteq (Z \cup W)$. Le système $(\mathcal{S}^{Z,W})$ admet une solution, alors, il y a un seul arc ω^0-presque complémentaire voisin de $\rho(Z, W)$: si $(k, \omega^0_k) \in Z$, c'est $\gamma(Z \backslash \{(k, \omega^0_k)\}, W)$ et si $(k, \omega^0_k) \in W$ c'est $\gamma(Z, W \backslash \{(k, \omega^0_k)\})$.

La proposition 19 implique la proposition suivante :

Proposition 20 (Chemin fini)**.** *Pour $\mathcal{S}_N$ un PCP non-dégénéré et $\omega^0 \in \Omega$, il existe un unique chemin fait d'une séquence de noeuds et d'arcs ω^0-presque complémentaires de différents niveaux qui, en partant au niveau N d'un noeud complémentaire, se termine en un autre noeud complémentaire au niveau N ou bien un noeud complémentaire au niveau* 1*.*

Preuve de la proposition 20 : Prenons $\mathcal{S}_N$, un PCP non-dégénéré et $\omega^0 \in \Omega$. Supposons que nous avons un noeud complémentaire non initial au niveau N.

Comme mentionné dans la proposition 19, un noeud complémentaire a un seul arc ω^0-presque complémentaire borné au niveau N pour voisin. Cet arc borné a, à son (autre) extrémité, un noeud presque complémentaire qui peut éventuellement être complémentaire ou initial.

Supposons que le noeud atteint n'est ni initial ni complémentaire. Il a lui aussi deux arcs ω^0-presque complémentaires voisins, l'un étant celui que nous venons de parcourir. Nous pouvons donc effectuer un nouveau parcours d'arc vers un autre noeud ω^0-presque complémentaire. Supposons maintenant que le noeud atteint soit complémentaire. Il y a un seul arc ω^0-presque complémentaire attaché a ce noeud. Nous avons donc terminé le parcours de chemin en atteignant un autre noeud complémentaire au niveau N.

Nous pouvons autrement atteindre un noeud initial. Lorsque nous avons un noeud initial (ou complémentaire initial) au niveau N, d'après la définition 38, il lui correspond un noeud complémentaire au niveau $N-1$. Ce qui veut dire que nous allons descendre de niveau. Compte tenu que nous avons un chemin acyclique, nous rencontrerons forcement un noeud complémentaire ou initial à l'extrémité d'un chemin pour un niveau donné.

Au niveau $N-1$, comme au niveau N, en partant d'un noeud complémentaire, nous avons un chemin qui se terminera donc en un noeud complémentaire ou initial. S'il est initial, nous descendons du niveau $N-2$ (puis au niveau $N-3$ si un noeud initial est rencontré, et ainsi de suite). S'il est complémentaire, nous allons effectuer un passage du niveau $N-1$ au niveau N car un noeud complémentaire à un niveau k est associé à un noeud initial au niveau supérieur $k+1$. Nous atteignons donc un noeud initial au niveau N, qui est l'extrémité d'un chemin dont l'autre extrémité est un noeud initial ou un nouveau noeud complémentaire.

Les changements de niveau entre un niveau k et un niveau $k-1$ ou $k+1$ s'effectuent tous de la même manière (que nous décrirons plus loin).

Lorsqu'un noeud initial au niveau 2 est atteint, il a un noeud complémentaire au niveau 1 associé, noeud qui correspond à la fin du chemin à parcourir.

Le chemin que nous venons de décrire peut être parcouru dans le sens inverse, ce qui nous donne :

Corollaire 3 (Chemin fini inverse)**.** *Pour un PCP $\mathcal{S}_N$ non-dégénéré et $\omega^0 \in \Omega$, il est possible de parcourir un chemin unique partant d'un noeud complémentaire au niveau 1 se terminant en un noeud complémentaire au niveau N*

Preuve (Corollaire 3)**.** *Conséquence de la proposition 20.*

Remarquez que si le PCP est non-dégénéré, il existe un unique noeud ω^0-complémentaire au niveau 1 (correspondant à une stratégie jointe pure $(\omega_2^0, ..., \omega_N^0)$ des autres joueurs).

Pour un niveau k, une représentation des différents "types" de chemins possibles est donnée dans la figure 5.3.

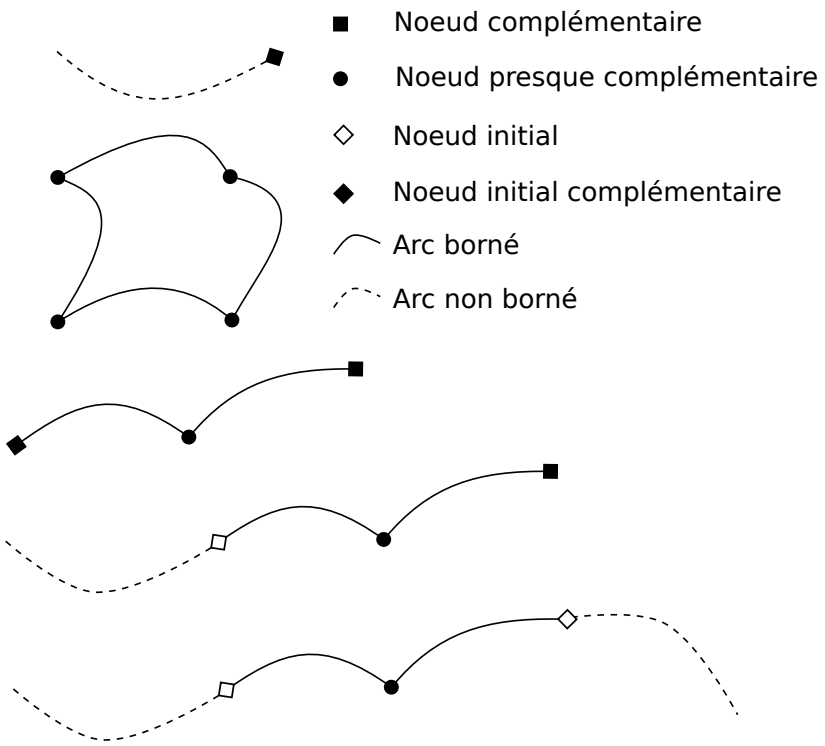


FIGURE 5.3 – Exemples de chemins presque-complémentaires à un niveau k.

Avec cette description combinatoire (elle se base sur des paires d'ensembles Z et W correspondant à des points presque complémentaires) en tête, nous pouvons décrire une version algébrique de l'algorithme de Wilson en utilisant les systèmes d'équations à résoudre successivement.

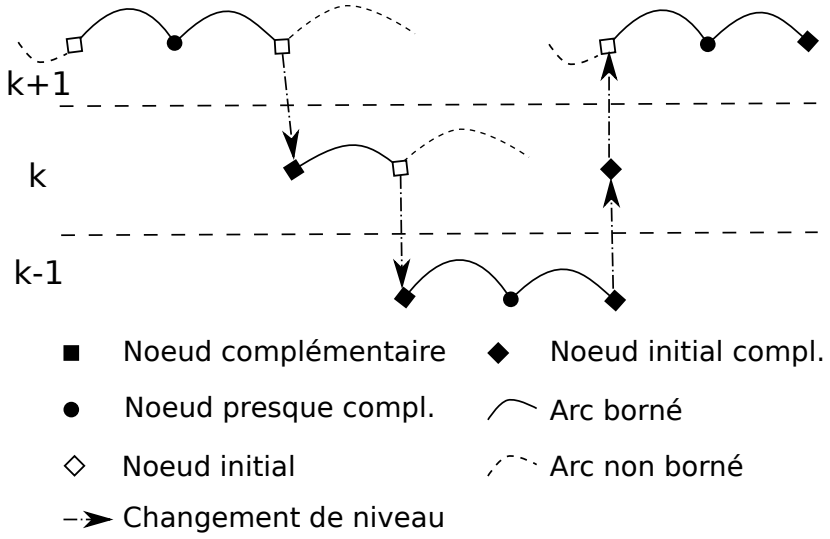


FIGURE 5.4 – Chemin avec changement de niveau.

5.5 Version "implémentable" de l'algorithme de Wilson

Nous avons décrit dans la section précédente, l'algorithme de [Wilson, 1971]. Cet algorithme consiste à parcourir un chemin constitué de noeuds ($\rho(Z, W)$) et d'arcs

$(\gamma(Z,W))$ ω^0-presque complémentaires et complémentaires à différents niveaux, en partant d'un noeud complémentaire au niveau 1, et jusqu'à un noeud complémentaire au niveau N, correspondant à un équilibre de Nash.

Dans cette section, nous allons décrire plus finement les différents éléments de l'algorithme de manière à le rendre implémentable. Plus précisément :

- Nous allons montrer qu'un noeud $\rho(Z,W)$ ω^0-complémentaire (resp : ω^0-presque complémentaire ou ω^0 initial) au niveau k d'un PCP $\mathcal{S}_N$ correspond à un noeud ω^0-presque complémentaire $\rho^k(Z,W)$ (resp : ω^0-presque complémentaire ou ω^0 initial) d'un sous PCP $\mathcal{S}_k$, comportant moins de variables et contraintes que $\mathcal{S}_N$
- Nous décrirons précisément la procédure de "traversée" d'un arc, qui permet de passer d'un noeud au suivant. Nous montrerons qu'elle nécessite un calcul de "variété algébrique", pouvant être mené exactement, en utilisant un solveur dédié.
- Nous montrerons que la procédure de montée, passage d'un noeud complémentaire au niveau k à un noeud initial au niveau $k+1$, s'exprime également comme une traversée d'arc (non borné).
- Enfin, nous montrerons que la descente de niveau ne nécessite pas de calcul particulier.

Pour l'implémentation plusieurs choix s'offrent à nous, il est possible de définir un PCP $\mathcal{S}_k$ pour chaque *sous jeu* possible obtenu en fixant les stratégies des joueurs $k' > k$ à $\omega^0_{k'}$, ou bien de définir des *"sous-PCP"* $\mathcal{S}_k$ à partir d'un PCP initial $\mathcal{S}_N$ en fixant les valeurs $x_i^{k'} > 0$. Ces deux approches sont parfaitement équivalentes, et nous allons présenter une version utilisant la seconde méthode. Utiliser l'une méthode ou l'autre revient à ajouter ou soustraire un entier (ou rationnel si les utilités du jeu sont des rationnels) aux utilités de chaque joueur, ce qui ne change pas les équilibres (à cause de la non normalisation des x^n).

5.5.1 Sous-PCP à un niveau k

Pour tout $1 \leqslant k \leqslant N$, nous notons $\mathbb{R}^{D_k}$, l'espace euclidien basé sur les variables x_i^n avec $(n,i) \in I_k$. La dimension de $(\mathbb{R}^+)^{D_k}$ est $D_k = |I_k|$ (et $D = D_N$).

$A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right)$ est le polynôme multivarié obtenu à partir de $A_i^n(x^{-n})$ en "fixant" toutes les valeurs de x_j^ν à 0 lorsque $\nu > k$ et $j \neq \omega^0_\nu$. Lorsque $\nu > k$ et $j = \omega^0_\nu$, nous "fixons" les x_j^ν à 1 (le choix de cette valeur est arbitraire mais elle doit être supérieure à 0). Nous avons donc :

$$A_i^{n,k}\left(x^{\{1,..,k\}\setminus\{n\}}\right) = \sum_{\substack{\omega\in\Omega,\omega_n=i\\ \omega_m=\omega^0_m,\forall m>k}} a^n_\omega \prod_{\substack{\nu\leqslant k,\\ \nu\neq n}} x^\nu_{\omega_\nu}. \tag{5.15}$$

$A_i^{n,k}$ est un polynôme multilinéaire de degré $k-1$. Remarquez que si ξ est une stratégie jointe mixte, $A_i^{n,k}\left(\xi^{\{1,..,k\}\setminus\{n\}}\right)$ est la désutilité espérée du joueur n jouant la stratégie $i \in \Omega_n$ lorsque les joueurs $1,...,k$ à l'exception de n jouent leur stratégie mixte dans ξ, tandis que les joueurs $k+1,...N$ jouent leur stratégie pure dans ω^0 et le joueur n joue la stratégie pure i.

A partir d'un PCP $\mathcal{S}_N$, on définit la séquence de sous-PCP $\mathcal{S}_k$, pour $k = 1,...,N-1$:

Définition 40 (Sous-PCP)**.** *Soit $\mathcal{S}_N$ un PCP et ω^0 une stratégie jointe pure fixée. Pour $2 \leqslant k < N$, le sous-PCP $\mathcal{S}_k$ est défini comme le système d'équations et inéquations*

polynomiales suivant, sur $\mathbb{R}^{D_k}$ *:*

$$\forall (n,i) \in I_k, \left\{ \begin{array}{rcl} x_i^n & \geqslant & 0 \\ A_i^{n,k}\left(x^{\{1,..,k\}\backslash\{n\}}\right) & \geqslant & 1 \\ x_i^n \cdot \left(A_i^{n,k}\left(x^{\{1,..,k\}\backslash\{n\}}\right) - 1\right) & = & 0 \end{array} \right. (\mathcal{S}_k)$$

Pour $n = 1$*, le PCP* $\mathcal{S}_1$ *est légèrement diffèrent. Cette construction différente est nécessaire si l'on veut maintenir la formulation du problème sous la forme d'un PCP, étant donné que les polynômes sont maintenant des constantes :*

$$\left\{ \begin{array}{rcl} x_i^1 & \geqslant & 0, \forall i \in S_1 \\ x_i^1 \cdot \left(\frac{a^1_{(i,\omega^0_{-1})}}{\min_{j\in\Omega_1} a^1_{(j,\omega^0_{-1})}} - 1\right) & = & 0, \forall i \in S_1 \\ \sum_{i\in\Omega_1} x_i^1 & = & 1 \end{array} \right. (\mathcal{S}_1)$$

Supposer que $\mathcal{S}_1$ est non dégénéré sous entend que le minimum $\min_{j\in\Omega_1} a^1_{(j,\omega^0_{-1})}$ est atteint par un seul indice j^*. Ainsi, il est possible de facilement calculer le point complémentaire au niveau 1, qui est caractérisé par $Z^1 = \Omega_1\backslash\{j^*\}$ et $W^1 = \{j^*\}$.

Comme le PCP initial, le sous-PCP $\mathcal{S}_k$ est supposé non dégénéré. Maintenant, il est possible de définir le système d'équations polynomiales $\mathcal{S}_k^{Z,W}$ pour $Z, W \subseteq I_k$:

$$\left\{ \begin{array}{rl} & x \in \mathcal{D}^k, \\ x_i^n = 0, & \forall (n,i) \in Z, \\ A_i^{n,k}(x^{\{1..k\}\backslash\{n\}}) = 1, & \forall (n,i) \in W, \end{array} \right. \qquad \left(\mathcal{S}_k^{Z,W}\right)$$

où $\mathcal{D}^k$ est défini par les inéquations de la définition 40.

Nous allons noter $\rho^k(Z,W)$ et $\gamma^k(Z,W)$ les noeuds et arcs ω^0-presque complémentaires au niveau k du sous PCP $\mathcal{S}_k$. En utilisant cette nouvelle notation, on peut redéfinir de la manière suivante les différentes classes de noeuds et arcs :

Définition 41 (Noeud complémentaire d'un sous PCP)**.** *Un noeud* $\rho^k(Z,W)$ *avec* $Z, W \subseteq I_k$ *est complémentaire pour le sous PCP* $\mathcal{S}_k$ *si et seulement si :*

$$Z \cap W = \varnothing \text{ et } Z \cup W = I_k.$$

Définition 42 (Noeud presque complémentaire d'un sous PCP)**.** *Un noeud* $\rho^k(Z,W)$ *est presque complémentaire pour le sous PCP* $\mathcal{S}_k$ *si et seulement si :*

$$|Z| + |W| = |I_k| \text{ et } |Z \cap W| \leqslant 1,$$

et si $|Z \cap W| = 1$ *alors* $(k, \omega_k^0) \notin (Z \cup W)$*.*

Définition 43 (Noeud initial d'un sous PCP)**.** *Un noeud* $\rho^k(Z,W)$ *est initial pour le sous PCP* $\mathcal{S}_k$ *si et seulement si il est presque complémentaire et satisfait :*

$$(I_k\backslash I_{k-1})\backslash Z = (k, \omega_k^0) \text{ et } |(I_k\backslash I_{k-1}) \cap W| = 1.$$

Un noeud initial peut aussi être un noeud complémentaire si $(I_k\backslash I_{k-1}) \cap W = \{(k, \omega_k^0)\}$

Définition 44 (Arc presque complémentaire d'un sous PCP)**.** *Un arc* $\gamma^k(Z,W)$ *est presque complémentaire pour* $\mathcal{S}_k$ *si et seulement si :*

$$Z \cap W = \varnothing \text{ et } Z \cup W = I_k\backslash\{(k, \omega_k^0)\}.$$

5.5.2 Traversée d'arc, montée et descente pour un système $\mathcal{S}_k$

Comme mentionné dans la section précédente, le parcours de chemin à un niveau k va se faire en traversant des arcs, jusqu'à atteindre un noeud complémentaire ou initial à partir duquel nous effectuerons une montée ou une descente.

Les arcs $\gamma^k(Z', W')$ et $\gamma^k(Z'', W'')$ voisins d'un noeud $\rho^k(Z, W)$ ω^0-presque complémentaire pour $\mathcal{S}_k$ sont "récupérés" de la même manière que dans l'explication de la section précédente : Un noeud $\rho^k(Z, W)$ a pour arcs voisins $\gamma^k(Z\backslash W, W)$ et $\gamma^k(Z, W\backslash Z)$.

Pour un noeud $\rho^k(Z, W)$ complémentaire, son unique arc ω^0-presque complémentaire voisin est soit l'arc $\gamma^k(Z\backslash\{(k, \omega_k^0)\}, W)$, soit $\gamma^k(Z, W\backslash\{(k, \omega_k^0)\})$.

Pour un noeud initial $\rho^k(Z, W)$, l'arc borné voisin sera $\gamma^k(Z\backslash W, W)$ et l'arc non borné voisin sera $\gamma^k(Z, W\backslash Z)$. Lorsque nous avons un noeud complémentaire, le noeud initial correspondant est calculé en ajoutant tous les couples du joueur k dans Z sauf (k, ω_k^0) et en calculant quel couple (de k) ajouté dans W permet d'avoir un noeud presque complémentaire faisable.

Lorsque l'on passe d'un noeud presque complémentaire à un arc presque complémentaire voisin, en fonction du sens emprunté, il va s'agir de retirer un couple $(n, i) \in I_k$ de Z ou de W pour ajouter un nouveau couple $(\nu, j) \in I_k$ dans l'ensemble Z ou W, pour atteindre un nouveau noeud presque complémentaire. Par exemple, si nous venons d'atteindre le noeud $\rho^k(Z, W)$ en venant de l'arc $\gamma^k(Z\backslash W, W)$ cela signifie que nous venons "d'ajouter" un couple (n, i) dans Z, couple qui est à la fois dans Z et W pour le noeud $\rho^k(Z, W)$. Donc le prochain arc que nous devons parcourir est $\gamma^k(Z, W\backslash Z)$ (nous retirons (n, i) de W). Le parcours dans le sens inverse est aussi possible (nous venons d'ajouter un couple dans W et retire ce même couple de Z).

Notre algorithme de parcours de chemin est très dépendant de la procédure de *traversée* d'un arc $\gamma^k(Z, W)$ presque complémentaire, pour $\mathcal{S}_k$ Cette procédure retourne un noeud presque complémentaire $\rho^k(Z', W')$ avec soit (i) $Z = Z'\backslash W'$ et $W = W'$ ou bien (ii) $Z = Z'$ et $W = W'\backslash Z'$. Ce problème de traversée d'arc peut être exprimé de manière algébrique. Premièrement, notez que par définition, $\gamma^k(Z, W) = \mathcal{V}\left(\mathcal{S}_k^{Z,W}\right) \cap \mathcal{D}^k$ et $\rho^k(Z', W') = \mathcal{V}\left(\mathcal{S}_k^{Z',W'}\right) \cap \mathcal{D}^k$, où $\mathcal{V}(\mathcal{S})$ est l'ensemble des solutions de $(\mathcal{S})$, ignorant la contrainte de domaine (c'est à dire les conditions de faisabilité). En termes algébriques, $\mathcal{V}(\mathcal{S})$ est appelée une variété algébrique affine [Cox *et al.*, 2015]. Quand un système $(\mathcal{S})$ est non dégénéré, la variété contient un seul point pour un noeud et sa dimension est 1 pour un arc. Les systèmes peuvent être résolus numériquement de manière approximative en utilisant des solveurs standards. Cependant au lieu de faire ceci, nous allons utiliser un solveur algébrique exact, basé sur le calcul de bases de Gröbner [Datta, 2010]. Ce type de solveur peut être plus lent que les solveurs numériques et limite la taille des jeux que nous pouvons résoudre. Cependant, il garantit des représentations finies et "exactes" des coordonnées des noeuds, qui seront nécessaires à la validation du bon fonctionnement de l'algorithme. Les coordonnées d'un point sont représentées par les solutions des polynômes univariés à coefficients rationnels. Ainsi en "stockant" les coefficients rationnels (ce qui nécessite un espace fini), on peut recalculer les solutions non rationnelles, à la volée, avec une précision arbitraire. Cette représentation exacte est particulièrement utile dans le cas des jeux dégénérés (à leur "détection" notamment). Dans notre implémentation, nous utilisons les fonctions implémentées dans la boîte à outils `Singular`, à laquelle nous accédons via l'environnement `Sagemath`, pour calculer les base de Gröbner et les variétés (voir `https://www.sagemath.org/index.html` pour plus d'information).

Le problème de traversée d'un arc pour $\mathcal{S}_k$ consiste, pour les deux paires données

(Z', W') et (Z, W), à calculer $(Z'', W'') \in I_k$ correspondant au noeud ω^0-presque complémentaire suivant, $\rho^k(Z'', W'')$. L'algorithme 1 calcule ce noeud presque complémentaire en essayant d'ajouter tous les couples $(\nu, j) \in I_k$, soit à Z' soit à W'.

Algorithm 1: TraverseArc$((Z, W), (Z', W'), I_k, \mathcal{S}_k)$.

```
   /* Calcule le noeud à la fin de l'arc γ^k(Z,W), pour un noeud de
      départ presque complémentaire ρ^k(Z',W') donné.              */
   /* Initialisation                                               */
 1 Sol ← ∅;
 2 for (ν, j) ∈ I_k do
 3 |  if (ν, j) ∈ I_k\Z' then
 4 |  |  Z_loc ← Z ∪ {(ν, j)}, W_loc ← W;
 5 |  |  if dim(V(S_k^{Z_loc,W_loc})) = 0 then
 6 |  |  |  ρ_loc ← V(S_k^{Z_loc,W_loc}) ∩ D^k;
 7 |  |  |  if ρ_loc ≠ ∅ then Sol ← Sol ∪ {(Z_loc, W_loc, ρ_loc)};
 8 |  if (ν, j) ∈ I_k\W' then
 9 |  |  Z_loc ← Z, W_loc ← W ∪ {(ν, j)};
10 |  |  if dim(V(S_k^{Z_loc,W_loc})) = 0 then
11 |  |  |  ρ_loc ← V(S_k^{Z_loc,W_loc}) ∩ D^k;
12 |  |  |  if ρ_loc ≠ ∅ then Sol ← Sol ∪ {(Z_loc, W_loc, ρ_loc)};
13 return Sol
```

La solution *Sol* retournée par l'algorithme 1 est un singleton quand $\mathcal{S}_k$ est non dégénéré :

Proposition 21 (Traversée d'un arc)**.** *Quand un PCP est non dégénéré, l'algorithme 1 retourne un seul triplet* $\left(Z'', W'', \mathcal{V}\left(\mathcal{S}_k^{Z'',W''}\right)\right)$*, et* $\mathcal{V}\left(\mathcal{S}_k^{Z'',W''}\right)$ *contient un seul point.*

Preuve (Proposition 21)**.** *En effet, pour un PCP non dégénéré, puisque* $\left(\mathcal{S}_k^{Z'',W''}\right)$ *contient* $|I_k|$ *équations, il n'y a pas deux points dans* $\mathcal{D}^k$ *qui peuvent les satisfaire et* ρ_{loc}*, calculé à la ligne 6 est soit vide soit un singleton.*

Supposons qu'il existe 2 solutions distinctes dans Sol (calculé à la ligne 7) associées à deux paires distinctes (Z'', W'') *et* (Z''', W''')*.* $\rho^k(Z'', W'')$ *et* $\rho^k(Z''', W''')$ *appartiennent à l'arc* $\gamma^k(Z, W)$*. Si nous considérons une paramétrisation de l'arc* $\gamma^k(Z, W)$*par un paramètre valant* 0 *pour le noeud presque complémentaire précédant*[2]*, alors le point* $\rho^k(Z'', W'')$ *ou* $\rho^k(Z''', W''')$ *qui aura la plus petite valeur de paramètre sera inclus dans Sol.*

5.5.3 Montée depuis un noeud complémentaire au niveau k

Nous allons décrire la procédure de changement de niveau (passage entre un noeud complémentaire au niveau k et un noeud initial au niveau $k+1$) avec l'utilisation des sous PCP.

Proposition 22 (Montée depuis un noeud complémentaire)**.** *Prenons* $\mathcal{S}_k$ *un sous-PCP de* $\mathcal{S}_N$ *au niveau* $1 \leqslant k < N$ *et la stratégie jointe pure arbitraire* ω^0*. Supposons*

2. C'est-à-dire qu'un arc $\gamma^k(Z, W)$ est paramétré avec x_j^ν si (ν, j) est sortie de Z ou avec $y_j^\nu = A_j^{\nu,k}\left(x^{\{1,..,k\}\setminus\nu}\right) - 1$ si (ν, j) est sortie de W

que $Z, W \subseteq I_k$ et que $\mathcal{S}_k^{Z,W}$ définit un noeud complémentaire de $\mathcal{S}_k$ dans $\mathbb{R}^{D_k}$. Alors $\mathcal{S}_{k+1}^{Z',W}$ définit un arc ω^0-presque complémentaire du sous-PCP au niveau $k+1$, $\mathcal{S}_{k+1}$, où $Z' = Z \cup \left\{(k+1,i), i \in \Omega_{k+1} \backslash \{\omega_{k+1}^0\}\right\}$. De plus, cet arc est voisin d'un seul noeud (initial) presque complémentaire au niveau $k+1$ (c'est un arc non borné)

Preuve (Proposition 22). *Prenons le PCP $(\mathcal{S}_{k+1})$ au niveau $k+1$. Si nous avons*

$$Z' = Z \cup \left\{(k+1,j), j \in \Omega_{k+1} \backslash \{\omega_{k+1}^0\}\right\},$$

alors $\left(\mathcal{S}^{Z',W}\right)$ définit un arc ω^0-presque complémentaire dans $\mathbb{R}^{D_{k+1}}$,$\gamma^{k+1}(Z',W)$. La presque complémentarité est vérifiée car

$$Z' \cap W = \varnothing \text{ et } Z' \cup W = I_{k+1} \backslash \left\{(k+1,\omega_{k+1}^0)\right\}.$$

Le fait que cet arc soit voisin d'un seul noeud presque complémentaire au niveau $k+1$ est une conséquence directe du lemme 2 de [Wilson, 1971], où les coordonnées des noeuds presque complémentaires sont exprimées en fonction des coordonnées du noeud complémentaire au niveau k.

Une fois cet arc trouvé, on cherche un couple $(k+1,i) \in I_{k+1} \backslash I_k$ à ajouter dans l'ensemble W afin d'obtenir un ensemble W' définissant un noeud $\rho^{k+1}(Z',W')$ presque complémentaire (ou complémentaire) de $\mathcal{S}_k$. Un système $\mathcal{S}^{Z',W'}$ avec une solution existe (c'est une conséquence du lemme 2 de [Wilson, 1971]). De plus, il est unique si le PCP est non dégénéré. Le calcul va s'effectuer de manière similaire à une traversée d'arc, la particularité étant que l'on considère seulement l'ajout de couples dans W.

Cette procédure s'effectue de cette manière étant donné que les conditions de complémentarité sont satisfaites pour tous les joueurs $n < k$, l'ajout d'un de leur couple n'est pas nécessaire et ne permettrait d'atteindre un noeud presque complémentaire au niveau k. Compte tenu que nous "défixons" le joueur k, cela revient à dire que pour le noeud initial correspondant au noeud complémentaire au niveau $k-1$, le joueur k joue seulement une action, ω_k^0, donc tous ses couples sont dans Z sauf celui correspondant à cette action. Si nous ajoutons un couple de k (le couple (k,ω_k^0)) dans Z alors cela correspondrait à ne jouer aucune action (de plus, nous aurions $A_i^n(x) = 0$ pour tous les joueurs et toutes leurs actions), donc seul l'ajout d'un couple de k dans W est envisageable.

Notez que si le noeud initial atteint $\rho^k(Z,W)$ n'est pas complémentaire, le prochain arc parcouru sera toujours l'arc $\gamma^k(Z \backslash W, W)$ (car nous venons d'ajouter un couple dans W).

5.5.4 Descente à partir d'un noeud initial au niveau k

Contrairement à la montée, la descente, peut se faire sans avoir à effectuer de calculs. Il suffit, pour un noeud initial $\rho^k(Z,W)$ de retirer les couples du niveau k de Z et W (tous les couples $(k,i) \in I_k \backslash I_{k-1}$).

Proposition 23 (Descente d'un noeud initial). *Prenons $\mathcal{S}_k^{Z,W}$ définissant un noeud initial au niveau $k > 1$. Prenons $Z' = Z \cap I_{k-1}$ et $W' = W \cap I_{k-1}$. Alors, $Z \cap W = \varnothing$ et $\mathcal{S}_{k-1}^{Z',W'}$ définit le noeud complémentaire.*

Preuve (Proposition 23). *Un noeud initial $\rho^k(Z',W')$ peut être obtenu à partir d'un noeud complémentaire au niveau $k-1$ en effectuant une montée suivant la procédure de la section précédente. Cette procédure ne modifie pas les couples de I_{k-1}. Ainsi, l'opération réciproque de descente consiste à garder dans Z' et W' les couples de Z et W appartenant à I_{k-1}.*

5.5.5 Algorithme complet

En appliquant le corollaire 3, nous pouvons proposer un algorithme complet implémentable permettant de calculer la solution d'un PCP $\mathcal{S}_N$.

Algorithm 2: WilsonProcedure($\mathcal{S}_N, I_N, \omega^0$).

```
   /* Calcule le noeud a la fin de l'arc γ^k(Z,W), pour un noeud de
      départ presque complémentaire ρ^k(Z',W') donné.              */
   /* Initialisation                                               */
 1 k ← 1;
 2 S ← getAllSk(S_N, ω^0);
 3 node ← firstlevelnode(S_1, I_1);
 4 prevnode ← None;
 5 while k ≠ N or iscomplementary(node)=False do
 6 |   tmpnode ← None;
 7 |   if k < N and iscomplementary(node) then
 8 |   |   if level(prevnode) ⩽ k then
 9 |   |   |   tmpnode ← lift(node, S_{k+1}, I_{k+1});
10 |   |   |   k ← k + 1 ;
11 |   |   else
12 |   |   |   if isinitial(node) then
13 |   |   |   |   tmpnode ← descend(node, S_{k-1}, I_{k-1});
14 |   |   |   |   k ← k − 1 ;
15 |   |   |   else
16 |   |   |   |   nextarc ← getnextarc(node, prevnode, I_k);
17 |   |   |   |   tmpnode ← traversearc(nextarc, node, I_k, S_k)
18 |   else if level(prevnode) = k and isinitial(node) then
19 |   |   tmpnode ← tmpnode ← descend(node, S_{k-1}, I_{k-1});
20 |   |   k ← k − 1 ;
21 |   else
22 |   |   nextarc ← getnextarc(node, prevnode, I_k);
23 |   |   tmpnode ← traversearc(nextarc, node, I_k)
24 |   prevnode ← node;
25 |   node ← tmpnode;
26 return node
```

Tout d'abord, pour un PCP $\mathcal{S}_N$ et une stratégie jointe arbitraire ω^0, nous initialisons les sous PCP $\mathcal{S}_k$ (ligne 2)

Le noeud complémentaire du niveau 1 est calculé (ligne 3).

Le chemin sera parcouru jusqu'à ce que le niveau k actuel corresponde au niveau N et que le noeud actuel soit complémentaire (ligne 5).

A la ligne 8, lorsque le noeud qui précède le noeud actuel est au même niveau ou au niveau inférieur du niveau actuel, cela signifie que le noeud est complémentaire ou complémentaire initial et qu'il a été atteint "normalement", donc une montée est effectuée. Autrement, à la ligne 11, le noeud qui précède le noeud complémentaire est à un niveau supérieur, ce qui veut dire qu'une descente était le dernier "déplacement" effectué. Nous effectuons une descente si le noeud actuel est complémentaire initial (ligne 12) ou bien une traversé d'arc normale autrement (ligne 15).

Si un noeud initial vient d'être atteint en venant d'un noeud du même niveau alors une descente est effectuée (ligne 18).

Lorsque le noeud est un noeud ω^0-presque complémentaire "normal" alors une traversé d'arc est effectuée (ligne 21)

Exemple : Un exemple d'exécution complet dans un jeu non dégénéré est illustrée dans l'annexe B.

5.6 Extension aux jeux dégénérés et jeux hypergraphiques

5.6.1 Jeux dégénérés

Comme évoqué dans la définition 35, il y a deux façons pour un PCP à un niveau k d'être dégénéré : (i) s'il y a des points satisfaisant plus de $|I_k|$ équations et (ii) s'il y a des systèmes qui admettent plus d'une solution.

Dans le premier cas, plus fréquent que le second en pratique, il est possible qu'une erreur se produise dans l'algorithme de traversée d'un arc (algorithme 1). Quand on suit un arc $\gamma^k(Z, W)$, lors de la traversée d'un arc ou la montée vers un noeud initial, on rencontre en temps normal un seul noeud faisable presque complémentaire qui est soit $\rho^k(Z \cup \{z\}, W)$, soit $\rho^k(Z, W \cup \{w\})$ (ici z et w sont des paires (ν, j)). Cependant, pour certains PCP, il est possible que la traversée d'un arc sature plus d'une nouvelle contrainte. Par exemple, il est possible qu'il existe w_1 et w_2 tels que $\rho^k(Z, W \cup \{w_1\})$ et $\rho^k(Z, W \cup \{w_2\})$ définissent tous les deux la même solution faisable (les coordonnées des noeuds sont identiques). Un tel noeud est dégénéré par définition. Alors que le passage par des noeuds non dégénérés ne laisse aucune ambiguïté, il y a une ambiguïté lorsque l'on rencontre des noeuds dégénérés puisqu'ils sont voisins d'au minimum trois arcs presque complémentaires. Lorsque c'est le cas, le choix arbitraire d'un arc suivant peut potentiellement amener l'algorithme dans un cycle. La figure 5.5 montre un exemple avec des noeuds dégénérés et des chemins les reliants.

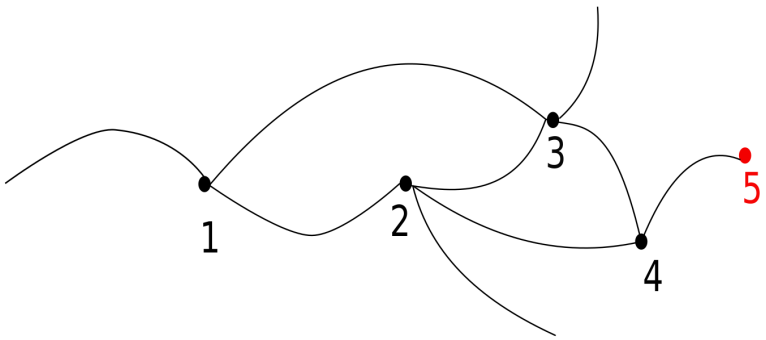


FIGURE 5.5 – Un PCP où 4 noeuds presque complémentaires dégénérés sont rencontrés. Le noeud 5 est complémentaire

Dans cet exemple, les noeuds 1,2,3 et 4 sont dégénérés. Le graphe contient des cycles ($\{1, 2, 3\}$, $\{2, 3, 4\}$, $\{1, 2, 4, 3\}$). Même dans un cas dégénéré, le graphe contient toujours au moins un chemin partant du noeud initial au niveau 1 vers le noeud complémentaire au niveau N. C'est une conséquence du fait qu'une perturbation infinitesimale des coefficients du PCP résoudrait le problème de dégénérescence en rendant certaines

contraintes non saturées. Ce faisant, certains arcs engendrant la dégénérescence seraient supprimés. En tenant compte de cette propriété, il est possible de modifier l'algorithme de parcours de chemin en utilisant une méthode de recherche en "profondeur d'abord" sur les arcs partant des noeuds dégénérés, comme le montre l'algorithme 3 :

Algorithm 3: PARCOURS DE GRAPHE EN PROFONDEUR D'ABORD

Data: Un PCP et un premier noeud presque complémentaire dégénéré, $firstdegennode$
Result: Un noeud complémentaire

```
1 listofarcs ← ∅;
2 listofarcs.append(firstdegennode.outgoingarcs);
3 while listofarcs≠ ∅ do
4 |   currentarc ← listofarcs.pop();
5 |   nextnode ← traversepath(currentarc);
6 |   if nextnode.is_complementary then break;
7 |   if nextnode = ∅ then currentarc ← listofarcs.pop();
8 |   else listofarcs.append(nextnode.outgoingarcs);
9 return nextnode
```

La procédure $traversepath()$ effectue le parcours d'un chemin presque complémentaire ne comportant que des noeuds non dégénérés, jusqu'à que (i) le noeud complémentaire soit atteint, (ii) un nouveau noeud dégénéré soit rencontré ou (iii) un noeud dégénéré déjà rencontré soit atteint. Dans les cas (i) et (ii) la variable *nextnode* est mise à la valeur du noeud correspondant, alors que dans le cas (iii), elle prend la valeur $\varnothing$.

A chaque fois qu'un noeud dégénéré est atteint, les arcs sortants (il y en a plusieurs, puisque le noeud est dégénéré) sont ajoutés à la liste $listofarcs$ (lignes 2 et 8). Le prochain arc traversé est le dernier stocké dans $listofarcs$ (d'où la recherche en profondeur) et celui-ci est retiré de la liste (lignes 4 et 7).

Proposition 24. *Dans le cas d'un PCP dégénéré du premier type, l'algorithme* 2 *retourne un noeud complémentaire.*

Preuve (Proposition 24)**.** *Puisqu'il existe un chemin depuis le noeud initial au niveau* 1 *vers un noeud complementaire au niveau N, le parcours en profondeur d'abord finira par le trouver.*

Exemple 36. *En appliquant l'algorithme 3 à l'exemple de la figure 5.5, la séquence de piles d'arcs suivante peut être obtenue*[3] *:*
$\{(1, up), (1, down)\} \rightarrow \{(1, up), (2, up), (2, middle), (2, bottom)\} \rightarrow$
$\{(1, up), (2, up), (2, middle)\} \rightarrow \{(1, up), (2, up), (4, up), (4, bottom)\} \rightarrow$ *Le noeud 5 est retourné.*

Dans le cas des LCP (pour les jeux bimatriciels et polymatriciels), une heuristique utilisée pour ordonner les "arcs à explorer" est connue et permet à l'algorithme 3 de ne jamais faire de retour en arrière [Von Stengel, 2002]. Cette heuristique, basée sur une perturbation lexicographique des systèmes linéaires, est utile car elle permet d'éviter les parcours d'arcs non nécessaires, qui sont coûteux. Le même genre d'approche peut potentiellement être appliqué sur les PCP. Cependant, calculer cette heuristique pour

3. Des séquences différentes peuvent être obtenues avec une implémentation différente de la méthode *append*.

un PCP demanderait de résoudre des systèmes polynomiaux, donc le bénéfice de cette approche n'est pas clair.

Le second cas de dégénérescence mentionné correspond à une situation où certaines variétés $\mathcal{V}\left(\mathcal{S}_k^{Z,W}\right)$ rencontrées lors de l'exécution de l'algorithme sont de dimension strictement positive (un arc, une hypersurface...) parce que les équations les définissant sont redondantes. Bien que l'utilisation des bases de Gröbner permette de représenter de telles variétés de dimension positive, le moyen par lequel ces variétés peuvent être traversées lors de la procédure reste à définir. Dans le cas où une de ces variétés correspond à un noeud complémentaire au niveau N, il est possible de fournir directement l'ensemble des solutions du PCP sous forme paramétrée (la base de Gröbner). Ce qu'il est envisageable de faire dans ce cas de dégénérescence est d'appliquer une petite perturbation aléatoire aux coefficients du PCP initial et de résoudre ce nouveau PCP perturbé. Ce "nouveau" PCP perturbé ne sera pas "en général" dégénéré. En faisant ceci, pour ce type de cas dégénéré, un équilibre de Nash *approché* est obtenu.

5.6.2 Jeux hypergraphiques

Le calcul d'équilibre de Nash basé sur les PCP s'étend naturellement aux jeux polymatriciels [Yanovskaya, 1968], aux jeux graphiques [Kearns *et al.*, 2001] et aux jeux hypergraphiques [Papadimitriou et Roughgarden, 2008].

Nous rappelons la définition d'un jeu hypergraphique (ici avec des désutilité) $\Gamma^N = \langle S, \Omega, E, a\rangle$ où $a^g = (a_n^g(\omega_g))_{n\in S_g}$ avec $a_n^g(\omega_{S_g})$ la désutilité du joueur n dans le jeu local g pour l'action jointe $\omega_g \in \Omega_{S_g}$ avec $g \in \{1, ..., |E|\}$ et où $S_g \subseteq S$ est un sous ensemble de joueurs.

Puisqu'un jeu hypergraphique peut être représenté comme un jeu sous forme normale (qui prendra, potentiellement, exponentiellement plus d'espace à représenter), le calcul d'un équilibre de Nash dans un jeu hypergraphique admet une formulation en PCP de taille exponentielle. Heureusement, il est possible d'exploiter la factorisation des fonctions de désutilité du jeu hypergraphique afin de calculer le PCP correspondant d'une taille "raisonnable". Dans un premier temps je montre que :

Proposition 25 (Factorisation du PCP)**.** *Soit Γ^N un jeu hypergraphique, pour $n \in S$ et $i \in \Omega_n$:*

$$A_i^n(x^{-n}) = \sum_{g, n\in g} Q_g\left(x^{S\setminus S_g}\right) R_{n,i,g}\left(x^{S_g\setminus\{n\}}\right), \text{ où}$$

$$\begin{aligned} Q_g\left(x^{S\setminus S_g}\right) &= \prod_{\nu\in S\setminus S_g}\left(\sum_{\omega_\nu\in S_\nu} x_{\omega_\nu}^\nu\right) \text{ et} \\ R_{n,i,g}\left(x^{S_g\setminus\{n\}}\right) &= \sum_{\substack{\omega_{S_g}\\ \omega_n=i}} a_n^g(\omega_{S_g}) \prod_{\nu\in S_g\setminus\{n\}} x_{\omega_\nu}^\nu. \end{aligned}$$

Preuve (Proposition 25)**.** *La désutilité d'un joueur $n \in S$ pour la stratégie jointe ω est :*

$$a_n(\omega) = \sum_{g, n\in S_g} a_n^g(\omega_{S_g})$$

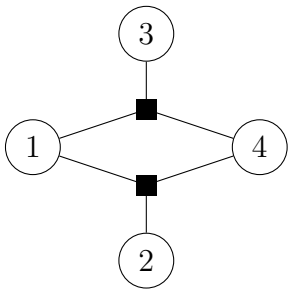
Utilisant cette expression de a^n_ω, il est possible de réécrire $A^n_i(x^{-n})$:

$$\begin{aligned} A^n_i(x^{-n}) &= \sum_{\substack{\omega\in\pi\\ \omega_n=i}} \left(\sum_{g,n\in S_g} a^g_n(\omega_{S_g}) \right) \prod_{\nu\neq n} x^\nu_{\omega_\nu}, \forall n \in S, i \in \Omega_n, \\ &= \sum_{g,n\in S_g} \sum_{\substack{\omega\in\pi\\ \omega_n=i}} a^g_n(\omega_{S_g}) \prod_{\nu\neq n} x^\nu_{\omega_\nu}, \\ &= \sum_{g,n\in S_g} \sum_{\substack{\omega_{S_g}\\ \omega_n=i}} a^g_n(\omega_{S_g}) \left(\prod_{\nu\in S_g\setminus\{n\}} x^\nu_{\omega_\nu} \right) \sum_{\omega_{S\setminus S_g}} \prod_{\nu\in S\setminus S_g} x^\nu_{\omega_\nu}, \\ &= \sum_{g,n\in S_g} \sum_{\substack{\omega_{S_g}\\ \omega_n=i}} a^g_n(\omega_{S_g}) \left(\prod_{\nu\in S_g\setminus\{n\}} x^\nu_{\omega_\nu} \right) \left(\prod_{\nu\in S\setminus S_g} \left(\sum_{\omega_\nu\in\Omega_\nu} x^\nu_{\omega_\nu} \right) \right). \end{aligned}$$

Remarquez que l'égalité $\sum_{\omega_{S\setminus S_g}} \prod_{\nu\in S\setminus S_g} x^\nu_{\omega_\nu} = \prod_{\nu\in S\setminus S_g} \left(\sum_{\omega_\nu\in\Omega_\nu} x^\nu_{\omega_\nu} \right)$

est le résultat d'une application répétée de la distributivité de la multiplication par rapport à l'addition. La proposition en est la conséquence.

Exemple 37 (Factorisation d'un HGG)**.** *Prenons un jeu hypergraphique à quatre joueurs avec deux jeux locaux à trois joueurs. Pour le graphe suivant*



Avec les deux tables d'utilités qui suivent :

ω_1	ω_2	ω_4	u_1	u_2	u_4
0	0	0	74	51	18
0	0	1	6	97	16
0	1	0	9	98	29
0	1	1	60	83	86
1	0	0	77	90	100
1	0	1	30	1	96
1	1	0	47	2	93
1	1	1	0	47	64

FIGURE 5.6 – Jeu local entre les joueurs $\{1, 2, 4\}$

ω_1	ω_3	ω_4	u_1	u_3	u_4
0	0	0	84	10	1
0	0	1	48	88	36
0	1	0	27	72	68
0	1	1	5	47	0
1	0	0	100	13	53
1	0	1	62	24	5
1	1	0	16	42	61
1	1	1	62	72	97

FIGURE 5.7 – Jeu local entre les joueurs $\{1, 3, 4\}$

$$
\begin{aligned}
A_0^1\left(x^2,x^3,x^4\right) &= (27x_0^2x_0^4+95x_0^2x_1^4+92x_1^2x_0^4+41x_1^2x_1^4)*(x_0^3+x_1^3)+\\
&\quad (17x_0^3x_0^4+53x_0^3x_1^4+74x_1^3x_0^4+96x_1^3x_1^4)*(x_0^2+x_1^2)\\
A_1^1\left(x^2,x^3,x^4\right) &= (24x_0^2x_0^4+71x_0^2x_1^4+54x_1^2x_0^4+101x_1^2x_1^4)*(x_0^3+x_1^3)+\\
&\quad (x_0^3x_0^4+39x_0^3x_1^4+85x_1^3x_0^4+39x_1^3x_1^4)*(x_0^2+x_1^2)\\
A_0^2\left(x^1,x^3,x^4\right) &= (48x_0^1x_0^4+2x_0^1x_1^4+9x_1^1x_0^4+98x_1^1x_1^4)*(x_0^3+x_1^3)\\
A_1^2\left(x^1,x^3,x^4\right) &= (x_0^1x_0^4+16x_0^1x_1^4+97x_1^1x_0^4+52x_1^1x_1^4)*(x_0^3+x_1^3)\\
A_0^3\left(x^1,x^2,x^4\right) &= (79x_0^1x_0^4+x_0^1x_1^4+76x_1^1x_0^4+65x_1^1x_1^4)*(x_0^2+x_1^2)\\
A_1^3\left(x^1,x^2,x^4\right) &= (17x_0^1x_0^4+42x_0^1x_1^4+47x_1^1x_0^4+17x_1^1x_1^4)*(x_0^2+x_1^2)\\
A_0^4\left(x^1,x^2,x^3\right) &= (83x_0^1x_0^2+72x_0^1x_1^2+x_1^1x_0^2+8x_1^1x_1^2)*(x_0^3+x_1^3)+\\
&\quad (100x_0^1x_0^3+33x_0^1x_1^3+48x_1^1x_0^3+40x_1^1x_1^3)*(x_0^2+x_1^2)\\
A_1^4\left(x^1,x^2,x^3\right) &= (85x_0^1x_0^2+15x_0^1x_1^2+5x_1^1x_0^2+37x_1^1x_1^2)*(x_0^3+x_1^3)+\\
&\quad (65x_0^1x_0^3+101x_0^1x_1^3+96x_1^1x_0^3+4x_1^1x_1^3)*(x_0^2+x_1^2)
\end{aligned}
$$

Notez que $A_i^n(x^{-n})$ est toujours un polynôme de variables x_j^m, $(m,j)\neq(n,i)$ avec un nombre de termes de dégré $N-1$ égale à $|\Omega_{-n}|$.

Néanmoins, nous avons montrés qu'il est représentable de façon factorisée. Dans l'annexe C, nous proposons une formulation du PCP d'un jeu hypergraphique de taille polynomiale en la taille du jeu, grâce à l'ajout de variables supplémentaires. Néanmoins, en pratique, cette formulation n'apporte pas un gain (les logiciels de calcul de base de Gröbner exploitant la forme développée des polynômes).

5.7 Conclusion et Perspectives

5.7.1 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle méthode combinatoire et algébrique de calcul d'un équilibre de Nash mixte (exact). Cette méthode est basée sur l'approche géométrique de parcours de chemin définie théoriquement par [Wilson, 1971]. Nous proposons aussi un algorithme implémentant cette méthode. Nous étendons cette méthode aux jeux dégénérés et aux jeux hypergraphiques.

5.7.2 Perspectives

Les travaux présentés dans ce chapitre ouvrent de nombreuses perspectives, sur la résolution des jeux dégénérés ou hypergraphiques en particulier.

L'algorithme implémenté utilise un solveur "exact", ce qui peut rendre l'exécution lente. On peut utiliser des solveurs "approchés" efficaces pour résoudre les systèmes lors des traversées d'arc. Lorsqu'une traversée amène à plusieurs noeuds non distinguables, nous devons tout de même utiliser la résolution exacte des systèmes avec les bases de Gröbner pour vérifier s'il y a bien dégénérescence.

Comme mentionné précédemment, il existe un cas de dégénérescence où nous avons un noeud de dimension strictement positive. C'est-à-dire que le noeud $\rho^k(Z,W)$ (resp. $\gamma^k(Z,W)$) est une partie d'une variété affine de dimension supérieure à 0 (resp. supérieure à 1). On peut toujours, dans ce cas, faire entrer d'autres paires (n,i) dans Z où W. Puis de "poursuivre" le chemin parcouru. Une approche formelle pour le choix du couple reste à écrire.

Plusieurs aspects de la résolution des jeux hypergraphiques peuvent être étudiés. Compte tenu du calcul de l'utilité (désutilité) espérée d'un jeu hypergraphique, il serait intéressant de pouvoir écrire un PCP avec des polynômes de dégré inférieur à la taille des jeux locaux du jeu hypergraphique sans ajouter de variables supplémentaires. Ce qui permet d'espérer que cela est possible, c'est l'existence d'une modélisation par un LCP des jeux polymatriciels (à N joueurs). Cette piste de recherche semble prometteuse.

Il peut aussi être intéressant d'étudier l'utilisation de la structure de l'hypergraphe et d'un ordre sur les joueurs pour obtenir des polynômes de degré inférieur lorsque nous explorons des sous-PCP (de dégré inférieur à $k-1$ lorsque nous sommes au niveau k). La conception d'une heuristique d'ordonnancement des niveaux des joueurs semble accessible facilement.

Un autre aspect intéressant à explorer serait l'utilisation d'un algorithme pour la résolution de jeux stochastiques. Il s'agirait d'intégrer des méthodes de programmation dynamique à l'algorithme présenté dans ce chapitre.

Chapitre 6

Jeux de Conservation

6.1 Introduction

En ce qui concerne le cadre d'application de ces travaux, nous nous sommes intéressé à un problème de protection de la biodiversité, dans une situation de lutte contre le braconnage.

Les agents sont non coopératifs, non communicants et il faut gérer des informations partielles sur les préférences des autres agents. Le modèle proposé est celui des *"jeux de conservation à information incomplète"* (ICG).

Le modèle proposé pour l'étude des stratégies de conservation de la biodiversité est à la fois "puissant" et "efficace". Il est "puissant" car il permet de représenter des problèmes de conservation complexes, avec potentiellement de nombreux agents non coopératifs, non communicants, ainsi qu'avec une connaissance partielle sur l'état des ressources naturelles. Il est efficace car il permet une représentation concise du problème, dans le cadre des *jeux bayésiens polymatriciels*, lorsque les interactions entre les agents sont exclusivement deux à deux.

Nous allons tout d'abord mentionner quelques exemples d'utilisation de la théorie des jeux dans des problèmes de conservation de la biodiversité. Ensuite nous présenterons un modèle simple et générique de jeu de conservation où plusieurs *braconniers* et *gardes forestiers* attaquent et défendent un ensemble de ressources de biodiversité. Nous montrerons que ces jeux sont représentables de manière concise par des jeux polymatriciels.

Dans des problèmes de conservation réalistes, la connaissance des ressources de biodiversité est souvent incomplète. Le modèle sera enrichi afin de permettre la modélisation d'une connaissance partielle dans les jeux de conservation. Nous proposons un modèle bayésien des jeux de conservation où les braconniers ont une information partielle sur la ressource de biodiversité. Nous montrerons ensuite que ce problème peut être formulé sous la forme d'un jeu bayésien polymatriciel.

6.2 Théorie des jeux pour la conservation de la biodiversité

6.2.1 Jeux sous forme normale et conservation

Plusieurs travaux se sont déjà intéressés à l'utilisation des jeux sous forme normale afin d'étudier des problèmes de conservation, en général liés à la chasse.

[Redpath *et al.*, 2018] s'intéressent à l'utilisation de la théorie des jeux dans des problèmes de conservation de la biodiversité. Cet article présente différents types d'approches possibles, théoriques, expérimentales et constructives, leurs avantages, inconvénients et objectifs. Une description de la pertinence des approches en fonction des objectifs du problème à formuler est faite. Un exemple présente comment les trois approches peuvent être appliqué pour l'étude de l'impact des oies sur l'agriculture en Suède (les objectifs étant différents). Quelques questions sur les limites et l'éthique d'utilisation des jeux sont aussi soulevées.

[Colyvan *et al.*, 2011] s'intéressent à la modélisation de trois types de problèmes de conservation. Le premier type concerne la protection d'espèces ayant une population partagée entre différents pays, demandant donc une coopération sur la mise en place d'un plan de conservation. Dans le second type, un ensemble d'acteurs partagent une ressource qu'ils "moissonnent"/"consomment" dans une zone commune. La consommation de cette ressource par l'un des acteurs diminue la consommation potentielle des autres, comme c'est le cas dans les problèmes de braconnage. Le troisième type de problème abordé consiste à modéliser des agents "humains" (organisations, gouvernements, etc) jouant contre un agent représentant la "nature". La "nature" n'est pas toujours prévisible et peut réagir de manière inattendue aux actions d'un humain (comme aux méthodes de marquage qui peut changer les comportements des animaux). Elle est modélisée sous la forme d'un agent non coopératif. Par exemple, en hydrologie, lorsqu'un joueur humain souhaite construire un lac artificiel, il joue contre la nature qui "souhaite" faire descendre le niveau de l'eau au plus bas.

Des travaux plus proches des nôtres visent des problèmes de recherche d'équilibre de Nash pour des problèmes de décisions pour la conservation comme dans [Gibson et Marks, 1995] et [Frank et Sarkar, 2010].

Dans [Gibson et Marks, 1995], les raisons des échecs des différentes stratégies de lutte contre le braconnage en Afrique sont étudiées en modélisant le problème de braconnage sous forme de jeu (sous forme extensive). L'étude a permis de déterminer que le manque de succès des stratégies utilisées était lié à une mauvaise interprétation des causes économiques, politiques et sociales de la chasse locale. Ce problème est modélisé comme un jeu sous forme extensive à 2 joueurs où le premier joueur, un groupe de gardes forestiers, a le choix entre faire appliquer ou non la réglementation ; le second groupe de joueurs, les chasseurs locaux, ont trois actions, chasser du gros gibier, du petit gibier ou ne pas chasser. Via une modélisation des différentes stratégies mises en place pour lutter contre le braconnage, le papier montre que les programmes mis en place pour lutter contre le braconnage ne comprennent pas les chasseurs et leur relations vis-à-vis de la faune locale.

Dans [Frank et Sarkar, 2010], plusieurs problèmes de gestion de la faune sont modélisés. L'un d'eux est un conflit entre des défenseurs de l'environnement et des gardiens de troupeaux, concernant la gestion de chiens sauvages en Afrique du Sud (qui s'échappent des réserves naturelles). Les premiers ont le choix entre continuer ou pas le déplacement des populations de chiens sauvages, les seconds ont le choix entre tuer ou ne pas tuer les chiens sauvages qui se sont échappés. Ce problème est modélisé sous la forme d'un dilemme du prisonnier (à deux joueurs). Un autre conflit est présenté, entre 3 acteurs : des gardes-chasses (également chasseurs de Lagopède d'Écosse), des protecteurs de Busard Saint-Martin et des protecteurs d'aigle royal. Chacun des joueurs souhaite privilégier ses intérêts via des actions pouvant entraîner une "baisse" des populations des espèces que les autres joueurs "préfèrent". Ceci se

modélise par un jeu sous forme normale à 3 joueurs. Le troisième problème s'intéresse à un conflit entre des pêcheurs de poissons dans un récif de corail. Chaque pêcheur a le choix entre deux actions, pêcher autant que possible ou ne pas dépasser un quota, pour un maintien durable des niveaux de production. Ce jeu est modélisé sous la forme d'un jeu symétrique à N joueurs où l'utilité d'un joueur dépend (de son action et) du nombre de joueurs qui décident d'effectuer le plus de pêche possible. Les analyses de ces jeux se font notamment via l'identification d'équilibres Pareto optimaux et d'incitations à faire sur leurs différents joueurs pour changer les utilités (et donc changer les équilibres).

6.2.2 Jeux meneur-suiveur et conservation

D'autres travaux s'intéressent à des problèmes de conservation via l'utilisation de jeux meneur suiveur (leader-follower) [Osborne et Rubinstein, 1994].

Les Green Security Game [Yang *et al.*, 2014, Fang *et al.*, 2015, Fang *et al.*, 2016, Fang *et al.*, 2017], en particulier, ont le même cadre applicatif que nous. C'est un modèle relativement récent qui est issu des "security games", eux même issus des jeux de Stackelberg [Von Stackelberg, 1934, Osborne et Rubinstein, 1994].

Dans les jeux de Stackelberg, il y a une "hiérarchie" dans l'ordre des actions jouées par les joueurs. Un meneur joue en premier (un leader dans un marché rend sa stratégie publique) et ensuite c'est au tour des suiveurs de jouer (les autres participants du marché réagissent à cette stratégie). Dans ces jeux ce n'est pas un équilibre de Nash qui est recherché mais un équilibre de Stackelberg. C'est-à-dire un équilibre parfait en sous-jeu ; ici cela signifie que le premier joueur (meneur) joue une stratégie qui, selon les meilleurs réponses des "seconds" joueurs (les suiveurs) à sa stratégie, maximise son utilité [Osborne et Rubinstein, 1994].

Les Security games [Kiekintveld *et al.*, 2009, Tambe, 2011] (parfois appelés Stackelberg Security games) sont une classe de jeux de Stackelberg qui s'intéressent à la modélisation et la résolution de problèmes de "sécurité". Par exemple des problèmes de patrouilles dans un aéroport [Pita *et al.*, 2008], métro [Yin *et al.*, 2012] ou port [Fang *et al.*, 2013]. Dans ce type de jeu, nous identifions un défenseur (un meneur) et un attaquant (un suiveur). Les green security games [Fang *et al.*, 2015] modélisent des problèmes de lutte contre le braconnage via des security games.

Nous nous intéressons à un cadre diffèrent des jeux meneur suiveur et de la recherche des équilibres de Stackelberg. Nous allons néanmoins reprendre une situation similaire à celle présentée dans le cadre des Green Security games. Nous considérons un groupe d'attaquants et de défenseurs, mais nous faisons l'hypothèse que les défenseurs et les attaquants jouent de manière simultanée. Nous avons donc un jeu sous forme normale.

6.3 Jeux de conservation à information complète

Comme mentionné précédemment nous nous intéressons à la modélisation de problèmes de conservation de la biodiversité, plus particulièrement à des problèmes de surveillance et protection de réserves naturelles par des gardes forestiers, contre des attaques de braconniers sur la faune.

Nous allons tout d'abord définir un jeu de conservation, puis le jeu sous forme normale équivalent à ce dernier.

6.3.1 Jeux de conservation

Un jeu de conservation est un jeu impliquant un ensemble d'attaquants et de défenseurs, sur un ensemble de sites contenant une certaine quantité de ressources écologiques. Le but des attaquants est d'obtenir le plus de ressources possible, tandis que les défenseurs veulent arrêter/entraver le plus d'attaquants possibles.

Définition 45 (Jeu de conservation)**.** *Un jeu de conservation est défini par un triplet* $CG(\lambda, L, K)$ *:*

- $\lambda = (\lambda_1, \ldots, \lambda_M)$ *est un vecteur de* M *entiers, représentant les niveaux de ressources pour chacun des* M *sites du jeu.*
- L *est le nombre d'attaquants dans le jeu de conservation,* $l \in \{1, ..., L\}$ *est un attaquant individuel.*
- K *est le nombre de défenseurs dans le jeu de conservation,* $k \in \{L+1, ..., L+K\}$ *est un défenseur individuel.*

Dans un jeu de conservation, tous les attaquants et défenseurs choisissent simultanément les sites qu'ils vont visiter et récupèrent ensuite leur récompense (ou pénalité) en fonction des autres joueurs visitant le même site. Ces choix amènent à une interaction entre les attaquants et les défenseurs.

Exemple 38 (Jeu de conservation)**.** *La figure 6.1 montre la structure des interactions entre les joueurs et les ressources résultant des choix des attaquants et défenseurs dans un jeu où* $L = K = 3$ *et* $\lambda = (5, 3, 4)$*, les attaquants choisissent respectivement d'attaquer les sites* 1*,* 3 *et* 3 *et les défenseurs de défendre les sites* 1*,* 3 *et* 2*.*

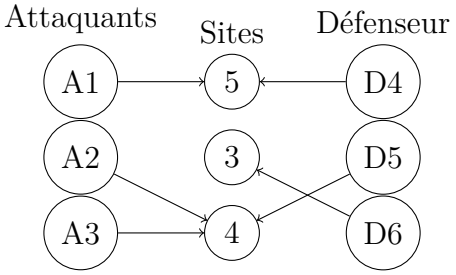


FIGURE 6.1 – Représentation graphique d'un jeu de conservation où $L = 3$, $K = 3$ et $\lambda = (5, 3, 4)$.

Les attaquants ont une récompense correspondant au niveau de ressource présent sur le site qu'ils attaquent, duquel sera soustrait le nombre de défenseurs qu'ils rencontrent dans ce même site (ils sont pénalisés). Une fonction de pénalité plus complexe, basée sur la valeur marchande des ressources et les montants des amendes encourues (et de la valeur du matériel confisqué) peut être envisagée, sans altérer le modèle. Seuls les attaquants sont intéressés par la valeur des sites. Celle-ci peut servir à représenter le bénéfice (économique notamment) associé aux ressources qu'il est possible de braconner dans un site particulier. Par exemple s'il y a des espèce "convoitées" par les braconniers, comme des éléphants ou des pangolins, on peut considérer que le site aura une valeur élevée. La valeur soustraite pour chaque défenseur rencontré peut correspondre à une valeur "économique" perdue (amende, demande de ressource plus élevée pour répondre à la présence de défenseurs, piège rendu inefficace par les défenseurs, etc).

Les défenseurs, eux, reçoivent une récompense pour chaque attaquant qu'ils rencontrent sur le site qu'ils ont choisi de visiter, indépendamment de la ressource de ce dernier (encore une fois cette récompense peut être complexifiée).

6.3.2 Forme normale d'un jeu de conservation

Avec cette interprétation, un jeu de conservation, défini par $CG(\lambda, L, K)$, peut être modélisé comme un jeu sous forme normale $G = \langle S, \Omega, u \rangle$:

Définition 46 (Jeu de conservation sous forme normale)**.** *Tout jeu $CG = (\lambda, L, K)$ définit un jeu sous forme normale $G = \langle S, \Omega, u \rangle$*

- *$S = \{1, \ldots, N\}$ est l'ensemble des joueurs de G (où $N = L + K$), $S = \{1, \ldots, L, L+1, \ldots, L+K\}$.*
- *$\Omega = \Omega_1 \times \ldots \times \Omega_N$ où $\Omega_n = \{1, \ldots, M\}, \forall n \in S$ est l'ensemble des stratégies jointes du jeu. Tout joueur (attaquant ou défenseur) choisit un site à visiter. Par exemple, dans la figure 6.1, la stratégie jointe pure $\omega = (1, 3, 3, 1, 3, 2)$ est représentée.*
- *$u = (u_n(\omega))_{n=1..N, \omega \in \Omega}$ représente les tables d'utilité des joueurs. $u_n(\omega)$ est l'utilité du joueur n lorsque la stratégie ω est jouée.*

Dans un jeu de conservation, comme mentionné précédemment, la définition des utilités d'un joueur sera différente pour un attaquant ou un défenseur. Dans le cas simple que nous considérons, elles sont définies de la manière suivante :

$$\textbf{Attaquants :}\ u_l(\omega) = \lambda_{\omega_l} - c_l \times \left(\sum_{k=L+1}^{L+K} \delta(\omega_l, \omega_k) \right), \forall l = 1, ..., L \tag{6.1}$$

$$\textbf{Défenseurs :}\ u_k(\omega) = \sum_{l=1}^{L} \delta(\omega_l, \omega_k), \forall k = L+1, ..., L+K \tag{6.2}$$

où $\delta(\omega_l, \omega_k) =_{def} 1$ si $\omega_l = \omega_k$ et $\delta(\omega_l, \omega_k) =_{def} 0$ si $\omega_l \neq \omega_k$. c_l est le coût d'une interaction d'un attaquant avec un défenseur (amende).

Par la suite nous ferons une distinction entre les joueurs lorsqu'ils sont attaquant (l) ou défenseur (k).

Les équations 6.1 et 6.2 correspondent à la définition intuitive des utilités décrites précédemment(avec $c_l = 1$). Pour les attaquants (équation 6.1), l'utilité correspond à la valeur de la ressource du site choisi auquel on soustrait c_l fois le nombre de défenseurs sur ce site. Pour les défenseurs (équation 6.2), l'utilité correspond au nombre d'attaquants présents sur le site choisi.

Exemple 39 (Jeu de conservation sous forme normale)**.** *Prenons un plus petit exemple avec 2 sites, 2 attaquants et 2 défenseurs où $\lambda = (5, 3)$, $L = 2$ et $K = 2$. La figure 6.2 représente le graphe du jeu de conservation avec toutes les stratégies jouables et la figure 6.3 montre la table d'utilité du jeu sous forme normale correspondant. Dans la suite, nous considérons que $c_l = 1$ pour limiter les notations*

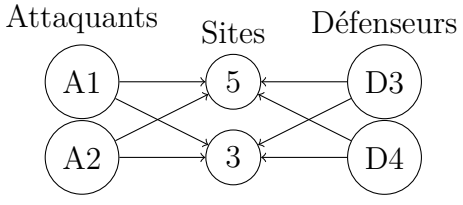


FIGURE 6.2 – Interactions dans un jeu de conservation avec $L = 2$,$K = 2$ et $\lambda = (5, 3)$

ω_1	ω_2	ω_3	ω_4	u_1	u_2	u_3	u_4
1	1	1	1	3	3	2	2
1	1	1	2	4	4	2	0
1	1	2	1	4	4	0	2
1	1	2	2	5	5	0	0
1	2	1	1	3	3	1	1
1	2	1	2	4	2	1	1
1	2	2	1	4	2	1	1
1	2	2	2	5	1	1	1
2	1	1	1	3	3	1	1
2	1	1	2	2	4	1	1
2	1	2	1	2	4	1	1
2	1	2	2	1	5	1	1
2	2	1	1	3	3	0	0
2	2	1	2	2	2	0	2
2	2	2	1	2	2	2	0
2	2	2	2	1	1	2	2

FIGURE 6.3 – Table d'utilités du jeu de conservation où $L = 2$,$K = 2$ et $\lambda = (5, 3)$

Il existe trois équilibres de Nash purs $\omega = (1, 1, 1, 1)$, $\omega = (1, 2, 1, 1)$ *et* $\omega = (2, 1, 1, 1)$.

Le premier équilibre peut être interprété de la manière suivante : Étant donné que le site 1 *a plus de ressources que le site* 2*, il est plus intéressant de l'attaquer, mais donc aussi de le défendre. Compte tenu du nombre de ressources et de défenseurs, même si tous les défenseurs vont sur le site la récompense sera, au pire, égale à celle du site 2.*

Notez que l'équilibre $\omega = (1, 1, 1, 1)$ *est aussi une stratégie Pareto optimale, contrairement aux deux autres équilibres.*

Si au lieu de $c_l = 1$*, nous avons* $c_l = 2$ *alors il n'y a plus d'équilibre pur.*

6.3.3 Représentation par un jeu polymatriciel

Dans cette sous section, nous montrerons qu'il est possible d'exprimer le jeu décrit précédemment sous la forme d'un jeu polymatriciel équivalent $PG = (S, \Omega, E, u)$.

Dans ce jeu équivalent, tous les attaquants jouent à des jeux bimatriciels simultanés avec tous les défenseurs, et les défenseurs jouent à des jeux bimatriciels simultanés avec tous les attaquants. Il y a donc un total de $L * K$ jeux locaux bimatriciels.

En d'autre termes, il existe une représentation sous forme d'un jeu polymatriciel $PG = (S, \Omega, E, u)$ équivalente au jeu sous forme normale $G = (S, \Omega, u)$ correspondant au jeu de conservation $CG(\lambda, L, K)$.

Proposition 26 (Représentation polymatricielle d'un jeu de conservation)**.** *Pour un jeu de conservation* $CG(\lambda, L, K)$*. La forme normale équivalente de* $CG(\lambda, L, K)$ *est*

elle même équivalente à un jeu polymatriciel.

$$PG = \left(S, \Omega, E = ((l,k))_{\substack{l=1..L,\\ k=L+1..L+K}}, u = \left(u_l^{(l,k)}, u_k^{(l,k)} \right)_{\substack{l=1..L,\\ k=L+1..L+K}} \right),$$

où les utilités (bimatricielles) locales sont définies, pour tout $e = (l,k) \in E$ [1] *pour* $c_l = 1$ *:*

$$u_l^{(l,k)}(\omega_l, \omega_k) = \frac{\lambda_{\omega_l}}{K} - \delta(\omega_l, \omega_k) \text{ et} \tag{6.3}$$

$$u_k^{(l,k)}(\omega_l, \omega_k) = \delta(\omega_l, \omega_k), \forall (l,k) \in \{1, \ldots, L\} \times \{L+1, \ldots, L+K\}. \tag{6.4}$$

Preuve (Proposition 26)**.** *En reprenant les équations 6.1 et 6.2 :*

$$\begin{aligned} u_l(\omega) &= \lambda_{\omega_l} - \sum_{k=L+1}^{L+K} \delta(\omega_l, \omega_k), \forall l = 1, ..., L, \\ u_k(\omega) &= \sum_{l=1}^{L} \delta(\omega_l, \omega_k), \forall k = L+1, ..., L+K. \end{aligned}$$

Puisque, en reprenant et reformulant les équations 6.3 et 6.4, nous avons $\delta(\omega_l, \omega_k) = \frac{\lambda_{\omega_l}}{K} - u_l^{(l,k)}(\omega_l, \omega_k)$ *et* $\delta(\omega_l, \omega_k) = u_k^{(l,k)}(\omega_l, \omega_k)$*, nous avons :*

$$\begin{aligned} u_l(\omega) &= \lambda_{\omega_l} - \sum_{k=L+1}^{L+K} \frac{\lambda_{\omega_l}}{K} - u_l^{(l,k)}(\omega_l, \omega_k) \\ u_l(\omega) &= \lambda_{\omega_l} - \lambda_{\omega_l} + \sum_{k=L+1}^{L+K} u_l^{(l,k)}(\omega_l, \omega_k) \\ u_l(\omega) &= \sum_{k=L+1}^{L+K} u_l^{(l,k)}(\omega_l, \omega_k) \end{aligned} \tag{6.5}$$

De la même manière on peut montrer que pour les défenseurs :

$$u_k(\omega) = \sum_{l=1}^{L} u_k^{(l,k)}(\omega_l, \omega_k) \tag{6.6}$$

Exemple 40. *Les jeux représentés précédemment dans les figures 6.1 et 6.2 ont des jeux polymatriciels correspondant qui sont respectivement entre 6 joueurs (avec 9 jeux locaux) et entre 4 joueurs (avec 4 jeux locaux).*

Les représentations graphiques de ces deux jeux sont présentées dans les figures 6.4 et 6.5. Notez que les graphes sont bipartis, selon une partition entre joueurs attaquants et défenseurs.

1. Il est possible de considérer qu'il existe des jeux entre des joueurs de même rôle, (l,l') et (k,k'), mais dans ce cas ils auront tous des matrices d'utilités nulles.

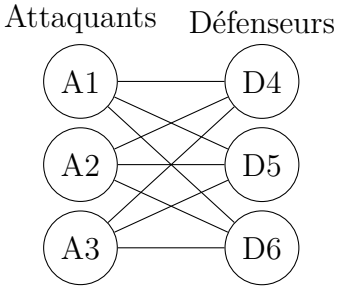


FIGURE 6.4 – Représentation polymatricielle du jeu de la Figure 6.1

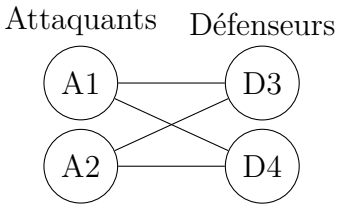


FIGURE 6.5 – Représentation polymatricielle du jeu de la Figure 6.2

Pour les deux exemples, les tables d'utilités locales sont toutes identiques pour toutes les paires $(l,k)_{l=1..L,k=L+1..L+K}$.

ω_l	ω_k	u_l	u_k
1	1	$\frac{2}{3}$	1
1	2	$\frac{5}{3}$	0
1	3	$\frac{5}{3}$	0
2	1	1	0
2	2	0	1
2	3	1	0
3	1	$\frac{4}{3}$	0
3	2	$\frac{4}{3}$	0
3	3	$\frac{1}{3}$	1

ω_l	ω_k	u_l	u_k
1	1	$\frac{3}{2}$	1
1	2	$\frac{5}{2}$	0
2	1	$\frac{3}{2}$	0
2	2	$\frac{1}{2}$	1

FIGURE 6.6 – Tables d'utilités des jeux locaux dans les jeux polymatriciels représentant le jeu de conservation de la Figure 6.1 (à gauche) et de la Figure 6.2 (à droite).

Le fait que les tables d'utilités de tous les jeux locaux soient identiques, permet un gain supplémentaire en représentation, car une seule table est nécessaire pour représenter les $L \times K$ jeux locaux. La représentation polymatricielle du jeu de conservation est ainsi très concise. Pour tous jeux de conservation $\langle \lambda, L, K \rangle$, il faut $M^{L+K} * (L+K)$ nombres (rationnels) pour représenter les utilités sous la forme normale (64 dans l'exemple 6.2) tandis que la représentation sous forme polymatricielle en demande $2 \times M^2$, en utilisant le fait que les tables d'utilités locales sont identiques (8 dans l'exemple). Si nous n'exploitions pas cette propriété il en faudrait $(2 \times M^2) \times (L \times K)$ (32 dans l'exemple). Lorsque le nombre de joueurs augmente (attaquants ou défenseurs), la différence de taille de représentation augmente exponentiellement.

6.4 Jeux de conservation avec information incomplète sur les ressources

6.4.1 Définition

Il est possible que les joueurs d'un jeu de conservation aient une connaissance incomplète du jeu. Les attaquants, en particulier, ont souvent une capacité d'observation limitée et ne connaissent pas précisément le niveau de ressource de chacun des sites. Ils ont probablement une connaissance d'un sous-ensemble limité des sites voisinant leur position actuelle (car ils en sont proches). Le cadre des jeux bayésiens (vu dans le chapitre 3) peut être utilisé pour modéliser le fait que certains joueurs ont seulement une connaissance incomplète du jeu auxquel ils jouent.

Nous montrerons comment modéliser un jeu de conservation avec information incomplète sous la forme d'un jeu bayésien. Dans un premier temps, je présente le modèle d'un jeu de conservation avec information incomplète.

Définition 47 (Jeu de conservation avec information incomplète)**.** *Un jeu de conservation avec information incomplète est un tuple* $ICG(\hat{\lambda}, L, K, \mathcal{J}, \hat{P})$*. Il comprend* L *attaquants et* K *défenseurs. Les autres éléments sont :*

- $\hat{\lambda} = \left(\hat{\lambda}_1, \ldots, \hat{\lambda}_M\right)$ *représente le niveau maximum de ressource possible pour chacun des* M *sites. Le niveau exact de ressource dans le site* $m \in \{1, .., M\}$*,* λ_m *est inconnu mais supposé entier :* $\lambda_m \in \{0, \ldots, \hat{\lambda}_m\}$*. Nous écrirons* $\Lambda = \prod_{m=1}^{M}\{0, \ldots, \hat{\lambda}_m\}$ *le domaine des potentiels vecteurs de ressource (notés* λ*, comme précédemment).*
- $\mathcal{J} = \{J_1, ..., J_L\}$ *est un ensemble de sous ensembles de sites.* $J_l \subseteq \{1, \ldots, M\}$ *est l'ensemble des sites que l'attaquant* l *observe. L'attaquant* l *connaît précisément la valeur* λ_{J_l} *du vrai vecteur ressource* λ *sur les sites de* J_l*, mais ne connaît pas les valeurs des autres sites. Par définition, si* $J_l = \{j_1, j_2, .., j_{q_l}\}$*,* $\lambda_{J_l} = (\lambda_{j_1}, \lambda_{j_2}, ..., \lambda_{j_{q_l}})$*. Pour terminer, notez que les* J_l *ne sont pas forcément disjoints et que leur union peut ne pas être égale à* $\{1, \ldots, M\}$*. Certains sites peuvent rester non observés par tous les attaquants et certains peuvent être observés par plusieurs d'entre eux.*
- $\hat{P}$ *est une distribution de probabilité jointe sur les vecteurs ressources :* $\hat{P} : \Lambda \to [0, 1]$*.* $\hat{P}$ *est une connaissance commune a priori sur les vecteurs de ressources, parmi les attaquants (les défenseurs connaissant précisément le vecteur des ressources). En partant de ces probabilités jointes, tous les attaquants* l *seront capables de calculer la probabilité des niveaux de ressource dans les sites qu'ils n'observent pas, sous la forme d'une probabilité conditionnelle* $\hat{P}(\lambda_{\overline{J_l}}|\lambda_{J_l})$*.*

On considère que les défenseurs ont une information complète sur les ressources. Ils ont plus de moyen que les attaquants et n'ont pas besoin de se "cacher". Par exemple ils peuvent effectuer des patrouilles ou effectuer d'autres formes de surveillance pour détecter précisément la quantité de ressources de chacun des sites. Les braconniers ont moins de moyens à leur disposition pour la détection et ne peuvent circuler librement. L'information qu'ils ont est incomplète.

6.4.2 Forme bayésienne d'un jeu de conservation à information incomplète.

Il est possible de formuler les jeux de conservation décrit dans la sous-section précédente sous la forme d'un jeu bayésien.

Définition 48 (Jeu de conservation bayésien)**.** *Un jeu de conservation à information incomplète* $ICG(\hat{\lambda}, L, K, \mathcal{J}, \hat{P})$ *peut être décrit par un jeu bayésien sous une forme normale* $BG = \langle S, \Omega, \Theta, P, u \rangle$*, où* $S = \{1, ..., L+K\}$*,* $\Omega = \Omega_1 \times ... \times \Omega_{L+K}$ *avec* $\Omega_n = \{1, ..., M\}$*.*

- $\Theta = \Theta_1 \times \ldots \times \Theta_L \times \Theta_{L+1} \times \ldots \times \Theta_{L+K}$ *l'ensemble des* types *joints des joueurs.*
- $P : \Theta \to [0, 1]$ *est la distribution de probabilité jointe (a priori) sur les types.*
- $u = (u_n(\omega, \theta))_{n \in S, \omega \in \Omega, \theta \in \Theta}$ *représente les utilités des joueurs.* $u_n(\omega, \theta)$ *est l'utilité obtenue par le joueur* n *lorsque l'action jointe* ω *est jouée, lorsque le type joint est* θ*.*

On montre dans la suite de cette section comment Θ, P et u sont construits.

Types joints Θ dans un jeu de conservation bayésien

Le jeu "réel" qui est joué entre les attaquants et défenseurs est défini de manière unique par λ, le vecteur ressource réel. Les joueurs ne connaissent pas λ mais ont une connaissance partielle, modélisée par le vecteur $\theta = (\theta_1, ..., \theta_N)$ de "type joints". Le *type* θ_n de tout joueur $n = 1, ..., N$ modélise la connaissance du joueur n sur le vecteur ressource. Il existe une fonction "type" $\tau : \Lambda \to \Theta$ qui spécifie le type de tous les joueurs pour un vecteur de ressources donné. τ modélise l'information que les joueurs ont sur l'état des ressources quand l'état réel est λ.

La situation est différente pour les attaquants et défenseurs :

- Un attaquant $l \in \{1, \ldots, L\}$ a une connaissance limitée de λ. Pour être plus précis, l'attaquant l connaît seulement λ_{J_l}. Par conséquent, nous pouvons simplement dire que $\theta_l = \tau_l(\lambda) = \lambda_{J_l}$ et $\Theta_l = \Lambda_{J_l}$. Le *type* d'un attaquant correspond à la portion du vecteur ressource qu'il observe.
- Un défenseur k avec $k \in \{L+1, \ldots, L+K\}$ a une connaissance complète du vecteur ressource. Donc $\theta_k = \tau_k(\lambda) = \lambda$ et $\Theta_k = \Lambda$. Le type d'un défenseur correspond donc au vecteur ressource complet réel.

Tout type joint *faisable* $\theta = \tau(\lambda) = (\theta_1, \ldots, \theta_L, \theta_{L+1}, \ldots, \theta_{L+K})$ est donc défini de manière unique par le vecteur ressource λ :

$$\begin{aligned} \theta_l &= \lambda_{J_l}, \forall l = 1, \ldots, L \\ \theta_k &= \lambda, \forall k = L+1, \ldots, L+K. \end{aligned}$$

En d'autre termes :

$$\begin{aligned} \Theta_l &= \Lambda_{J_l}, \forall l = 1, \ldots, L \\ \Theta_k &= \Lambda, \forall k = L+1, \ldots, L+K. \end{aligned}$$

L'application $\tau : \Lambda \to \Theta$ est injective, c'est-à-dire que $\lambda \neq \lambda' \Rightarrow \tau(\lambda) \neq \tau(\lambda')$. En d'autres termes, les types $\theta \in \Theta$ ne sont pas tous possibles (seuls ceux qui sont *compatibles* entre eux le sont), mais un unique type joint $\theta = \tau(\lambda)$ correspond à un seul vecteur ressource $\lambda \in \Lambda$,.

Exemple 41. *Prenons un exemple où* $L = 2, K = 2, M = 3$. *Supposons que* $\hat{\lambda} = (2, 2, 2)$ *(un vecteur ressource de 3 sites avec une valeur max possible à 2),* $\lambda = (1, 2, 1)$ *(le vecteur ressource réel), avec les observations des attaquants sur les sites suivant* $J_1 = \{2, 3\}$ *et* $J_2 = \{1, 3\}$. *Alors* $\Lambda_{J_1} = \Lambda_{J_2} = \{(0, 0), (0, 1), (0, 2), \ldots, (2, 1), (2, 2)\}$. *Nous avons par exemple :*

$$\theta = \tau(\lambda) = (\lambda_{J_1}, \lambda_{J_2}, \lambda, \lambda) = ((2, 1), (1, 1), (1, 2, 1), (1, 2, 1)) .$$

Probabilité jointe a priori dans un jeu de conservation bayésien

Considérons une distribution de probabilité jointe a priori sur les vecteurs ressources $\hat{P}$. On définit la distribution P sur Θ de la manière suivante :

Propriété 2 (Distribution de probabilité sur les types joints)**.** *La distribution* P *sur* Θ *issue de la distribution* $\hat{P}$ *sur* Λ *est définie par*[2] *:*

$$\forall \theta \in \Theta, P(\theta) = \sum_{\lambda \in \Lambda, \tau(\lambda) = \theta} \hat{P}(\lambda) \tag{6.7}$$

Preuve (Propriété 2)**.** *Puisque* τ *est injective :*

- *Si* $\exists \lambda \in \Lambda, \theta = \tau(\lambda)$, *alors* $P(\theta) = \hat{P}(\lambda)$,
- *Sinon* $P(\theta) = 0$.

P, *définie en* 6.7, *est bien une distribution de probabilité sur* Θ *(puisque* $\hat{P}$ *est une distribution de probabilité jointe a priori). Par ailleurs, pour tout défenseur* k, $\theta_k = \lambda$ *et tout attaquant* l, $\theta_l = \lambda_{J_l}$. *Donc* $P(\theta_l) = \hat{P}(\lambda_{J_l}) = \sum_{\lambda', \tau_l(\lambda') = \lambda_{J_l}} \hat{P}(\lambda')$.

De plus, il est possible de calculer les distributions de probabilité conditionnelles qui suivent pour tout $\theta = \tau(\lambda)$ *:*

- *Pour tout attaquant* l, $P(\theta_{-l}|\theta_l) = \hat{P}(\lambda|\lambda_{J_l}) = \hat{P}(\lambda_{\overline{J_l}}|\lambda_{J_l})$. *Ici* $_{-l}$ *désigne l'ensemble des joueurs différents de* l *(attaquants comme défenseurs).*
- *Pour tout défenseur* k, $P(\theta_{-k}|\theta_k) = P(\theta_{-k}|\lambda) = 1$ *si* $\theta_n = \tau_n(\lambda), \forall n = 1..N$ *sinon* 0. *Puisque le type du joueur* k *est directement* λ, *une connaissance complète des types des autres joueurs est assurée.*

Utilités dans un jeu de conservation bayésien

Définissons finalement les utilités u dans un jeu de conservation bayésien.

Définition 49. *Pour tout* $\lambda \in \Lambda$, $\omega \in \Omega, l \in \{1, ...L\}$ *et* $\theta = \tau(\lambda)$, $\hat{u}_l(\omega, \lambda_{\omega_l})$ *est définie comme dans l'équation 6.1, pour un jeu à information complète avec* λ_{ω_l} *connue.*

Attaquants :

$$u_l(\omega, \theta) = \hat{u}_l(\omega, \lambda_{\omega_l}) = \lambda_{\omega_l} - \sum_{k=L+1}^{L+K} \delta(\omega_l, \omega_k), \forall l = 1..L. \tag{6.8}$$

où λ_{ω_l} *est la valeur de ressource pour le site* ω_l *pour le type joint* $\theta = \tau(\lambda)$.
Pour les défenseurs, $\hat{u}_k(\omega)$ *est définie par l'équation 6.2*

Défenseurs :

$$u_k(\omega, \theta) = \hat{u}_k(\omega) = \sum_{l=1}^{L} \delta(\omega_l, \omega_k), \forall k = L + 1..L + K. \tag{6.9}$$

2. Puisque τ est injective, la somme ne comprend qu'un seul terme non nul

Les équations définissant $u_l(\omega, \theta)$ et $u_k(\omega, \theta)$ sont équivalentes aux équations 6.1 et 6.2. Ce choix d'utilisation des mêmes notations avec un nombre d'arguments différent a été fait afin de ne pas complexifier les notations. Étant donné que $u_k(\omega, \theta) = u_k(\omega)$ et que $u_l(\omega)$ et $u_l(\omega, \theta)$ sont différenciés par l'utilisation de θ, il n'y a pas de risque de confusion.

Dans un jeu bayésien, une stratégie mixte conditionnelle $\sigma(\cdot|\theta)$ est recherchée. Elle associe une distribution sur les stratégies jointes pures ω à tout type joint θ. L'utilité espérée des joueurs pour une stratégie jointe mixte (pour tout type joint donné) est calculée de la manière suivante (voir chapitre 3) :

$$EU_n(\sigma|\theta_n) = \sum_{\theta_{-n}} P(\theta_{-n}|\theta_n) \sum_{\omega\in\Omega} u_n(\omega, \theta) \left(\prod_{\nu=1}^{N} \sigma_\nu(\omega_\nu|\theta_\nu) \right), \forall n = 1, \ldots, N.$$

Pour l'expression de $P(\theta_{-n}|\theta_n)$ donnée ci-dessus, dans le contexte d'un ICG l'utilité espérée devient, pour un vecteur de ressource λ fixé (partiellement caché aux attaquants) :

— Pour tout attaquant $l = 1, \ldots, L$ ayant observé (de type) $\theta_l = \lambda_{J_l}$:

$$\begin{aligned} \widehat{EU}_l(\sigma|\lambda_{J_l}) = & \sum_{\lambda_{\overline{J_l}}\in\Lambda_{\overline{J_l}}} \hat{P}\left(\lambda_{\overline{J_l}}|\lambda_{J_l}\right) \sum_{\omega\in\Omega} \hat{u}_l(\omega, \lambda_{\omega_l}) \\ & \times \prod_{k=1}^{L} \sigma_l(\omega_l|\lambda_{J_l}) \prod_{k=L+1}^{K} \sigma_k(\omega_k|\lambda), \end{aligned} \tag{6.10}$$

— De manière similaire (mais plus simple), pour tout défenseurs $k = L+1, \ldots, L+K$, qui a connaissance du vecteur λ :

$$\widehat{EU}_k(\sigma|\lambda) = \sum_{\omega\in\Omega} \hat{u}_k(\omega) \prod_{k'=L+1}^{K} \sigma_{k'}(\omega_{k'}|\lambda) \prod_{l=1}^{L} \sigma_l(\omega_l|\lambda_{J_l}) \tag{6.11}$$

Exemple 42. *Prenons un exemple où $L = 2, K = 2, |\lambda| = 2$ avec $\lambda_{max} = 2$ et où $J_1 = \{1\}$ et $J_2 = \{2\}$. Ceci signifie que nous avons 3 types par attaquants (le site connu a une valeur à 0, 1 ou 2), 9 types par défenseur (car il y a 9 vecteurs λ possibles). Il y a donc 729 types joints possibles (avec seulement 9 d'entre eux de probabilité non nulle). L'ensemble des types θ de probabilité jointe $P(\theta)$ non nulle sera, pour tout $\theta = (\theta_1, \theta_2, \theta_3, \theta_4) = (\lambda_{J_1}, \lambda_{J_2}, \lambda, \lambda)$:*

λ_1	λ_2	$\hat{P}(\lambda_1, \lambda_2)$
0	0	$\frac{2}{50}$
0	1	$\frac{2}{50}$
0	2	$\frac{9}{50}$
1	0	$\frac{5}{50}$
1	1	$\frac{7}{50}$
1	2	$\frac{6}{50}$
2	0	$\frac{5}{50}$
2	1	$\frac{9}{50}$
2	2	$\frac{5}{50}$

$\theta_1 = \lambda_{J_1}$	$\theta_2 = \lambda_{J_2}$	$\theta_3 = \lambda$	$\theta_4 = \lambda$	$P(\theta)$
(0)	(0)	(0,0)	(0,0)	$\frac{2}{50}$
(0)	(1)	(0,1)	(0,1)	$\frac{2}{50}$
(0)	(2)	(0,2)	(0,2)	$\frac{9}{50}$
(1)	(0)	(1,0)	(1,0)	$\frac{5}{50}$
(1)	(1)	(1,1)	(1,1)	$\frac{7}{50}$
(1)	(2)	(1,2)	(1,2)	$\frac{6}{50}$
(2)	(0)	(2,0)	(2,0)	$\frac{5}{50}$
(2)	(1)	(2,1)	(2,1)	$\frac{9}{50}$
(2)	(2)	(2,2)	(2,2)	$\frac{5}{50}$

FIGURE 6.7 – Distribution de probabilité $\hat{P}$ sur l'ensemble Λ (à gauche) et la distribution de probabilité P sur les types joints possibles (à droite)

Pour le type joint $\theta = ((2), (1), (2, 1), (2, 1))$, les utilités $u_l(\omega, \theta)$ sont données dans la table de la figure 6.8.

ω_1	ω_2	ω_3	ω_4	u_1	u_2	u_3	u_4
1	1	1	1	0	0	2	2
1	1	1	2	1	1	2	0
1	1	2	1	1	1	0	2
1	1	2	2	2	2	0	0
1	2	1	1	0	1	1	1
1	2	1	2	1	0	1	1
1	2	2	1	1	0	1	1
1	2	2	2	2	−1	1	1
2	1	1	1	1	1	1	1
2	1	1	2	0	1	1	1
2	1	2	1	0	1	1	1
2	1	2	2	−1	2	1	1
2	2	1	1	1	1	0	0
2	2	1	2	0	0	0	2
2	2	2	1	0	0	2	0
2	2	2	2	−1	−1	2	2

FIGURE 6.8 – Table d'un jeu de conservation bayésien où $L = 2$,$K = 2$, $|\hat{\lambda}| = 2$ pour $\theta = ((2), (1), (2, 1), (2, 1))$.

6.4.3 Représentation par un jeu bayésien polymatriciel

De manière similaire à la version à information complète, il est possible d'utiliser une forme polymatricielle pour représenter un jeu bayésien de conservation.

L'ensemble des jeux locaux E est défini comme dans la version polymatricielle du jeu de conservation à information complète $E = ((l,k))_{\substack{l=1..L,\\ k=L+1..L+K}}$. Un jeu bayésien bimatriciel sera défini pour toutes les combinaisons attaquant/défenseur possibles.

Types joint locaux dans un jeu de conservation bayésien bimatriciel.

Pour chaque jeu local $e = (l,k), e \in E$, l'ensemble des types locaux est $\Theta_{(l,k)} = \Theta_l \times \Theta_k$. Les éléments de $\Theta_{(l,k)}$ sont de la forme $\theta_{(l,k)} = (\theta_l, \theta_k) = (\lambda_{J_l}, \lambda)$ avec $\lambda_{J_l} \in \Lambda_{J_l}, \lambda \in \Lambda$

L'interprétation du lien entre un type θ et un vecteur λ est la même que sous la forme bayésienne normale, pour les attaquants (connaissance partielle du vecteur) comme pour les défenseurs (connaissance complète du vecteur). De manière similaire à la forme bayésienne normale, nous pouvons considérer $\tau_{(l,k)} : \Lambda \mapsto \Theta_{(l,k)}$ associant à tout type joint local $\theta_{(l,k)} = \tau_{(l,k)}(\lambda) = (\theta_l = \tau_l(\lambda), \theta_k = \lambda)$.

Probabilités dans un jeu de conservation bayésien polymatriciel

Il est possible de définir une distribution de probabilité locale $P_{(l,k)}$ a partir de $\hat{P}(\lambda)$. Pour l'attaquant l, $P_{(l,k)}(\theta_k|\theta_l) = \hat{P}(\lambda|\lambda_{J_l}) = \hat{P}(\lambda_{\overline{J_l}}|\lambda_{J_l})$. Pour tout défenseur k, $P_{(l,k)}(\theta_l|\theta_k) = P(\theta_l|\lambda) = \hat{P}(\lambda_{J_l}|\lambda) = 1$ si $\theta_l = \tau_l(\lambda)$ et 0 sinon. Notez que pour deux défenseurs différents k,k' et un attaquant l, $P(\theta_k|\theta_l) = P(\theta_{k'}|\theta_l)$. A partir de la distribution jointe P sur Θ, nous pouvons calculer :

$$P_{(l,k)}(\theta_{(l,k)}) = \sum_{\theta_{-(l,k)}} P(\theta_{(l,k)}.\theta_{-(l,k)}), \forall (l,k) \in E, \theta_{(l,k)} \in \Theta_{(l,k)}.$$

où $\theta_{(l,k)} = (\theta_l, \theta_k)$, $\Theta_{(l,k)} = \Theta_l \times \Theta_k$.

Puisque $\theta_{(l,k)} = (\theta_l, \theta_k) = (\lambda_{J_l}, \lambda)$ alors $P_{(l,k)}(\theta_{(l,k)}) = \hat{P}(\lambda)$ où $\theta = \tau(\lambda)$ si $\theta_l = \lambda_{J_l}$ (et 0 sinon). D'après la propriété 2, $\sum_{\theta_{-(l,k)}} P(\theta_{(l,k)}, \theta_{-(l,k)}) = \hat{P}(\lambda)$ si $\theta = \tau(\lambda)$.

Utilité dans un jeu de conservation bayésien polymatriciel

L'utilité dans un jeu bayésien bimatriciel local peut être définie comme dans un jeu de conservation polymatriciel à information complète, à savoir :

Définition 50 (Utilité bayésienne polymatricielle)**.**

$$u_l^{(l,k)}(\omega_l, \omega_k, \theta_l, \theta_k) = \hat{u}_l^{(l,k)}(\omega_l, \omega_k, \lambda_{\omega_l}) = \frac{\lambda_{\omega_l}}{K} - \delta(\omega_l, \omega_k) \text{et} \quad (6.12)$$

$$u_k^{(l,k)}(\omega_l, \omega_k, \theta_l, \theta_k) = \hat{u}_k^{(l,k)}(\omega_l, \omega_k) = \delta(\omega_l, \omega_k). \quad (6.13)$$

où $(l,k) \in \{1..L\} \times \{L+1..L+K\}$

Proposition 27 (Représentation polymatricielle)**.** *Soit σ une stratégie jointe bayésienne, les utilités espérées de σ dans un ICG et dans le BG correspondant sont identiques. En d'autres termes :*

$$\begin{aligned} EU_l(\sigma|\theta_l = \lambda_{J_l}) &= \sum_{k=L+1}^{L+K} \widehat{EU}_l^{(l,k)}(\sigma_l, \sigma_k|\lambda_{J_l}) \\ EU_k(\sigma|\theta_k = \lambda) &= \sum_{l=1}^{L} \widehat{EU}_k^{(l,k)}(\sigma_l, \sigma_k|\lambda) \end{aligned}$$

Avec les utilités espérées locales définies par :

$$\begin{aligned} \widehat{EU}_l^{(l,k)}(\sigma_l, \sigma_k|\lambda_{J_l}) &= \sum_{\lambda_{\overline{J_l}} \in \Lambda_{\overline{J_l}}} \hat{P}_l\left(\lambda_{\overline{J_l}}|\lambda_{J_l}\right) \sum_{\omega_{l,k} \in \Omega_{l,k}} \hat{u}_l^{(l,k)}(\omega_l, \omega_k, \lambda_{\omega_l}) \sigma_l(\omega_l|\lambda_{J_l}) \sigma_k(\omega_k|\lambda) \\ \widehat{EU}_k^{(l,k)}(\sigma_l, \sigma_k|\lambda) &= \sum_{\omega_{l,k} \in \Omega_{l,k}} \hat{u}_k^{(l,k)}(\omega_l, \omega_k) \sigma_l(\omega_l|\lambda_{J_l}) \sigma_k(\omega_k|\lambda) \end{aligned}$$

Proposition 28. *Pour un jeu de conservation à information incomplète $ICG(\hat{\lambda}, L, K, \mathcal{J}, \hat{P})$, il existe un jeu bayésien polymatriciel $BPMG = \langle S, \Omega, \Theta, E = \{1, ..., L\} \times \{L+1, ..., L+K\}, \{P_{(l,k)}\}, \{u_l^{(l,k)}, u_k^{(l,k)}\}\rangle$. équivalent. En particulier, $\forall \omega \in \omega, \forall \theta \in \Theta$, $u_l(\omega, \theta) = \sum_{k=L+1}^{L+K} u_l^{(l,k)}(\omega_l, \omega_k, \theta_l, \theta_k)$ et $u_k(\omega, \theta) = \sum_{l=1}^{L} u_k^{(l,k)}(\omega_l, \omega_k, \theta_l, \theta_k)$*

Preuve (Proposition 28)**.** *Dans un jeu bayésien polymatriciel, nous avons :*

$$u_l(\omega, \theta) = \sum_{(l,k) \in E} u_l^{(l,k)}(\omega_{(l,k)}, \theta_{(l,k)}), \forall \omega \in \Omega$$

Qui, en reprenant l'équation 6.12, est équivalent à :

$$\begin{aligned} u_l(\omega, \theta) &= \sum_{(l,k) \in E} \frac{\lambda_{\omega_l}}{K} - \delta(\omega_l, \omega_k) \\ u_l(\omega, \theta) &= \lambda_{\omega_l} - \sum_{k=L+1}^{L+K} \delta(\omega_l, \omega_k) \end{aligned}$$

Ce qui correspond à l'utilité d'un attaquant dans un jeu de conservation bayésien.

La même chose peut être démontrée pour les défenseurs :

$$u_k(\omega, \theta) = \sum_{(l,k) \in E} u_k^{(l,k)}(\omega_{(l,k)}, \theta_{(l,k)}), \forall \omega \in \Omega$$

Qui, en reprenant l'équation 6.13, est équivalent à :

$$\begin{aligned} u_k(\omega, \theta) &= \sum_{(l,k) \in E} \delta(\omega_l, \omega_k) \\ u_k(\omega, \theta) &= \sum_{l=1}^{L} \delta(\omega_l, \omega_k) \end{aligned}$$

Concernant les probabilités, nous avons $P_{(l,k)}(\theta_{(l,k)}) = P(\theta), \forall (l,k) \in E$. En reprenant les notations précédentes nous avons :

$$P_{(l,k)}(\theta_{(l,k)}) = P_{(l,k)}(\theta_l, \theta_k) = P(\tau(\lambda)) = \hat{P}(\lambda)$$

Donc $P_e(\theta_e) = \hat{P}(\lambda)$ si et seulement si $\theta_e = \tau(\lambda)$, 0 sinon. Puisque comme vue avec la propriété 2, $P(\theta) = \hat{P}(\lambda)$ alors nous avons bien $P_e(\theta_e) = P(\theta), \forall e \in E$

Exemple 43. *Reprenons l'exemple du jeu bayésien précédent, où* $L = K = 2, |\lambda| = 2$, $\lambda_{max} = 2$ *et où les attaquants observent les sites suivants :* $J_1 = \{1\}$ *et* $J_2 = \{2\}$. *Un attaquant à trois types et un défenseur en possède neuf. Le jeu bayésien polymatriciel possède quatre joueurs et quatre jeux locaux. Dans chaque jeu bayésien local, il y a 27 types joints possibles (dont seuls 9 ont une probabilité non nulle). On peut représenter ce jeu par le graphe de la figure 6.9*

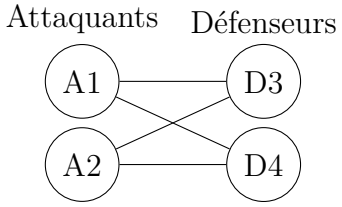


FIGURE 6.9 – Graphe d'un jeu polymatriciel bayésien lorsque $L = 2$ et $K = 2$.

En reprenant la distribution de probabilité précédente, l'ensemble des types $\theta_{(l,k)}$ *de probabilité jointe* $P(\theta_{(l,k)})$ *non nulle seront (avec* $\theta_{(l,k)} = (\theta_l, \theta_k) = (\lambda_{J_l}, \lambda)$*) :*

λ_1	λ_2	$\hat{P}(\lambda_1, \lambda_2)$
0	0	$\frac{2}{50}$
0	1	$\frac{2}{50}$
0	2	$\frac{9}{50}$
1	0	$\frac{5}{50}$
1	1	$\frac{7}{50}$
1	2	$\frac{6}{50}$
2	0	$\frac{5}{50}$
2	1	$\frac{9}{50}$
2	2	$\frac{5}{50}$

$\theta_1 = \lambda_{J_1}$	$\theta_k = \lambda$	$P(\theta_{(1,k)})$
(0)	(0,0)	$\frac{2}{50}$
(0)	(0,1)	$\frac{2}{50}$
(0)	(0,2)	$\frac{9}{50}$
(1)	(1,0)	$\frac{5}{50}$
(1)	(1,1)	$\frac{7}{50}$
(1)	(1,2)	$\frac{6}{50}$
(2)	(2,0)	$\frac{5}{50}$
(2)	(2,1)	$\frac{9}{50}$
(2)	(2,2)	$\frac{5}{50}$

$\theta_2 = \lambda_{J_2}$	$\theta_k = \lambda$	$P(\theta_{(2,k)})$
(0)	(0,0)	$\frac{2}{50}$
(0)	(1,0)	$\frac{5}{50}$
(0)	(2,0)	$\frac{5}{50}$
(1)	(0,1)	$\frac{2}{50}$
(1)	(1,1)	$\frac{7}{50}$
(1)	(2,1)	$\frac{9}{50}$
(2)	(0,2)	$\frac{9}{50}$
(2)	(1,2)	$\frac{6}{50}$
(2)	(2,2)	$\frac{5}{50}$

FIGURE 6.10 – Distribution de probabilité sur les Λ (à gauche), distribution de probabilité sur les types joints de l'attaquant 1 et d'un défenseur (centre) et de l'attaquant 2 et d'un défenseur (droite).

Notez que pour un attaquant, la distribution de probabilité locale sera la même quelle que soit le défenseur avec lequel il joue. Donc, bien qu'il y ait 4 jeux bayésiens locaux, seul deux distributions de probabilité différentes sur les types joints locaux doivent être stockées (une pour chaque attaquant).

Par exemple, pour les types joints locaux $\theta_{(1,k)} = (2, (2, 1))$ *et* $\theta_{(2,k)} = (1, (2, 1))$*, les utilités des jeux locaux entre un attaquant et un défenseur sont :*

ω_1	ω_k	u_1	u_k
1	1	0	1
1	2	1	0
2	1	$\frac{1}{2}$	0
2	2	$-\frac{1}{2}$	1

ω_2	ω_k	u_2	u_k
1	1	0	1
1	2	1	0
2	1	$\frac{1}{2}$	0
2	2	$-\frac{1}{2}$	1

FIGURE 6.11 – Table d'un jeu local pour $\theta_{(1,k)} = ((2), (2, 1))$

Enfin, bien qu'il y ait 4 *jeux locaux portant chacun sur* 9 *types joints locaux , les matrices d'utilités seront identiques quelle que soit l'identité du défenseur. Donc il y aura que* 2 *tables différentes à stocker (chacun portant sur* 9 *types joints locaux).*

Pour l'exemple précédent, la représentation sous forme normale des utilités demanderait 574 nombres rationnels (9 types joints et 4×16 utilités pour chacun de ces types joints). Sous la forme polymatriciel bayésienne, seuls 288 nombres rationnels sont nécessaire (4 jeux locaux, 9 types joints locaux et 2×4 utilités), 144 si nous utilisons le fait que plusieurs de ces tables sont identiques et qu'il n'est nécessaire d'en représenter qu'une par attaquant (au lieu d'une pour chaque défenseur avec lequel un attaquant joue).

6.5 Conclusion

Dans ce chapitre, nous avons proposé une formulation sous forme normale d'un problème de conservation de la biodiversité dans une situation de lutte contre le braconnage. Nous avons montré que ce modèle pouvait aussi être représenté sous une forme polymatricielle. Enfin, nous avons proposé une version à information incomplète de ces jeux de conservation. Cette formulation peut aussi tirer bénéfice d'une représentation succincte en utilisant la forme bayésienne polymatricielle.

Les travaux présentés dans ce chapitre peuvent être étendus de plusieurs manières. Il est possible de considérer des interactions plus complexes entre braconniers et gardes forestiers. Il est envisageable de tenir compte du "nombre" de braconniers et de gardes forestiers présents sur chaque site. On peut par exemple s'intéresser à des "interactions" par triplets $\delta(\omega_{l1}, \omega_{l2}, \omega_k)$ et $\delta(\omega_l, \omega_{k1}, \omega_{k2})$ (plutôt que par paires $\delta(\omega_l, \omega_k)$). En supposant que ces interactions sont additives, il est possible de modéliser le problème par un jeu hypergraphique avec des hyperarêtes de taille 3. Il est également possible d'envisager des jeux de conservation à information incomplète avec une incertitude sur le nombre d'attaquants. En effet, dans des cas réalistes, les gardes forestiers ne connaissent pas le nombre de braconniers dans la région. Ce type de jeu peut être modélisé comme un jeu à information incomplète particulier où le type d'un joueur peut être $\theta_n =$ "*absent*". [Ashlagi *et al.*, 2006], par exemple, présentent un jeu de sélection de ressource avec un nombre inconnue de joueurs, comparable à cet exemple. Des fonctions d'utilités plus réalistes, tenant compte de facteurs économiques(vente d'animaux,amendes), peuvent être facilement construites en utilisant l'expertise de spécialistes de la conservation.

Il est aussi possible d'explorer les liens entre la forme polymatricielle des jeux de conservation et les jeux anonymes. En effet, un braconnier ne joue qu'avec les gardes forestiers, et seules les actions jouées par les garde forestiers importent (indépendamment

de leur identité). L'inverse est vrai pour les gardes forestiers, jouant avec les braconniers.

Enfin, nous pouvons aussi nous intéresser à un modèle avec une évolution des ressources sur le temps avec l'aide des jeux stochastique [Filar et Vrieze, 1997]. En fin de compte, l'étude des interactions entre agents dans le domaine de la conservation de la biodiversité est un vaste champ d'étude qui est a peine effleuré ici.

Dans le chapitre suivant, l'implémentation python des différentes notions présentées dans les chapitres précédents sera décrite. Ensuite, nous décrirons les expérimentations menées sur les différentes contributions de la thèse, y compris les jeux de conservation.

Chapitre 7

Implémentation et expérimentations

7.1 Introduction

Comme vu dans les chapitres précèdents, le travail de ma thèse s'est concentré autour des représentations succinctes, bayésiennes et de la création d'un algorithme original de calcul d'equilibre de Nash mixte. Compte tenu du manque d'outils permettant de représenter et stocker les jeux sous une forme autre que les formes normales ou extensives, il s'est montré nécessaire de développer nos propres représentations. De plus compte tenu des outils nécessaires à la conception de notre algorithme, nous avons besoin d'outils spécifiques permettant de représenter des systèmes d'équations polynomiales et de les résoudre.

Nous avons décidé de développer une "boite à outils" permettant de modéliser différents types de jeux et de les résoudre (calculer un équilibre de Nash). Cette boite à outils a pris la forme d'un paquet python que nous avons appelé "gtnash". Celle ci fait appel à la bibliotheque Sagemath pour résoudre les systémes d'equations polynomiales.

Dans la section 7.2, je vais présenter de manière générale le contenu du paquet gtnash, en expliquant les raison des diffèrent choix d'implémentation qui ont été faits.

Dans la section 7.3, je présenterai les différentes expérimentations réalisées et leurs résultats.

7.2 Paquet gtnash

Pour pouvoir tester les différentes contributions présentées précédemment il est nécessaire de les implémenter. Notamment la version combinatoire et algébrique de l'algorithme de Wilson, présentée dans le chapitre 5 qui demande des outils particuliers ainsi que la formulation d'un jeu sous une forme bayésienne hypergraphique présentée dans le chapitre 4, qui demande de pouvoir stocker des jeux succincts.

Le langage choisit pour l'implémentation est Python (Python3 pour être exact). Ce langage a été choisi pour plusieurs raisons. L'une des principales raisons est liée au choix de l'utilisation de la bibliothèque SageMath. Sagemath [1] est un logiciel libre qui implémente une multitude de fonctionnalités dans diffèrents domaines mathématiques dont l'algèbre, la combinatoire ou la statistique...

1. `https://www.sagemath.org/index.html`

La principale utilisation de Sagemath sera pour écrire des systèmes d'équations polynomiales et pour les résoudre "exactement".

De plus, étant donné que Sagemath est utilisable via Python (Sagemath est écrit en python et cython), un langage avec lequel je suis familier, ceci a permi une mise en place plus rapide de l'implémentation. L'un des avantages de python est qu'il permet de faire du "prototypage" rapidement et d'être plus "flexible". Ceci est adapté à ce que nous souhaitions faire, vérifier le bon fonctionnement de l'algorithme décrit dans le chapitre 5. De plus, les questions d'optimisation des performances ne sont pas aussi importante que la question de validation du bon fonctionnement (même si elles peuvent être liées par moment, j'en reparlerai plus tard).

Le paquet gtnash, la documentation, les instructions pour son utilisation ainsi que des exemples d'utilisation sont disponible sur le projet GitLab public qui se trouve à l'adresse suivante : `https://forgemia.inra.fr/game-theory-tools-group/gtnash`.

Dans la suite de cette section je vais présenter brièvement les différents types de jeux modélisés et les algorithmes implémentés.

7.2.1 Classes de jeux

Notre paquet permet actuellement de modéliser les jeux suivants :

— Jeux sous forme normale : **NFG**
— Jeux hypergraphiques : **HGG** (incluant les jeux polymatriciels et graphiques, respectivement : **PMG** et **GG**)
— Jeux bayésiens : **BG**
— Jeux bayésiens hypergraphiques : **BHGG**

Bien qu'il existe des classes pour les jeux polymatriciels et graphiques, elles n'ont, actuellement, aucune fonctionnalité "notable", les différenciant des jeux hypergraphiques. Les jeux de classe PMG (héritant de HGG), cependant, sont utilisés par la classe implémentant l'algorithme de Howson.

La figure 7.1 représente un diagramme UML des interactions entre les classes. Toutes ces classes peuvent être utilisées en python sans nécessiter l'installation de Sagemath.

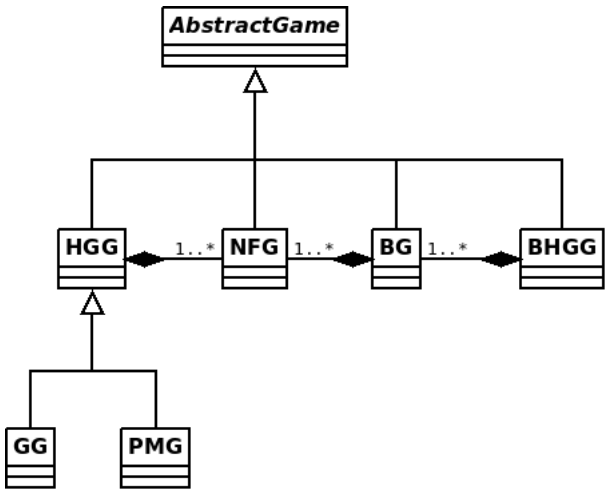


FIGURE 7.1 – Représentation de l'organisation des classes via un UML

Nous utilisons une classe abstraite **AbstractGame** qui sert à définir de manière abstraite un ensemble de fonctions qui seront utilisées dans tous les jeux, comme

notamment une fonction permettant de calculer l'utilité espéré d'une stratégie mixte, ou la lecture et écriture des jeux dans des fichiers.

Jeux sous forme normale

La première classe définie, et d'une certaine manière la plus importante car elle est indispensable au fonctionnement de toutes les autres classes est la classe **NFG**, celle des jeux sous forme normale.

Les objets de cette classe comprennent 3 données qui permettent de représenter un jeu "complet" : un nombre de joueurs, les stratégies des joueurs et leurs tables d'utilité.

Le nombre de joueurs et leurs stratégies sont données en entrée par une liste de sous-listes d'entiers. Chaque sous liste correspond aux stratégies du joueur dont l'indice est celui de la sous liste correspondante. Par exemple, prenons $[[0,1],[0,1,2],[0,1]]$, il y a 3 joueurs, le deuxième a trois actions tandis que le premier et le troisième en ont deux. Par convention, nous partons du principe que les stratégies d'un joueur sont représentées par des entiers qui commencent à 0. Nous nommons aussi les joueurs de manière similaire, le premier joueur sera le joueur 0, le second le joueur 1, etc... Ce choix a été effectué car il facilite la mise en place de plusieurs fonctionnalités en python. Les tables d'utilité sont représentées par une liste de sous-listes d'entiers. La première sous-liste correspond aux utilités du premier joueur, etc...

Les utilités doivent être organisées dans un ordre particulier, correspondant à un ordre sur les stratégies jointes possibles. Cet ordre correspond à celui des stratégies jointes énumérées dans figure 7.2. C'est-à-dire, en partant de la stratégie jointe où les joueurs jouent leurs première stratégie que nous "augmentons", les indices des stratégies des derniers joueurs en premier, comme si nous écrivions des nombres dans une base.

Par exemple si les N joueurs ont 2 actions (0 et 1) cela revient a écrire dans un ordre croissant tous les nombres binaires à N positions. Si nous avons un jeu avec trois joueurs et deux stratégies par joueur, avec les utilités présentées dans le tableau de la figure 7.2, la structure de données correspondante est une liste de trois sous-listes ordonnées de la manière suivante :
$[[6,7,0,5,1,4,2,3],[0,3,7,4,6,5,1,2],[4,7,4,0,3,2,6,1]]$

ω_0	ω_1	ω_2	u_0	u_1	u_2
0	0	0	6	0	4
0	0	1	7	3	7
0	1	0	0	7	4
0	1	1	5	4	0
1	0	0	1	6	3
1	0	1	4	5	2
1	1	0	2	1	6
1	1	1	3	2	1

FIGURE 7.2 – Représentation sous la forme normale du jeu

Le code permettant de créer une instance de ce jeu dans gtnash est :

```
NFG([[0,1], [0,1], [0,1]],
    [[6,7,0,5,1,4,2,3], [0,3,7,4,6,5,1,2], [4,7,4,0,3,2,6,1]]
    )
```

Jeux bayésiens

Les jeux bayésiens nécessitent la représentation d'une distribution de probabilité sur les types joints des joueurs. Les types des joueurs sont définis de manière similaire aux actions, par une liste de sous-listes d'entiers. La distribution de probabilité sur les types joints est définie par une liste d'entiers ordonnée (de manière similaire aux utilités avec les stratégies jointes). Lorsque le jeu est créé les entiers sont normalisés afin d'obtenir une distribution de probabilité. Ce choix permet de manipuler des probabilités rationnelles et de conserver des calculs exacts. Il permet également d'avoir des EN représentables par des nombres "rationnels" pour les jeux bayésiens polymatriciels.

Si un jeu a deux joueurs et deux type par joueurs, il y a 4 types joints possible. La distribution de probabilité sera décrite par une liste telle que $[5, 2, 1, 2]$ qui correspondra à la distribution représentée dans le tableau de la figure 7.3.

θ_0	θ_1	P	P_{norm}
0	0	5	$\frac{1}{2}$
0	1	2	$\frac{1}{5}$
1	0	1	$\frac{1}{10}$
1	1	2	$\frac{1}{5}$

FIGURE 7.3 – Représentation de la distribution de probabilité non normalisée et normalisée

Chacun des "jeux possibles" associés à chacun des types joints possibles sera une instance d'un jeu sous forme normale, comme définie précédemment.

```
BG(players_actions=[[0, 1, 2], [0, 1]],
    utilities=[
    [[-3, -7, -10, -1, -5, -9], [-3, -7, -1, -5, -10, -9]],
    [[-2, -6, -9, -1, -4, -8], [-2, -6, -1, -4, -9, -8]],
    [[-2, -5, -8, -1, -3, -7], [-2, -5, -1, -3, -8, -7]],
    [[-2, -4, -7, -1, -3, -6], [-2, -4, -1, -3, -7, -6]],
    [[-2, -4, -6, -1, -3, -5], [-2, -4, -1, -3, -6, -5]],
    [[-3, -7, -1, -5, -10, -9], [-3, -7, -10, -1, -5, -9]]],
    theta=[[0, 1], [0, 1, 2]],
    p=[2, 4, 4, 4, 2, 4])
```

Nous avons fait une méthode permettant d'obtenir un jeu hypergraphique en partant d'un jeu bayésien, comme démontré dans le chapitre 4

Jeux hypergraphiques

Un jeu hypergraphique, comme un jeu sous forme normale, sera composé d'une liste de sous-listes d'entiers afin de déterminer le nombre de joueurs et leurs actions. Une liste d'hyperarêtes, représentée par une liste de sous-listes d'indices de joueurs. Notez que l'ordre des joueurs dans une hyperarête est important. Le premier joueur membre d'une hyperarête est considéré comme le premier joueur du jeu local, le second joueur listé en sera le deuxième et ainsi de suite.

Par exemple, supposons que nous avons un jeu hypergraphique (polymatriciel) entre 3 joueurs avec un graphe complet. Les hyperarêtes peuvent être décrites par la

liste : $[[0,1],[0,2],[1,2]]$. Le joueur 0 est le premier joueur dans tous les jeux locaux auxquels il participe et le joueur 2 est toujours dernier.

Chacun des jeux locaux sera décrit par une instance d'un jeu sous forme normale dont l'ordre des joueurs est celui de l'hyperarête correspondante. Par conséquent au lieu de donner une liste de sous-listes d'entiers dans une instance de jeu hypergraphique, on donne une liste de sous-listes de sous-listes d'entiers.

Le code suivant crée une instance d'un jeu hypergraphique avec 3 jeux locaux, un à trois joueurs entre les joueurs 0, 1 et 2 et deux entre deux joueurs, un entre le joueur 1 et 3 et un entre le joueur 2 et 3.

```
HGG(players_actions=[[0, 1], [0, 1], [0, 1], [0, 1]],
    utilities=[
    [[-3, -7, -7, -10, -1, -5, -5, -9],
     [-3, -7, -1, -5, -7, -10, -5, -9],
     [-3, -1, -7, -5, -7, -5, -10, -9]],
    [[-1, -5, 0, -3], [-1, 0, -5, -3]],
    [[-1, -5, 0, -3], [-1, 0, -5, -3]]
    ],
    hypergraph=[[0, 1, 2], [1, 3], [2, 3]])
```

Nous avons aussi défini une méthode permettant d'obtenir le jeu hypergraphique (à information complète) équivalent au jeu bayésien hypergraphique, comme démontré dans le chapitre 4.

Jeux bayésiens hypergraphiques

Les jeux bayésiens hypergraphiques généralisent à la fois les jeux bayésiens (nécessitant de représenter une distribution de probabilité sur les types joints), et les jeux hypergraphiques (un jeu hypergraphique dont les jeux locaux sont bayésiens)

Le code qui suit correspond à la création d'une instance d'un jeu bayésien polymatriciel entre quatre joueurs avec quatre jeux locaux.

```
BHGG(players_actions=[[0, 1], [0, 1], [0, 1], [0, 1]],
    utilities=[
    [[[3, 0, 0, 2], [3, 0, 0, 2]], [[3, 0, 0, 2], [1, 0, 0, 2]],
    [[1, 0, 0, 2], [3, 0, 0, 2]], [[1, 0, 0, 2], [1, 0, 0, 2]]],
    ....,...,...
    ],
    hypergraph=[[0, 1], [1, 2], [2, 3], [0, 3]]
    theta=[[0, 1], [0, 1], [0, 1], [0, 1]],
    p=[[4, 1, 1, 4], [4, 1, 1, 4], [4, 1, 1, 4],[4, 1, 1, 4]])
```

7.2.2 Écriture des jeux dans un fichier

La boite à outils Gambit [McKelvey *et al.*, 2014] offre deux formats de fichiers (d'extension *.nfg* et *.efg*) pour stocker des jeux sous forme normale ou extensive.

Nous avons étendu ces formats pour stocker les types de jeux précédents, les extensions sont ".bg", ".hgg" et ".bhgg".

Format nfg (gambit)

Le format nfg (d'extension *.nfg*) propose deux versions d'écriture différente, la version "payoff" et la version outcome. Elles sont respectivement présentées dans les figures 7.4 et 7.5.

```
NFG 1 R "Selten (IJGT, 75), Figure 2, normal form" { "Player 1" "Player 2" } { 3 2
↪  }

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

FIGURE 7.4 – Jeux sous forme normale, version payoff

```
NFG 1 R "2x2x2 example with 3 pure, 2 incompletely mixed, and a continuum of
↪  completely mixed NE" { "" "" "" }

{ { "1" "2" }
{ "1" "2" }
{ "1" "2" }
}
""

{
{ "" 0, 0, 2 }
{ "" 3, 0, 0 }
{ "" 0, 3, 0 }
{ "" 0, 0, 0 }
{ "" 1, 1, 0 }
{ "" 0, 0, 0 }
{ "" 0, 0, 0 }
{ "" 0, 0, 3 }
}
1 2 3 4 5 6 7 8
```

FIGURE 7.5 – Jeux sous forme normale, version outcome

Nous avons basé nos formats sur la version "payoff" du format *.nfg*.

En reprenant le jeu décrit dans la figure 7.4, nous lisons la chose de la manière suivante. La première ligne sert à la description du jeu. Le début de la ligne "NFG 1 R" décrit le type du jeu ("NFG"), la version du fichier ("1") et le type des nombres ("R" pour rationnel). Le contenu entre guillemets est une description du jeu.

Le plus important sur cette ligne est le contenu après les guillemets. Les deux listes entre crochets "{ "Player 1" "Player 2" } et "{ 3 2 }" donnent respectivement les noms des joueurs et leurs nombres de stratégies. Ici nous avons un jeu avec 2 joueurs où le premier joueur a trois stratégies et le second en a deux.

Les utilités sont est représentées sur une ligne. Chaque nombre représente l'utilité d'un joueur pour une stratégie jointe particulière. Pour un jeu avec N joueurs, tous les N premier nombres correspondront à la première stratégie jointe avec le premier nombre correspondant à l'utilité du premier joueur, etc. Ici, les utilités "1 1 0 2 0 2 1 1 0 3 2 0" correspondent au découpage de la figure 7.6.

ω_1 ω_2	1 1	2 1	3 1	1 2	2 2	3 2
u_1 u_2	1 1	0 2	0 2	1 1	0 3	2 0

FIGURE 7.6 – Découpage des utilité

Si nous représentions les utilités sous une forme de tableau en gardant le même ordre d'énumération des stratégies jointes (différent de celui utilisé jusqu'à maintenant) nous aurions le tableau de la figure 7.7

ω_1	ω_2	u_1	u_2
1	1	1	1
2	1	0	2
3	1	0	2
1	2	1	1
2	2	0	3
3	2	2	0

FIGURE 7.7 – Utilité du jeu sous forme normale tel qu'écrit dans le fichier

Notez que l'ordre des utilités dans un fichier *.nfg* et dans un objet de la classe NFG n'est pas le même. Lorsque la méthode de lecture (et d'écriture) est utilisé, l'ordre des joueurs entre le jeu au format .nfg et l'objet NFG est inversé.

Format hgg

Nous avons étendu le format nfg de gambit pour proposer un format de fichier "concis" pour les jeux hypergraphiques

La première ligne est similaire à celle de la figure 7.8.

```
HGG 0 R "A Legend" { "Player 1" "Player 2" "Player 3" "Player 4"} { 2 2 2 2 }
```

FIGURE 7.8 – Premiere "ligne" de description d'un fichier ".hgg"

La principale différence avec le format .nfg intervient lorsque l'on définit les utilités.

Nous considérons que chaque jeu local est défini par deux lignes. La première sert a définir les joueurs impliqués dans le jeux local et la seconde définit leurs utilités (de la même manière que dans la version payoff d'un fichier ".nfg"). La figure 7.9 représente un jeu hypergraphique où il y à trois jeux locaux, un entre trois joueurs et deux entre deux joueurs.

```
{ 0 1 2 }
5 6 1 4 4 3 1 2 1 3 6 2 4 6 2 6 1 1 4 4 2 3 3 3

{ 1 3 }
2 3 1 1 1 2 2 2

{ 2 3 }
5 3 3 5 5 2 1 1
```

FIGURE 7.9 – Représentation des jeux locaux et de leurs utilités dans un fichier ".hgg"

Le jeu local entre les joueurs 1 et 3 est représenté dans la figure 7.10.

ω_1	ω_3	u_1	u_3
0	0	2	3
1	0	1	1
0	1	1	2
1	1	1	1

FIGURE 7.10 – Utilité du jeu entre le joueur 1 et le joueur 3

Format : .bg

Comme pour les jeux hypergraphiques, nous avons étendu le format *.nfg* afin de pouvoir stocker des jeux bayésiens.

A la différence des jeux à information complète, nous devons avoir un ensemble de types pour chacun des joueurs. Nous précisons les types des joueurs de manière similaire à leurs stratégies, par un ensemble d'entiers définis entre crochets, à la fin de la première ligne.

La figure 7.11 représente la premiere ligne d'un fichier *.bg* pour un jeu bayésien à deux joueurs qui ont deux stratégies et deux types.

```
BG 0 R "nothing" { "Player 0" "Player 1" } { 2 2 } { 2 2 }
```

FIGURE 7.11 – Premiere ligne d'un fichier ".bg"

Les utilités des joueurs sont représentées dans la figure 7.12. Ici les nombres entre crochets "{ 4 1 1 4 }" représentent la distribution de probabilité non normalisée sur les types joints, qui sont énuméré de la même manière que les stratégies jointes (et différemment de l'ordre utilisé dans gtnash). Chacune des lignes d'entiers représente les utilités des joueurs pour chacun des types joints possibles énumérés dans le même ordre que celui de la distribution de probabilité. Le jeu est représenté sous forme tabulaire dans la figure 7.13.

```
{ 4 1 1 4 }
3 3 0 0 0 0 2 2
1 3 0 0 0 0 2 2
3 1 0 0 0 0 2 2
1 1 0 0 0 0 2 2
```

FIGURE 7.12 – Représentation des utilités dans un fichier ".bg"

ω_1	ω_3	u_1	u_3
0	0	1	3
1	0	0	0
0	1	0	0
1	1	2	2

θ_0	θ_1	P_{unorm}	P
0	0	4	$\frac{2}{5}$
1	0	1	$\frac{1}{10}$
0	1	1	$\frac{1}{10}$
1	1	4	$\frac{2}{5}$

FIGURE 7.13 – Représentation des utilités pour le type joint $(1, 0)$ (à gauche) et de la distribution de probabilité sur les types (à droite)

Nouveau format : .bhgg

La représentation des jeux bayésiens hypergraphiques reprend la même logique d'écriture que pour les fichier ".hgg" et ".bg"

La première ligne fonctionne sur le même principe que celle d'un jeu bayésien : nous avons le nom des joueurs, suivi de leurs nombres de stratégies puis finalement de leurs nombres de types(figure 7.14).

```
BHGG 0 R "nothing" { "Player 0" "Player 1" "Player 2" } { 2 2 2 } { 2 2 2 }
```

FIGURE 7.14 – Première ligne d'un fichier ".bhgg"

Les utilités d'un jeu bayésien hypergraphique (polymatriciel ici) sont représentées dans la figure 7.15. Un jeu local bayésien est représenté par un groupe de lignes composé d'une première ligne listant les joueurs impliqués dans le jeu local, la seconde ligne représente la distribution de probabilité locale et le reste des lignes représente les utilités pour chaque types joint local. L'ordre d'énumération des stratégies et des types joints est le même que dans un fichier ".bg".

```
{ 0 1 }
{ 7 1 2 9 }
14 15 23 41 65 0 100 7
100 0 51 41 72 58 93 92
100 56 95 38 43 0 0 14
98 53 12 69 100 17 0 15

{ 1 2 }
{ 1 4 5 7 }
0 39 78 12 38 42 100 50
15 0 10 7 66 100 78 58
0 74 29 100 98 96 35 76
0 52 43 21 81 10 56 100
```

FIGURE 7.15 – Représentation des utilités dans un fichier ".bhgg"

ω_0	ω_1	u_0	u_1
0	0	14	15
1	0	23	41
0	1	65	0
1	1	100	7

ω_0	ω_1	u_0	u_1
0	0	100	0
1	0	51	41
0	1	72	58
1	1	93	92

ω_0	ω_1	u_0	u_1
0	0	100	56
1	0	95	38
0	1	43	0
1	1	0	14

ω_0	ω_1	u_0	u_1
0	0	98	53
1	0	12	69
0	1	100	17
1	1	0	15

FIGURE 7.16 – Jeu local entre le joueur 0 et 1 selon les types joints suivant (de gauche à droite), $(0,0)$, $(1,0)$, $(0,1)$ et $(1,1)$

θ_0	θ_1	P_{unorm}	P
0	0	7	$\frac{7}{19}$
1	0	1	$\frac{1}{19}$
0	1	2	$\frac{2}{19}$
1	1	9	$\frac{9}{19}$

FIGURE 7.17 – Distribution de probabilité non normalisée et normalisée pour chacun des types joint locaux possible entre le joueur 0 et 1

7.2.3 Algorithmes de calcul d'équilibres de Nash

Actuellement trois algorithmes sont implémenté dans gtnash :

— L'algorithme de Howson pour résoudre les jeux polymatriciels [Howson, 1972]
— La méthode d'énumération de supports (avec résolution de systèmes d'équations polynomiales) [Porter *et al.*, 2008]
— La méthode de parcours de chemin présentée dans le chapitre 5.

L'algorithme de Howson a été implementé sans utiliser Sagemath, la bibliothèque numpy étant "suffisante" pour implémenter cette méthode avec une efficacité "satisfaisante", nous utilisons un paquet spécifique pour manipuler des nombres rationnels. Donc même une installation Python3 "standard" peut l'utiliser (les jeux sont également modélisés sans utiliser Sagemath).

L'utilisation de Sagemath intervient principalement pour l'implémentation de deux choses, la résolution des systèmes d'équations polynomiales (et par extension dans les fonctions utilisant les PCP) et une implémentation de l'élimination répétée des stratégies dominées" (ERSD) pour les jeux hypergraphiques (utilisant la PLNE).

L'énumération de supports et notre méthode ont été implémentées, elles permettent de calculer un équilibre de Nash mixte "exact".

Comme mentionné précédemment, la bibliothèque Sagemath est utilisée pour représenter des objets de classe *PolynomialComplementaryProblem*, son utilisation est justifiée par plusieurs fonctionnalités.

— Tout d'abord, Sagemath nous permet de créer les variables x_i^n en définissant un anneau polynomial multivarié sur le champs des nombres algébriques. Un anneau polynomial multivarié est créé avec *ring=PolynomialRing(QQbar, var_xy, order='lex')*, sur des nombres algébrique (*QQbar*) avec l'ensemble des variables défini dans une liste de chaînes de caractère (*var_xy*) selon un ordre lexicogaphique (order='lex').
— Des polynômes sont définis en additionnant et multipliant les variables entre elles et avec des entiers (correspondant aux désutilités). Lorsque nous avons une variable dans une chaîne de caractère 'x_n_i' pour obtenir l'objet "variable" il faut utiliser l'anneau défini précédemment de la manière suivante : ring('x_n_i'). Lorsque plusieurs objets variables sont multipliés et additionnés avec des entiers, un polynôme est retourné. Les polynômes $A_i^n(x^{-n})-1$ sont définis comme décrit dans le chapitre 5.

Un PCP est définit par un ensemble de variables et de polynômes dans un anneau polynomial. Les deux algorithmes vont tous les deux chercher à résoudre des systèmes d'équations polynomiales définis à partir du PCP (des systèmes $\mathcal{S}^{\overline{W},W}$ ou $\mathcal{S}^{Z,W}$). Les

équations d'un système seront soit du type $x_i^n = 0$, soit $A_i^n(x^{-n}) - 1 = 0$. Un système est défini et résolu de la manière suivante :

- Selon une liste de variable et d'équations devant être nulles, nous définissons un idéal dans l'anneau défini. Un idéal est créé en utilisant *ide=ideal(my_equations)*. La fonction prend en entrée une liste d'équations définies sur un anneau particulier. Notez que la création est instantanée et semble n'effectuer aucun calcul "coûteux".
- Le calcul de la dimension d'un idéal se fait avec *ide.dimension()*. Si la dimension est 0, cela signifie que nous avons un noeud avec une potentielle solution.
- La solution d'un système est calculée via la variété algébrique de l'idéal en utilisant *ide.variety()*. Une liste de dictionnaires associant un nombre algébrique à chacune des variables est retournée. Notez que les solutions peuvent contenir une variable de valeur négative ou donner une valeur négative à l'un des polynômes. Il faut donc effectuer une vérification de chacune des solutions pour savoir si elle est valide.
- La solution d'un système peut être normalisée sous la forme d'une distribution de probabilité. Notez qu'il est possible de visualiser les valeurs d'un nombre algébrique sous une forme autre qu'un nombre à virgule avec la méthode *.radical_expression()*.

Il est important de noter que des calculs de bases de Gröbner sont effectués lorsque nous calculons la dimension d'un idéal et sa variété algébrique. Ce calcul peut potentiellement prendre du temps (il est exponentiel). Le calcul de base de Gröbner peut se faire via *ide.groebner_basis()* et retourne une liste d'équations sur l'anneau polynomial défini précédemment.

Nous nous sommes aussi intéressés à une implémentation de ERSD (appelé IRDA dans gtnash) pour les jeux hypergraphiques. C'est un problème qui devient plus difficile dans le cadre hypergraphique à cause de la représentation compacte des utilités. Nous modélisons ce problème sous la forme d'un problème de programmation linéaire en nombres entiers binaires avec *bilp=MixedIntegerLinearProgram(maximization=True, solver="GLPK")* et *b = bilp.new_variable(binary=True)*. Le solveur GLPK correspond au solveur "GNU Linear Programming Kit"

Des notebook jupyter sont accessible sur le projet gitlab de gtnash [2]. Ils illustrent quelques exemples d'utilisation des algorithmes.

7.3 Expérimentations

Les expérimentations ont été faites sur un projet Gitlab, différent de gtnash, qui n'est pas accessible en ligne. Ce projet a été utilisé afin de mettre en place la génération automatisée des différents jeux et l'exécution des algorithmes, en mesurant différents paramètres.

Le choix des jeux générés sera abordé lorsque nous parlerons de chacune des expérimentations effectuées. Les utilités des jeux ont été générées via le générateur

2. Dans le projet `https://forgemia.inra.fr/game-theory-tools-group/gtnash`, le dossier notebook contient plusieurs notebook jupyter illustrant l'utilisation de gtnash et des exemples de jeu

de jeux GAMUT [Nudelman *et al.*, 2004] (sauf pour les jeux de conservation). GAMUT est un générateur qui permet de créer différent types de jeux et notamment de retourner un fichier sous un format ".nfg". Nous avons utilisé les jeux "CovariantGame" et "RandomGame". Ce sont les seuls jeux de GAMUT qui nous permettent de créer des jeux "simplement" sans avoir une présence certaine d'un équilibre de Nash pur. La principale raison de ce choix est liée à la comparaison de PNS (énumération de supports [Porter *et al.*, 2008]) à notre implémentation de l'algorithme de Wilson, nous y reviendrons plus tard.

Les données des classes d'expérimentations sont stockées dans des fichiers ".pickle", représentables dans un fichier ".csv".

Chaque jeu a été résolu sur un noeud d'une plateforme HPC. Tous les noeuds utilisent un Intel Xeon E5-2680 v4 2,4 Ghz bi-processeurs avec 128 Gb de RAM, sous un système d'exploitation Linux[3].

7.3.1 Jeux bayésiens polymatriciels

Une première partie des expérimentations sur les jeux bayésiens hypergraphiques a été effectuée plus particulièrement sur des jeux bayésiens polymatriciels. Le but de ces expérimentations est de comparer l'efficacité des différentes méthodes de résolution pour trouver un équilibre de Nash et de montrer l'avantage de la représentation sous une forme bayésienne succincte. La comparaison s'est principalement faite avec deux approches différentes :

— Transformer le jeu bayésien polymatriciel en sa forme polymatricielle à information complète et utiliser l'algorithme de Howson [Howson, 1972] sur ce jeu (présenté dans la section 2.7.1 du chapitre 2).

— Transformer le jeu bayésien polymatriciel en sa forme normale à information complète et utiliser les algorithmes implémentés dans gambit.

Pour les expérimentations, nous aurons un nombre $N \in \{4, 5, ..., 10\}$ de joueurs, un nombre $t \in \{2, 3, ..., 10\}$ de types et un nombre $m \in \{2, 3, ..., 10\}$ d'actions.

En fonction de la limite du temps de calcul, certaines combinaisons ne seront pas nécessairement testées. D'autres ne le seront pas, tout simplement car il n'est pas possible de les générer actuellement (notamment à cause de la génération de la distribution de probabilité et de condition de cohérence globale qui nous demande quand même de générer les probabilités jointes globales).

Pour la génération de graphe, il est nécessaire d'avoir un nombre de jeux bayésiens bimatriciels à générer. Ce nombre $|E|$ de jeux locaux doit être au minimum $N-1$ et au plus $\frac{N!}{2*(N-2)!}$. Pour un nombre de joueurs donné, 5 nombres différents de jeux seront sélectionnés de manière à avoir des nombres espacés de manière équilibrée : $\{N-1, |E|_1, |E|_2, |E|_3, \frac{N!}{2*(N-2)!}\}$ pour chaque N avec $|E|_1$,$|E|_2$ et $|E|_3$ étant respectivement autour de 45%, 65% et 80% du nombre maximum de jeux locaux possibles.

Pour chaque configuration possible (nombre de joueurs, actions, types, type du jeu et nombre de jeux locaux) 40 jeux différents sont générés. Les valeurs des utilités dans les jeux locaux sont choisies aléatoirement uniformément entre 0 et 100 pour les RandomGame. Pour les CovariantGame, les utilités sont générées selon une loi jointe normale avec un coefficient de covariance r tiré uniformément entre $\frac{-1}{N-1}$ et 1 (N est le nombre de joueurs). Les utilités sont ensuite projetées sur les entiers entre 0 et 100.

3. Via le serveur MesolLR, plus d'information sur le lien suivant `https://meso-lr.umontpellier.fr/`

Pour l'algorithme de Howson, il y a plusieurs "points de départs" possibles en fonction du nombre de stratégies jointes pures. Certaines exécutions partant de stratégies jointes différentes peuvent amener vers un même équilibre et certains équilibres peuvent ne pas être atteints même en utilisant toutes les stratégies jointes initiales possibles. La stratégie initiale choisie sera celle où tous les joueurs jouent leur première stratégie.

Pour gambit, il existe plusieurs algorithmes permettant d'obtenir un équilibre de Nash mixte (approché) pour des jeux avec plus de deux joueurs.

Les quatre algorithmes suivants ont été utilisés :

— *gambit-enumpoly* : Énumération de support [Porter *et al.*, 2008]
— *gambit-gnm* : Global Newton Method [Govindan et Wilson, 2003]
— *gambit-ipa* : Iterated polymatrix approximation [Govindan et Wilson, 2004]
— *gambit-simpdiv* : Simplicial subdivision [van der Laan *et al.*, 1987]

Les algorithmes *gambit-enumpoly* et *gambit-gnm* peuvent retourner plusieurs équilibres de Nash, contrairement aux autres (*gambit-enumpoly* les retournera tous). Il n'y a, apparemment, pas d'option pour retourner seulement le premier rencontré. Pour chacun des algorithmes, l'écart entre la meilleure solution trouvée et une amélioration possible (en modifiant la stratégie mixte d'un joueur pour une stratégie pure) sera mesuré (s'il n'y en a pas l'erreur sera à 0)

Comparaison des résolutions via la forme polymatricielle et la forme normale

Pour la résolution sous forme polymatricielle, 73360 jeux différents ont été testés. Toutes les transformations en jeu polymatriciel ont été terminées avec succès. Un total de 51 313 exécutions de l'algorithme de Howson se sont terminées avant le timeout de 5 minutes. Notez que pour 81 jeux sur les 73360 jeux testés, l'équilibre retourné est considéré comme "Faux". On peut attribuer l'erreur à une approximation qui intervient lors des calculs.

Un nombre de configurations nettement moins élevé a été testé pour la résolution sous forme normale, car même si gambit fournit des algorithmes efficaces, le temps nécessaire pour obtenir un jeu sous forme normale à partir de la forme bayésienne polymatricielle devient vite contraignant.

Sur les 6066 jeux testés, 2153 transformations ont été terminées avec succès et 1105 transformations et exécutions de gambit ont été menées avec succès.

La figure 7.18 représente une comparaison des temps de transformation vers la forme polymatricielle, et vers la forme normale. On peut voir qu'il devient vite impossible de transformer un jeu vers sa forme normale en un temps raisonnable.

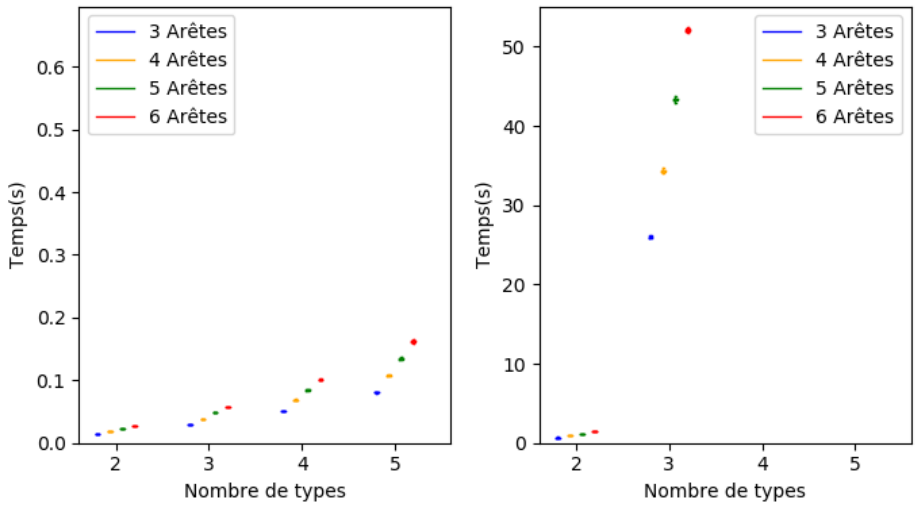


FIGURE 7.18 – Temps de transformation des jeux avec 4 joueurs et 2 actions vers la forme polymatricielle (gauche) et vers la forme normale (droite).

De plus, si nous regardons les meilleurs temps d'exécution de l'algorithme de Howson et de gambit dans la figure 7.19, on voit que la méthode de Howson commence à devenir moins performante bien après les algorithmes de gambit.

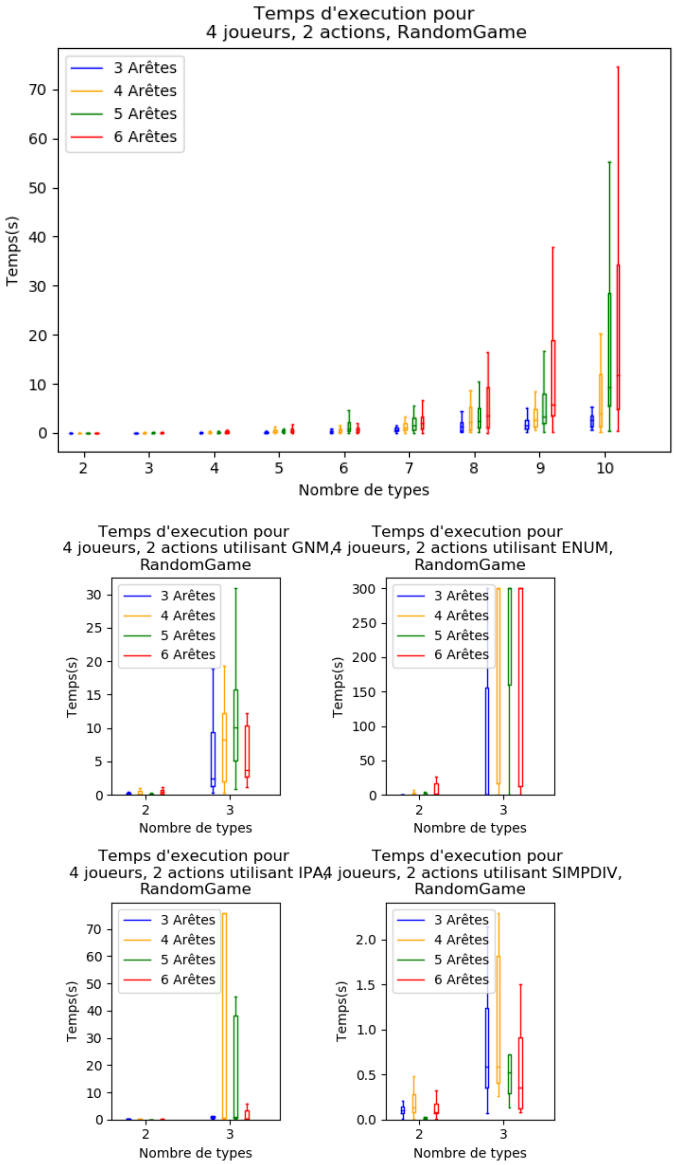


FIGURE 7.19 – Temps de résolution pour les jeux avec 4 joueurs et 2 actions pour l'algorithme de Howson et les différents algorithmes de gambit.

7.3.2 Jeux sous forme normale

Les expérimentations sur les jeux sous forme normale permettent de tester le bon fonctionnement et l'efficacité de notre implémentation de la version algébrique et combinatoire de l'algorithme de parcours de chemin de Wilson. La comparaison est effectuée avec une implémentation de l'algorithme d'énumération de supports [Porter *et al.*, 2008] utilisant des systèmes polynomiaux similaires à ceux utilisés dans l'algorithme de Wilson.

Les configurations utilisées font varier le nombre de joueurs, d'actions par joueur et le type du jeux (type gambit). Pour les expérimentations, nous avons un nombre $N \in \{3, 4, ..., 10\}$ de joueurs et un nombre $m \in \{2, 3, ..., 5\}$ d'actions. Les configurations utilisées sont toutes celles pour lesquelles le nombre total de variables ($N \times m$) est inférieur ou égal à 25.

Joueurs Actions	3	4	5	6	7	8	9	10
2	6	8	10	12	14	16	18	20
3	9	12	15	18	21	24	27	30
4	12	16	20	24	28	32	36	40
5	15	20	25	30	35	40	45	50

FIGURE 7.20 – Nombre de variables pour chaque combinaisons de nombre de joueurs et nombre d'actions par joueur.

Pour chaque configuration des tests sont effectués sur 200 jeux différents. Afin d'avoir des mesures plus intéressantes, nous voulons surtout nous intéresser au temps de calcul d'un équilibre de Nash mixte. Dans le cas de l'énumération de supports, la solution retournée est le premier équilibre trouvé, qui sera un équilibre de Nash pur s'il en existe un. Donc, avant d'utiliser l'algorithme de Wilson, une recherche des équilibres purs est effectuée.

Lorsqu'il n'existe pas d'équilibre pur, plusieurs stratégies initiales jointes différentes seront utilisées. Lorsque le nombre de stratégies jointes différentes est supérieur à 25, nous choisissons 25 stratégies jointes aléatoirement parmi celles qui sont possibles.

Comparaisons des deux approches

Tout d'abord, concernant les types des jeux, nous n'observons pas de différence notable entre les RandomGame et les CovariantGame.

Lorsque nous avons des problèmes de petite taille, l'énumération de support est plus performante que le parcours de chemin, comme on peut le voir dans la figure 7.21. On peut aussi remarquer que pour quelques jeux dans le cas à 5 joueurs et 2 actions, lorsque la solution calculée a un support de taille plus élevée (8 ici) l'énumération commence à prendre un peu plus de temps que le parcours de chemin.

On peut aussi voir que, comme attendu, la méthode de Wilson trouve parfois des supports de taille supérieure à la méthode PNS, ce qui est normal étant donné que PNS s'arrête après avoir trouvé la première solution, qui sera celle avec le support de taille minimale. Cependant, il arrive parfois que Wilson trouve des solutions avec un support de taille plus petite que PNS. Ceci est le cas lorsqu'il y a un jeu dégénéré. PNS peut aussi permettre de trouver cette solution, cependant il la trouvera pour un support de taille supérieure à la solution calculée.

Lorsque nous commençons à avoir des jeux de plus grande taille, le parcours de chemin commence à être plus performant que l'énumération de support, comme le montrent les figures 7.22, 7.23, 7.24 et 7.25. Dans certains cas, lorsque l'énumeration trouve une solution, le calcul d'une solution avec support de taille plus élevée est plus rapide avec le parcours de chemin.

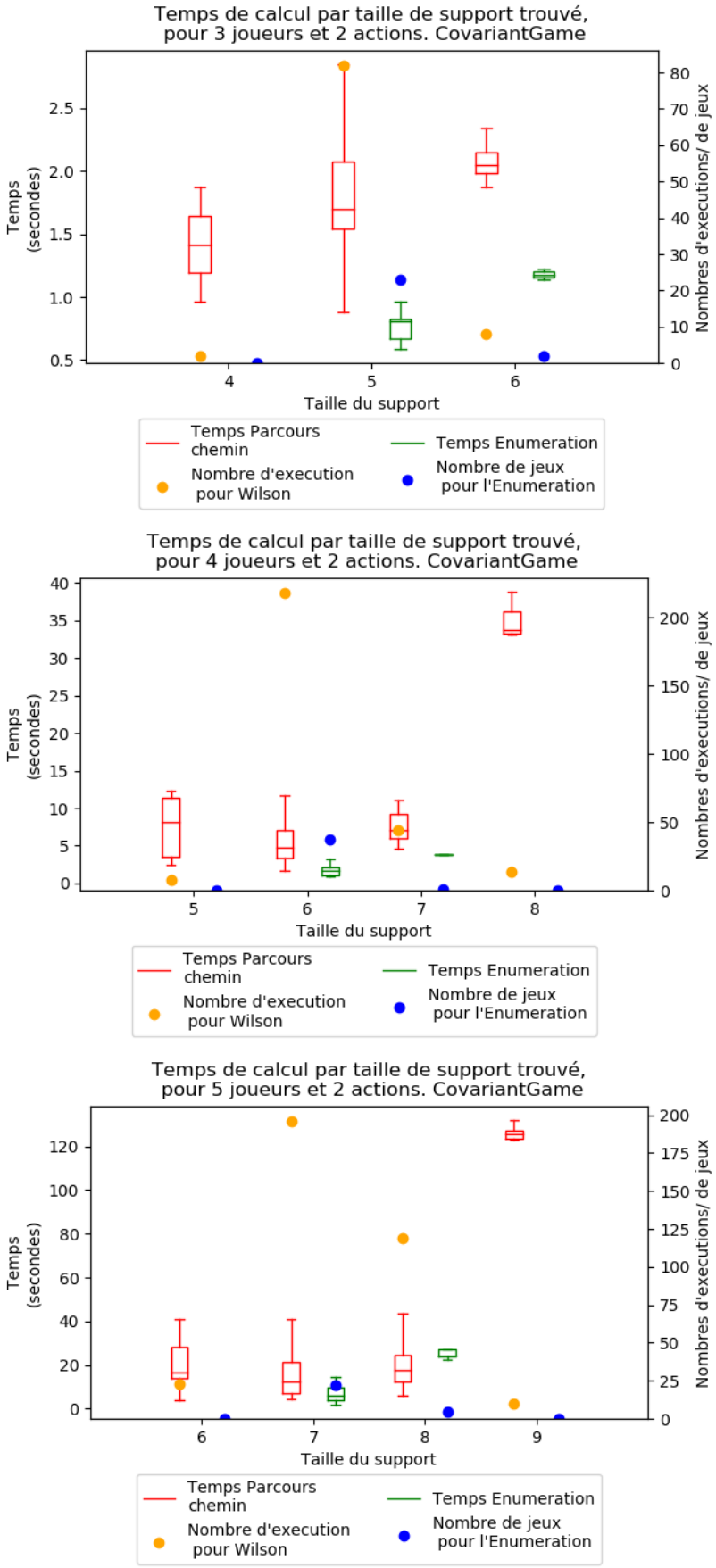


FIGURE 7.21 – Temps d'exécution des algorithmes pour 2 actions avec 3, 4 et 5 joueurs pour des CovariantGame.

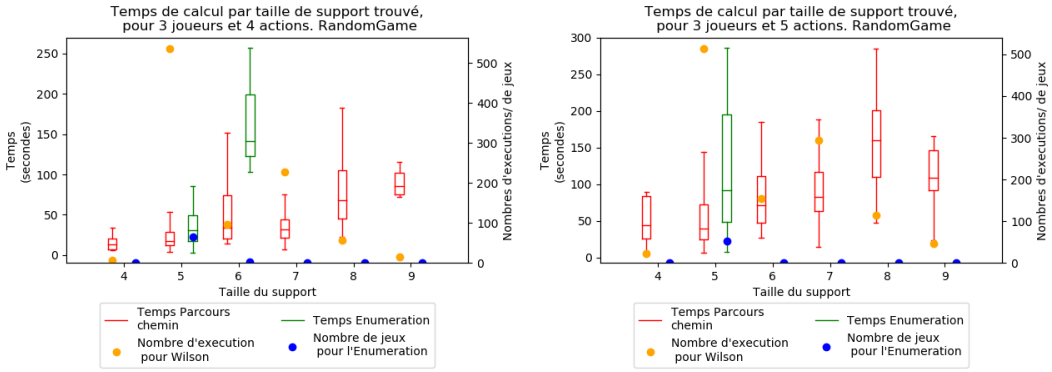


FIGURE 7.22 – Temps d'exécution des algorithmes pour 3 joueurs avec 4 actions (gauche) et 5 actions (droite) pour des RandomGame.

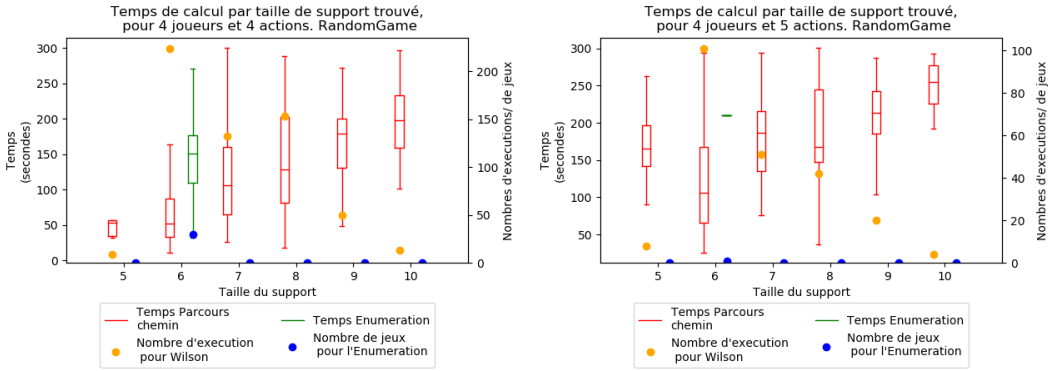


FIGURE 7.23 – Temps d'exécution des algorithmes pour 4 joueurs avec 4 actions (gauche) et 5 actions (droite) pour des RandomGame.

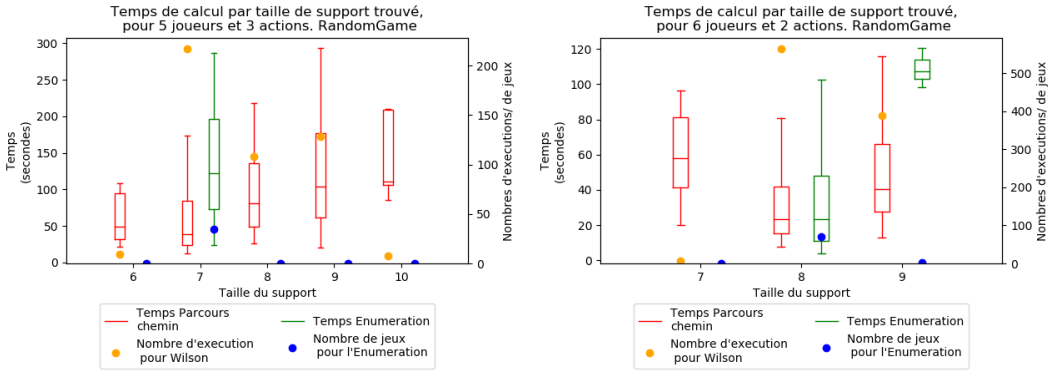


FIGURE 7.24 – Temps d'exécution des algorithmes pour 5 joueurs avec 3 actions (gauche) et 6 joueurs avec 2 actions (droite) pour des RandomGame.

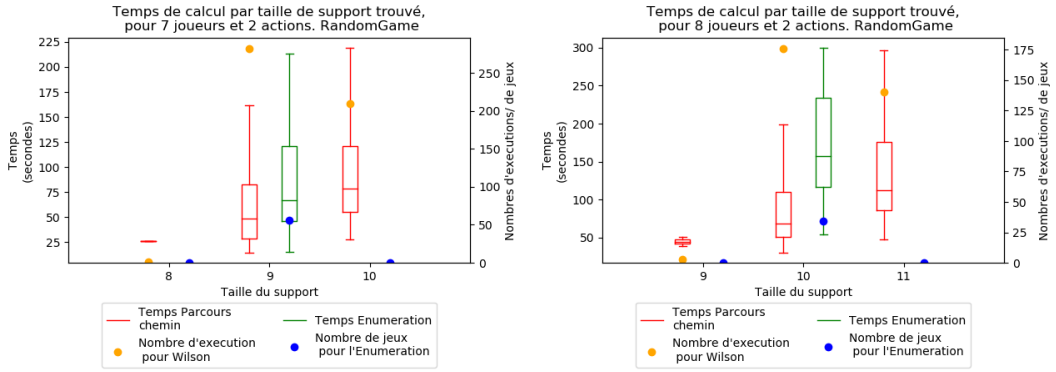


FIGURE 7.25 – Temps d'exécution des algorithmes pour 7 joueurs avec 2 actions (gauche) et 8 joueurs avec 2 actions (droite) pour des RandomGame.

En général, lorsque nous avons des PCP avec 18 variables ou plus, nous commençons a avoir un temps d'exécution qui dépasse très souvent la limite de 5 minutes et pas assez d'exécutions pour obtenir des temps analysables.

Notez qu'il y a toujours moins d'exécutions de l'énumération que de notre méthode, ceci est le cas car notre méthode peut prendre plusieurs points de départs différents pour un jeu (qui peuvent amener à la même solution) alors que l'énumération ne peut pas : il y a un seul "ordre" d'exécution possible (excepté bien sûr si nous définissons un ordre d'énumération particulier, mais cela revient à changer l'ordre des joueurs). Ceci est un avantage car il est possible de mettre en place des méthodes de parallélisation pour résoudre un jeu (et s'arrêter dès qu'une solution est trouvée).

Notez que les implémentations ne sont pas optimales, notamment pour PNS au niveau de la construction des supports, qui peut prendre du temps. Cependant les différences entre les méthodes restent notables avant que ces problèmes d'implémentation interviennent sur les performances, notamment dans le cas avec 3 joueurs et 4 ou 5 actions comme dans les figure 7.21, 7.22 et 7.23.

7.3.3 Jeux hypergraphiques

Nous avons aussi effectué une comparaison entre les performances des deux méthodes pour la résolution des jeux hypergraphiques. L'exécution et les mesures se font de la même manière que pour les jeux sous forme normale.

Les configurations impliquent $N \in \{4, 5, ...10\}$ joueurs et $m \in \{2, 3, 4, 5\}$ actions. Comme pour les jeux sous forme normale, nous vérifions au préalable si un jeu possède ou non un équilibre pur avant d'exécuter les algorithmes. Pour le parcours de chemin, le choix de la stratégie jointe initiale se fait comme décrit précédemment pour les jeux sous forme normale (premier joueur "statique" et au plus 25 ω^0 différents pour un jeu). De manière similaire aux jeux sous forme normale, nous ne résoudrons que des jeux avec au plus 25 variables pour le PCP correspondant.

Concernant la taille des sous-jeux locaux, ceux-ci auront 3 ou 4 joueurs. Il y a pour des jeux locaux de taille $|e|$ un minimum de $\lceil \frac{N-1}{|e|-1} \rceil$ et un maximum de $\frac{N!}{|e|*(N-|e|)!}$ jeux locaux possibles. Trois valeurs sont sélectionnées, le minimum, le maximum et la moyenne entre ces deux nombres.

Nous avons proposé un modèle de PCP incluant des variables additionnelles et dont les polynômes sont de degré moins élevé (voir Annexe C). Notre idée était que sachant que le calcul de bases de Gröbner est théoriquement simplement exponentiel en le

nombre de variables et doublement exponentiel en le degré maximum des polynômes du système à résoudre, cette modélisation pouvait permettre un calcul plus rapide des solutions d'un PCP (puisque résoudre un PCP nécessite de résoudre plusieurs systèmes d'équations polynomiales). Au final, ce nouveau modèle ne s'est pas révélé plus performant que le modèle de PCP "classique".

Pour un jeu hypergraphique nous effectuons trois exécutions différentes.

— Algorithme de Wilson.
— Algorithme de Wilson avec les variables auxiliaires y.
— Énumération de supports.

Aucune transformation sous forme normale n'est effectuée étant donné qu'il est possible de construire le PCP directement depuis la forme hypergraphique. Il est envisageable de le faire, cependant étant donné que le PCP du jeu sous forme normale est dérivé de celui sous la forme hypergraphique, les temps de résolution seront similaires. La seule différence est qu'il faut rajouter le temps de transformation du jeu hypergraphique vers la forme normale (qui augmente exponentiellement en la taille du jeu).

Concernant le choix des stratégies jointes initiales il se fait de la même manière que pour la forme normale, le premier joueur est "statique" et lorsqu'il y a plus de 25 stratégies jointes possibles, 25 sont choisies aléatoirement.

Comparaison des résultats

De manière similaire à ce qui a été vu pour les jeux sous forme normale, l'algorithme de Wilson a de meilleures performances pour des jeux de plus grande taille comme la figure 7.26 le montre. La méthode utilisant les variables auxiliaires a de moins bonnes performances et nous pouvons voir qu'il y a moins d'exécutions menées à terme que pour la méthode sans variables auxiliaires (les points marrons sont moins élevés que les points oranges).

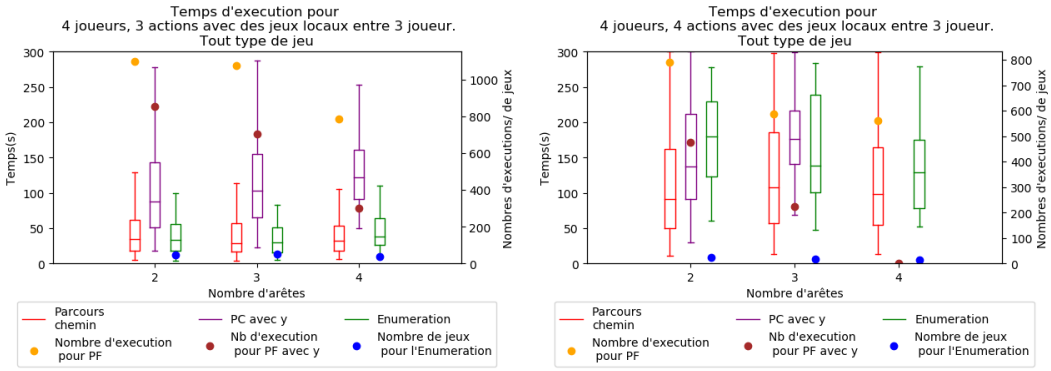


FIGURE 7.26 – Temps d'exécution des algorithmes pour 4 joueurs avec 3 actions (gauche) et 4 actions (droite) pour tous les types jeux confondus.

Il semble cependant que la structure du graphe (taille et nombre d'arêtes) ait une légère influence sur les temps d'exécution respectifs (et le nombre d'exécutions amenant à une solution) comme le montre la figure 7.27, il y a moins d'exécutions menées à terme lorsque les jeux locaux sont entre quatre joueurs au lieu de trois.

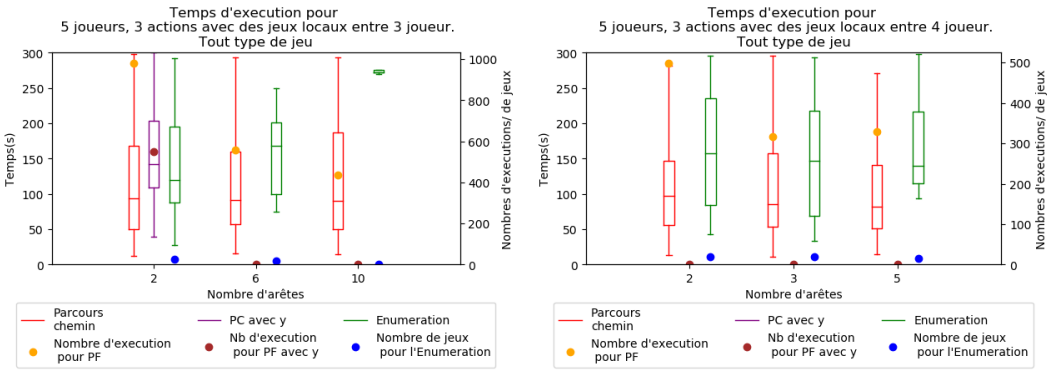


FIGURE 7.27 – Temps d'exécution des algorithmes pour 5 joueurs avec 3 actions et des jeux locaux entre 3 joueurs (gauche) et 4 joueurs (droite) pour tous les types de jeux confondus.

7.3.4 Jeux bayésiens sous forme normale

Les tests effectués sur les jeux bayésiens et bayésiens hypergraphiques ont servi à tester la transformation en un jeu hypergraphique et la résolution avec Wilson ou PNS.

Pour les jeux baysiens les configurations ont $N \in \{3, 4, 5, 6\}$ joueurs, $m \in \{2, 3\}$ actions et $t \in \{2, 3\}$ types. Ces choix ont été fait en fonction du nombre de joueurs après la transformation $(N \times t)$, du nombre de variables des PCP $(N \times t \times m)$ et des limites de calcul pour la méthode de Wilson. Comme pour les jeux sous forme normale, nous avons choisi les configurations avec au plus 25 variables pour le PCP obtenu. La figure 7.28 montre le nombre de variables pour les configurations possibles.

Joueurs Actions	3	4	5	6
2	12	16	20	24
3	18	24	30	36

Joueurs Actions	3	4	5
2	18	24	30
3	27	36	45

FIGURE 7.28 – Nombre de variable dans le PCP en fonction du nombre de joueurs et d'action pour 2 types (gauche) et 3 types (droite).

Résultats pour les jeux bayésiens

Il n'y a aucun problème à effectuer des transformations pour les jeux testés en jeux hypergraphiques en des temps raisonnables.

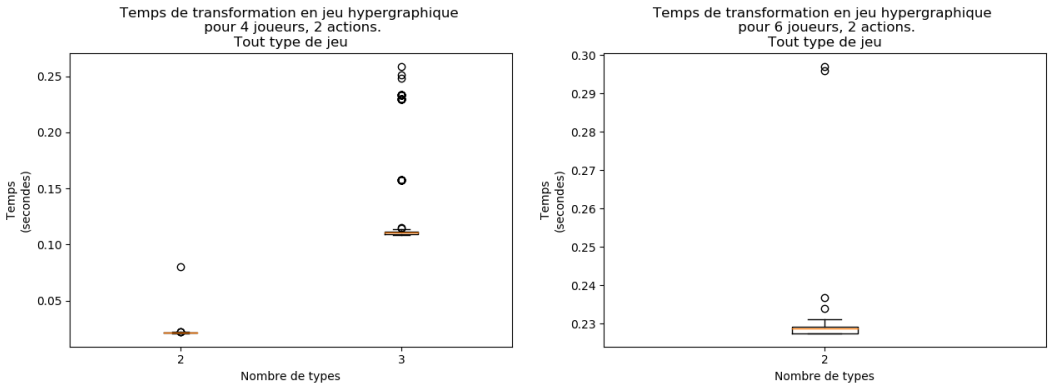


FIGURE 7.29 – Temps de transformation d'un jeu bayésien en un jeu hypergraphique avec 4 joueurs et 2 actions (gauche) et avec 6 joueurs et 2 actions (droite) pour tous les types de jeux confondus.

En ce qui concerne la résolution des jeux, compte tenu de la taille des jeux hypergraphiques et des performances vues précédemment pour les jeux hypergraphiques, la résolution atteint souvent la limite de 5 minutes.

Aucun des jeux avec 6 joueurs, 2 actions et 2 types n'est résolu et peu d'exécutions de notre méthode arrivent à terme pour les jeux avec 5 joueurs, 2 actions et 2 types (68 pour les RandomGame et 4 pour les CovariantGame). Même pour les configurations les plus petites, nous avons un temps d'exécution qui atteint vite la limite comme on peut le voir dans la figure 7.30

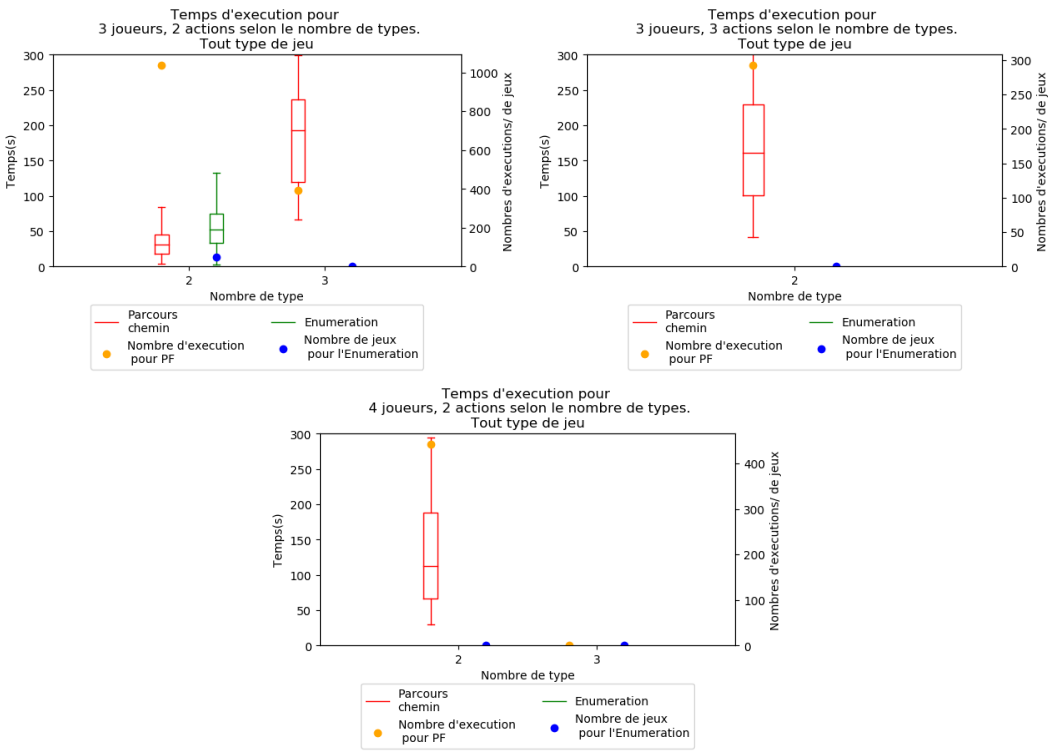


FIGURE 7.30 – Temps d'exécution des algorithmes pour des jeux bayésiens avec 3 joueurs et 2 actions, 3 joueurs et 3 actions et 4 joueurs et 2 actions pour tous types de jeux confondus.

7.3.5 Jeux bayesiens hypergraphiques

Les tests ont été effectués sur des configurations similaires à celles des jeux bayésiens sous forme normale. Les configurations ont $N \in \{4, 5, 6\}$ joueurs avec 2 actions et 2 types. Lorsque nous avons 4 joueurs nous testons aussi avec 3 actions et 3 types. Les graphes des jeux sont générés avec des arêtes entre 3 joueurs avec 3 nombres de jeux locaux différents, le minimum possible $\lceil \frac{N-1}{2} \rceil$, le maximum de $\frac{N!}{3*(N-3)!}$ et la valeur médiane.

Résultats

Le temps de transformation pour les jeux testés est relativement "insignifiant" comme on peut le voir dans la figure 7.31. Il faut toujours moins d'une seconde.

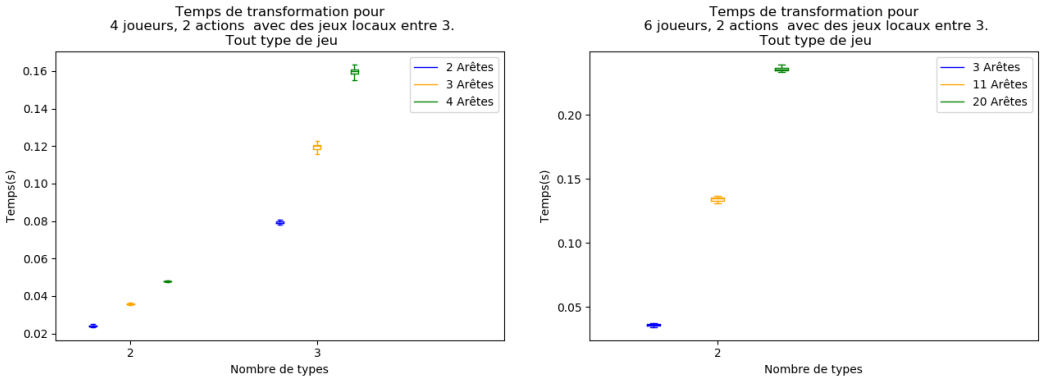


FIGURE 7.31 – Temps de transformation d'un jeu bayésien hypergraphique en jeu hypergraphique pour 4 joueurs et 2 actions (gauche) et pour 6 joueurs et 2 actions (droite) pour tous types de jeux confondus.

Comme pour les jeux bayésiens, nous atteignons le temps d'exécution limite très rapidement. Dans le cas avec 4 joueurs et 2 actions avec 2 arêtes de taille 3 (figure 7.32), l'énumération ne calcule les équilibres mixtes que de 2 jeux (tous types confondus, donc 2 sur 200). Tandis que notre méthode en calcule avec succès sur 48 jeux sur 200 (environ 800 exécutions sans timeout sur ces 48 jeux). Nous avons un comportement similaire pour 5 joueurs avec 2 actions dans la figure 7.33 où un seul calcul pour l'énumération s'est terminé.

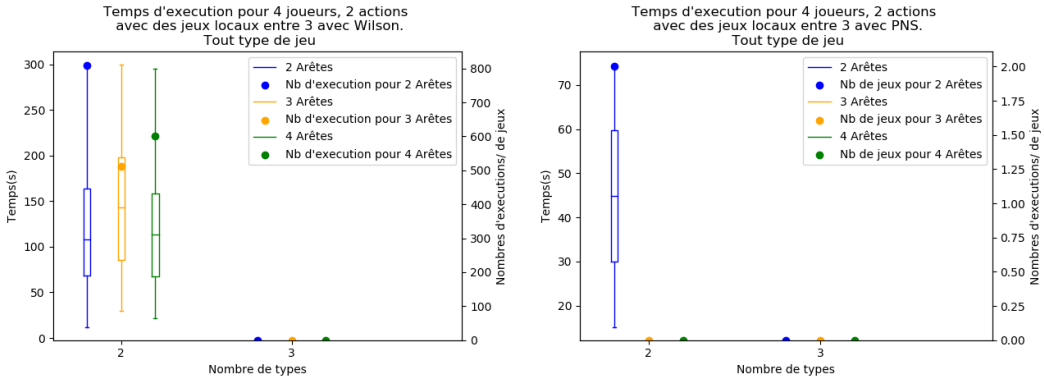


FIGURE 7.32 – Temps d'exécution d'un jeu bayésien hypergraphique pour 4 joueurs avec 2 actions avec la méthode de Wilson (gauche) et l'énumération (droite) pour tous types de jeux confondus.

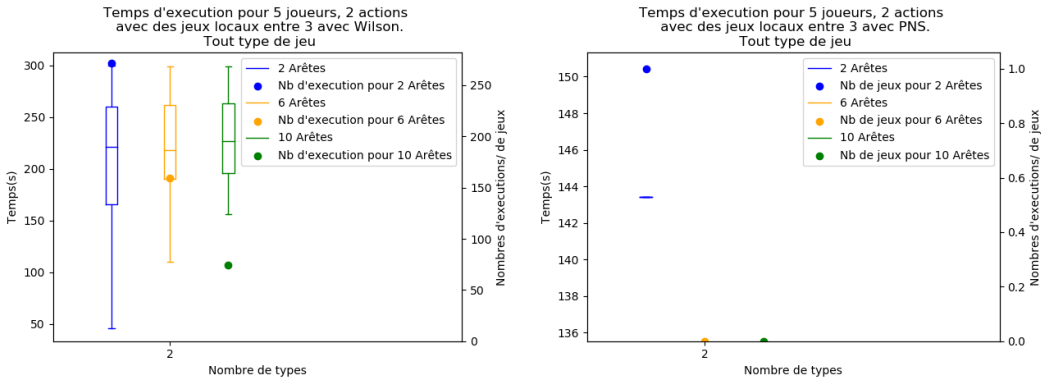


FIGURE 7.33 – Temps d'exécution d'un jeu bayésien hypergraphique pour 5 joueurs avec 2 actions avec la méthode de Wilson (gauche) et l'énumération (droite) pour tous types de jeux confondus.

7.3.6 Jeux de conservation

Compte tenu qu'un jeu de conservation est défini par les nombres d'attaquants, de défenseurs et un vecteur d'entiers ; il y a une multitude de configurations différentes pouvant être testées.

Nous avons testé des configurations où nous avons le même nombre d'attaquants et de défenseurs $N \in \{2, ..., 30\}$ et où le nombre de sites est $m \in \{N-1, N, N+1\}$. C'est à dire que si j'ai un jeu où $L = 15$ et $K = 15$ alors j'ai trois configurations pour lesquelles $m \in \{14, 15, 16\}$. Les vecteurs ressources sont générés aléatoirement ; la valeur d'un site varie uniformément entre 1 et 10. Pour chacune des configurations possibles, l'exécution de l'algorithme de Howson à été testée sur 50 jeux différents.

Le temps nécessaire à la création du jeu sous une forme polymatricielle commence à dépasser une seconde lorsque l'on a des jeux avec 18 attaquants, 18 défenseurs et 18 sites. La figure 7.34 représente l'évolution de ce temps. Les plus grand jeux testés sont entre 30 défenseurs et 30 attaquants avec 31 sites, dont la génération prend environ 7 secondes. Notez que l'implémentation actuelle ne profite pas de la "répétition" des tables, il en écrit 30 alors qu'elles sont identiques. Il est possible d'améliorer cet aspect là pour obtenir de meilleures performances lors de la création du jeu. Cependant il

sera nécessaire de représenter le jeu entier dans une matrice du schéma de Tucker pour l'exécution de l'algorithme de Howson.

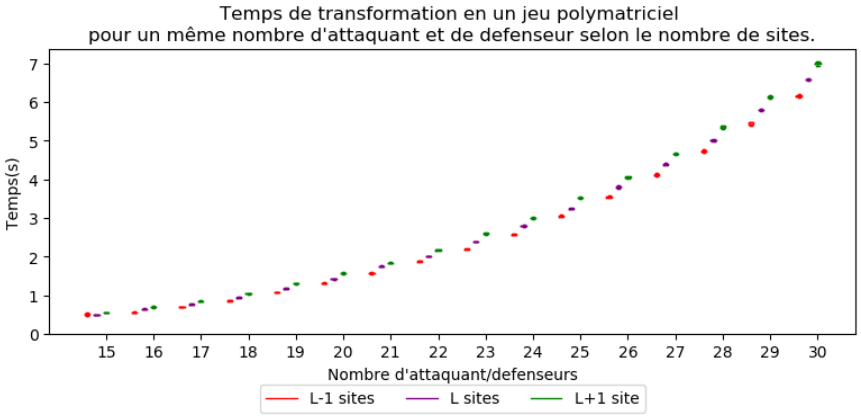


FIGURE 7.34 – Temps de transformation pour des jeux de conservation où L=K avec un nombre de stratégies variant en fonction du nombre d'attaquants/défenseurs.

La figure 7.35 représente le temps d'exécution pour trouver un équilibre de Nash jusqu'à des jeux où $L = K = 20$ et où le nombre de sites dépend du nombre d'attaquants/ défenseurs. A partir de 20 attaquants/défenseurs et sites, le temps limite de 300 secondes est dépassé systématiquement

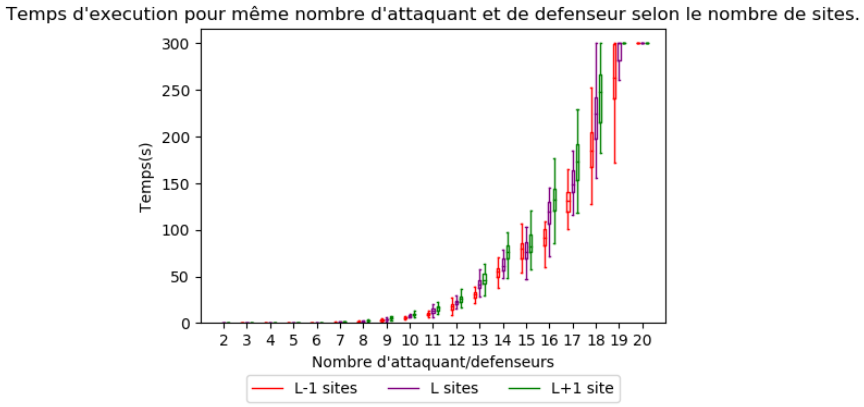


FIGURE 7.35 – Temps d'exécution pour des jeu de conservation où L=K avec un nombre de stratégies variant en fonction du nombre d'attaquants/défenseurs.

7.3.7 Jeux de conservation à information incomplète

Quelques tests préliminaires on été effectués sur les jeu de conservation à information incomplète.

On considère des configurations où il y a le même nombre d'attaquants et de défenseurs. Le nombre de sites varie comme précédemment avec $m \in \{N-1, N, N+1\}$ où N est le nombre d'attaquants ou de défenseurs. Nous considérons, en fonction du site, un vecteur de ressources max qui varie de 1 à 5. Donc si nous avons 4 sites nous avons 4 entiers différents représentant le nombre maximum de ressources.

Compte tenu de l'implémentation actuelle et pour des questions de mémoire, seuls quelques tests à petite échelle ont été effectués avec succès.

Le temps de génération du jeu bayésien polymatriciel à partir d'un tuple $(\hat{\lambda}, L, K, \mathcal{J}, \hat{P})$ augmente de manière exponentielle avec N même lorsqu'un seul site est observé par les attaquants. Ceci est aussi causé par la valeur max de ressources possibles (pouvant être 5 pour un ou plusieurs sites) et du nombre de sites, qui augmente le nombre de types de manière exponentielle (voir chapitre 6, section 6.4).

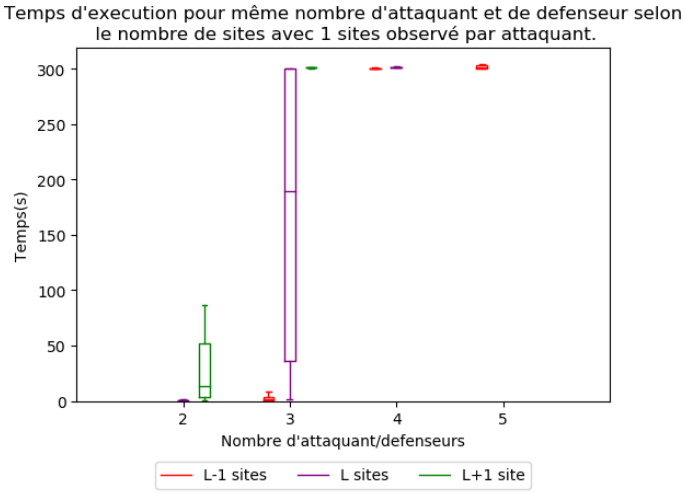


FIGURE 7.36 – Temps de transformation pour des jeux de conservation sous la forme bayésienne polymatricielle où L=K avec un nombre d'actions variant en fonction du nombre de joueurs.

La figure 7.37 représente le temps d'exécution pour plusieurs configurations avec un site observé. On peut voir que le temps d'exécution atteint vite la limite. Au dessus de 3 attaquants et défenseurs avec 4 sites, nous atteignons la limite de 5 minutes. Ceci correspond à un jeu bayésien polymatriciel à 6 joueurs, 4 actions où les attaquants peuvent avoir de 2 (observe un site avec au maximum une ressource) à 6 types (chaque attaquant observe un site avec au maximum 5 ressources) et les défenseurs de 16 (vecteur ressource avec 4 valeurs max à 1) à 1296 types (vecteur ressource avec 4 valeurs max à 5).

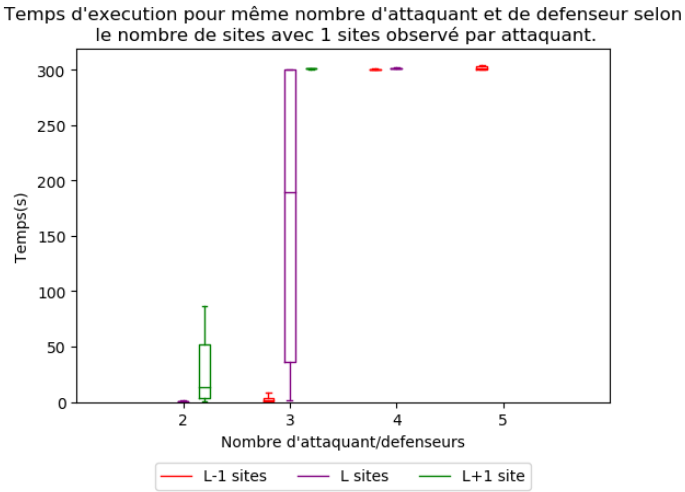


FIGURE 7.37 – Temps d'exécution pour des jeux de conservation sous la forme bayésienne polymatricielle où $L = K$ avec un nombre d'actions variant en fonction du nombre de joueurs.

7.4 Conclusion et perspectives

Les tests effectués permettent d'avoir une idée des performances des différents algorithmes pour les différents cadres proposés.

On peut voir, notamment avec les jeux de conservation (à information complète et incomplète) que la formulation de jeux sous la forme bayésienne polymatricielle permet un gain en espace considérable et l'exploitation, notamment dans le cas bayésien polymatriciel, d'algorithmes permettant de meilleures performances que l'utilisation de la forme normale.

En ce qui concerne la méthode de résolution basée sur la méthode de Wilson, elle est comparable avec l'énumération de supports et semble être plus "avantageuse" lorsque le nombre de joueurs augmente. En ce qui concerne la forme hypergraphique, notre méthode ne tire actuellement pas de bénéfice notable de la forme succincte, c'est donc un aspect qu'il faut continuer d'étudier.

Plusieurs aspects de la boite à outils gtnash peuvent être améliorés.

Il est possible de changer le choix de certain paramètres des algorithmes de calcul de bases de Gröbner pour les optimiser. L'utilisation du calcul de bases de Gröbner avant certaines opérations peut aussi potentiellement simplifier certains calculs.

Il est aussi possible d'étudier l'utilisation de méthodes "approchées" pour la résolution de systèmes (comme mentionné dans la conclusion du chapitre 5). La difficulté sera de ne pas faire des choix de noeuds incorrects.

Comme mentionné dans le chapitre 5, il est aussi intéressant d'étudier la création des PCP d'un jeu hypergraphique. Actuellement, nous pouvons écrire une forme factorisée des équations polynomiales, cependant ceci ne semble pas être avantageux en pratique, les fonctions de Sagemath développant ces formes factorisées.

Conclusion et perspectives

Conclusion

Dans cette thèse, nous avons proposé un cadre pour les jeux multi-joueurs à information incomplète : Le cadre des jeux bayésiens hypergraphiques. Ce cadre offre une représentation succincte des jeux. Par ailleurs, nous avons montré qu'un jeu bayésien hypergraphique peut être transformé, en temps polynomial en la taille du jeu, en un jeu hypergraphique à information complète. Ce résultat étend le théorème de [Howson et Rosenthal, 1974]. Il permet d'utiliser des algorithmes pour jeux hypergraphiques afin de résoudre des jeux bayésiens hypergraphiques. Cela offre une alternative avantageuse à la méthode classique de calcul d'équilibre de Nash dans les jeux bayésiens, qui consiste à les traduire en jeux à forme normale, de taille exponentiellement plus grande, puis à calculer un équilibre du jeu résultant. Les expérimentations effectuées montrent que pour les jeux bayésiens polymatriciels, il devient très vite coûteux de les résoudre en passant par la forme normale (à information complète), ce qui n'est pas le cas lorsqu'on les transforme d'abord en jeu polymatriciel à information complète.

Une nouvelle méthode algébrique et combinatoire de calcul d'équilibre de Nash exact a été proposée. Elle se base sur l'approche géométrique définie par [Wilson, 1971]. Un algorithme implémentant cette méthode a été décrit ainsi que des extensions aux jeux dégénérés et hypergraphiques. Nous avons montré expérimentalement lorsque que la taille du jeu augmente, notre méthode peut obtenir de meilleures performances que l'énumération de support, lorsque les deux méthodes utilisent des PCP.

En ce qui concerne les jeux bayésiens en forme normale ou hypergraphiques, la méthode de Wilson et l'énumération de supports permettent de calculer des équilibres de Nash "exacts" pour des jeux avec un nombre modéré de joueurs. La taille des jeux résolus est limitée par le nombre de variables du PCP correspondant.

Une boite à outil, *gtnash*, a été développée pour modéliser et résoudre différents types de jeux, incluant les jeux bayésiens hypergraphiques et leur spécialisations (graphiques, polymatricielles, ...). Elle implémente les contributions de cette thèse. Le projet GitLab de la bibliothèque gtnash est public [4]. La bibliothèque gtnash a servi de support à l'ensemble des expérimentations de cette thèse.

Nous avons enfin proposé une nouvelle formulation d'un problème de conservation de la biodiversité dans une situation de lutte contre le braconnage. Ce modèle peut être présenté sous forme normale ou sous une forme polymatricielle. Une version à information incomplète de ce modèle a aussi été proposée, tirant bénéfice de la forme bayésienne polymatricielle. Grâce à la forme polymatricielle (à information compléte) des jeux de conservation, il est possible de résoudre facilement des jeux de plus d'une dizaine d'attaquants, de défenseurs et de sites (20 joueurs au total et 10 actions). Les jeux de conservation à information incomplète sont plus difficiles à résoudre, bien sûr.

4. `https://forgemia.inra.fr/game-theory-tools-group/gtnash-git/`

Néanmoins, le passage par une forme bayésienne polymatricielle permet de résoudre des problèmes avec trois attaquants, trois défenseurs et 3 sites. Ceci correspond à des jeux bayésiens polymatriciels à six joueurs, 3 actions et 256 types joints dans les jeux locaux (64 de probabilité non nulle) !

Perspectives

Les travaux présentés dans cette thèse ouvrent plusieurs catégories de perspectives dans les domaines suivants :

— Modélisation des jeux à information incomplète.
— Algorithmique du calcul d'équilibres de Nash.
— Application à la conservation de la biodiversité.

En ce qui concerne les modèles étudiés, il serait relativement naturel d'étendre le cadre des jeux bayésiens hypergraphiques au cas des jeux séquentiels, et en particulier au cadre des "Partially Observed Stochastic Games (POSG)" [Hansen *et al.*, 2004]. Ce dernier cadre généralise la théorie des jeux (en forme normale) et le cadre des processus décisionnels de Markov partiellement observable [Kaelbling *et al.*, 1998]. La prise en compte des aspects séquentiels des problèmes de décision multi-joueurs serait particulièrement pertinente pour aborder des problèmes de conservation de la biodiversité. La représentation d'un jeu dans le cadre des POSG est plus coûteuse que celle d'un jeu bayésien, de même que le calcul d'un équilibre. Proposer une représentation concise et des algorithmes de calcul d'équilibre adaptés serait particulièrement intéressant.

En ce qui concerne l'algorithmique du calcul d'équilibres de Nash dans les jeux graphiques, plusieurs perspectives intéressantes s'offrent à nous. Tout d'abord, alors que l'on sait qu'un jeu polymatriciel peut se traduire en un "Linear Complementarity Problem", il est un peu frustrant de ne pas avoir réussi à traduire un jeu hypergraphique d'ordre k (i.e. dont les hyperarêtes sont de taille k au maximum) en un PCP dont les polynômes sont de degré k-1 ET sans augmenter le nombre de variables. La recherche (ou la démonstration de l'absence) d'une telle traduction nous semble prioritaire. Une perspective plus "algorithmique" et plus directe est d'exploiter la structure de l'hypergraphe d'un jeu hypergraphique afin d'optimiser l'ordre des variables, de manière à exploiter les indépendances entre joueurs pour construire et résoudre des sous-jeux de taille minimale. De manière assez directe également, il serait intéressant de pouvoir varier les solveurs utilisés au cours de l'algorithme de Wilson. L'intérêt d'utiliser un solveur exact (et donc lent) est d'éviter des erreurs dans le choix des pivots, liés à une mauvaise identification des équations qui "entrent" dans les ensembles Z et W. Utiliser un solveur approché permettrait de résoudre plus rapidement les systèmes d'équation, mais au risque d'identifier un mauvais pivot, lorsque deux systèmes ne variant que par une équation ont des solutions très proches. Mais ce cas ne se produit pas fréquemment. Aussi, l'utilisation réfléchie d'un solveur approché, puis d'un solveur exact en cas de risque de mauvaise identification de pivot, pourrait permettre un gain de temps appréciable dans le calcul d'équilibres et une augmentation de la taille des jeux résolus.

Enfin, sur le plan applicatif, le maintien et l'extension de la boite à outils gtnash ainsi que son utilisation dans différents domaines appliqués faisant appel à la théorie des jeux est une perspective importante. En ce qui concerne l'application à la conservation de la biodiversité, une mise à disposition du cadre est des outils développés est une priorité. Celles-ci devront s'accompagner d'une mise en oeuvre sur des problèmes de conservation réels, en collaboration avec des experts du domaine.

Bibliographie

[Amor *et al.*, 2020] Amor, N. B., Fargier, H., Sabbadin, R. et Trabelsi, M. (2020). Ordinal Polymatrix Games with Incomplete Information. *In KR*, pages 99–108, Rhodes, Greece.

[Ashlagi *et al.*, 2006] Ashlagi, I., Monderer, D. et Tennenholtz, M. (2006). Resource selection games with unknown number of players. *In AAMAS*, pages 819–825.

[Aumann et Brandenburger, 1995] Aumann, R. et Brandenburger, A. (1995). Epistemic Conditions for Nash Equilibrium. *Econometrica*, 63(5):1161.

[Aumann, 1974] Aumann, R. J. (1974). Subjectivity and correlation in randomized strategies. *Journal of mathematical Economics*, 1(1):67–96.

[Aumann, 1987] Aumann, R. J. (1987). Correlated equilibrium as an expression of bayesian rationality. *Econometrica*, pages 1–18.

[Barman *et al.*, 2015] Barman, S., Ligett, K. et Piliouras, G. (2015). Approximating nash equilibria in tree polymatrix games. *In SAGT*, pages 285–296. Springer.

[Battigalli et Bonanno, 1999] Battigalli, P. et Bonanno, G. (1999). Recent results on belief, knowledge and the epistemic foundations of game theory. *Research in Economics*, 53(2):149–225.

[Berg et Sandholm, 2017] Berg, K. et Sandholm, T. (2017). Exclusion method for finding nash equilibrium in multiplayer games. *In AAAI*, volume 31.

[Bhat et Leyton-Brown, 2004] Bhat, N. A. R. et Leyton-Brown, K. (2004). Computing nash equilibria of action-graph games. *In UAI*.

[Boryczka et Juszczuk, 2013] Boryczka, U. et Juszczuk, P. (2013). Differential evolution as a new method of computing nash equilibria. *In Transactions on Computational Collective Intelligence IX*, pages 192–216. Springer.

[Brandenburger, 2008] Brandenburger, A. (2008). Epistemic game theory : an overview. *The New Palgrave Dictionary of Economics, 2nd edition*.

[Brandenburger et Dekel, 1993] Brandenburger, A. et Dekel, E. (1993). Hierarchies of beliefs and common knowledge. *Journal of Economic Theory*, 59(1):189–198.

[Cai et Wurman, 2005] Cai, G. et Wurman, P. (2005). Monte Carlo approximation in incomplete information sequential auction games. *Decision Support Systems*, 39(2):153–168.

[Ceppi *et al.*, 2009] Ceppi, S., Gatti, N. et Basilico, N. (2009). Computing bayes-nash equilibria through support enumeration methods in bayesian two-player strategic-form games. *In IAT*, pages 541–548.

[Chapman *et al.*, 2010] Chapman, A., Farinelli, A., de Cote, E., Rogers, A. et Jennings, N. (2010). A distributed algorithm for optimising over pure strategy Nash equilibria. *In AAAI*, pages 749–755.

[Chen et Deng, 2005] Chen, X. et Deng, X. (2005). 3-nash is ppad-complete. *In ECCC*, volume 134, pages 2–29. Citeseer.

[Chen et Deng, 2006] Chen, X. et Deng, X. (2006). Settling the Complexity of two-Player Nash-Equilibrium. *In FOCS*, pages 261–272.

[Colyvan *et al.*, 2011] Colyvan, M., Justus, J. et Regan, H. (2011). The conservation game. *Biological Conservation*, 144:1246–1253.

[Conitzer et Sandholm, 2008] Conitzer, V. et Sandholm, T. (2008). New complexity results about Nash equilibria. *Games and Economic Behavior*, 63(2):621–641.

[Cottle et Dantzig, 1967] Cottle, R. W. et Dantzig, G. B. (1967). Complementary pivot theory of mathematical programming. Rapport technique, STANFORD UNIV CA OPERATIONS RESEARCH HOUSE.

[Cox *et al.*, 2015] Cox, D. A., Little, J. B. et O'Shea, D. (2015). *Ideals, Varieties, and Algorithms*. Springer, 4 édition.

[Daskalakis, 2006] Daskalakis, C. (2006). Computing pure Nash equilibria via Markov random fields.

[Daskalakis *et al.*, 2006] Daskalakis, C., Fabrikant, A. et Papadimitriou, C. (2006). The game world is flat : The complexity of Nash equilibria in succinct games. *In ICALP*, pages 513–524. Springer.

[Daskalakis *et al.*, 2009] Daskalakis, C., Goldberg, P. et Papadimitriou, C. (2009). The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259.

[Daskalakis et Papadimitriou, 2005] Daskalakis, C. et Papadimitriou, C. H. (2005). Three-player games are hard. *In ECCC*, volume 139, pages 81–87.

[Datta, 2010] Datta, R. S. (2010). Finding all Nash equilibria of a finite game using polynomial algebra. *Econonmics Theory*, 42:55–96.

[De Nittis *et al.*, 2018] De Nittis, G., Marchesi, A. et Gatti, N. (2018). Computing the strategy to commit to in polymatrix games. *In AAAI*.

[Dekel et Siniscalchi, 2015] Dekel, E. et Siniscalchi, M. (2015). Epistemic Game Theory. *In Handbook of Game Theory with Economic Applications*, volume 4, pages 619–702. Elsevier.

[Deligkas *et al.*, 2016] Deligkas, A., Fearnley, J., Igwe, T. P. et Savani, R. (2016). An empirical study on computing equilibria in polymatrix games. *In AAMAS*, pages 186–195. International Foundation for Autonomous Agents and Multiagent Systems.

[Deligkas *et al.*, 2015] Deligkas, A., Fearnley, J., Savani, R. et Spirakis, P. (2015). Computing approximate Nash equilibria in polymatrix games. *Algorithmica*, 77(2):487–514.

[Dickenstein *et al.*, 1991] Dickenstein, A., Fitchas, N., Giusti, M. et Sessa, C. (1991). The membership problem for unmixed polynomial ideals is solvable in single exponential time. *Discrete Applied Mathematics*, 33(1):73–94.

[Fabrikant *et al.*, 2004] Fabrikant, A., Papadimitriou, C. et Talwar, K. (2004). The complexity of pure nash equilibria. *In STOC*, pages 604–612.

[Fagin *et al.*, 1995] Fagin, R., Moses, Y., Halpern, J. et Vardi, M. (1995). *Reasoning about knowledge*.

[Fang *et al.*, 2013] Fang, F., Jiang, A. X. et Tambe, M. (2013). Designing optimal patrol strategy for protecting moving targets with multiple mobile resources. *In OPTMAS*.

[Fang *et al.*, 2016] Fang, F., Nguyen, T., Pickles, R., Lam, W., Clements, G. R., An, B., Singh, A., Tambe, M. et Lemieux, A. (2016). Deploying paws : Field optimization of the protection assistant for wildlife security. *In AAAI*, pages 3966–3973.

[Fang *et al.*, 2017] Fang, F., Nguyen, T., Sinha, A., Gholami, S., Plumptre, A., Joppa, L., Tambe, M., Driciru, M., Wanyama, F., Rwetsiba, A., Critchlow, R. et Beale, C. (2017). Predicting poaching for wildlife protection. *IBM Journal of Research and Development*, 61:3 :1–3 :12.

[Fang *et al.*, 2015] Fang, F., Stone, P. et Tambe, M. (2015). When security games go green : Designing defender strategies to prevent poaching and illegal fishing. *In IJCAI*, pages 2589–2595.

[Fargier *et al.*, 2021a] Fargier, H., Jourdan, P. et Sabbadin, R. (2021a). Jeux bayésiens hypergraphiques. *In RJCIA (PFIA)*.

[Fargier *et al.*, 2021b] Fargier, H., Jourdan, P. et Sabbadin, R. (2021b). Trouver un équilibre de nash mixte algébrique dans les jeux sous forme normale et succincts. *In CNIA*, pages pp–6.

[Fargier *et al.*, 2022a] Fargier, H., Jourdan, P. et Sabbadin, R. (2022a). On hypergraphical bayesian games. *In ICTAI*.

[Fargier *et al.*, 2022b] Fargier, H., Jourdan, P. et Sabbadin, R. (2022b). A path-following polynomial equations systems approach for computing nash equilibria. *In AAMAS*, pages 418–426.

[Filar et Vrieze, 1997] Filar, J. et Vrieze, K. (1997). *Competitive Markov Decision Processes*. Springer.

[Franc *et al.*, 2010] Franc, A., Goulard, M. et Peyrard, N. (2010). Chordal graphs to identify graphical model solutions of maximum of entropy under constraints on marginals. *SIAM J. Discret. Math.*, 24(3):1104–1116.

[Frank et Sarkar, 2010] Frank, D. M. et Sarkar, S. (2010). Group decisions in biodiversity conservation : Implications from game theory. *PLoS ONE*, 5(5):10p.

[Gibson et Marks, 1995] Gibson, C. C. et Marks, S. A. (1995). Transforming rural hunters into conservationists : An assessment of community-based wildlife management programs in africa. *World Development*, 23:941–957.

[Goldberg et Papadimitriou, 2006] Goldberg, P. W. et Papadimitriou, C. H. (2006). Reducibility among equilibrium problems. *In STOC*, pages 61–70.

[Gottlob *et al.*, 2005] Gottlob, G., Greco, G. et Scarcello, F. (2005). Pure Nash equilibria : Hard and easy games. *Journal of Artificial Intelligence Research*, 24:357–406.

[Govindan et Wilson, 2003] Govindan, S. et Wilson, R. (2003). A Global Newton Method to Compute Nash Equilibria. *Journal of Economic Theory*, 110(1):65–86.

[Govindan et Wilson, 2004] Govindan, S. et Wilson, R. (2004). Computing Nash Equilibria by Iterated Polymatrix Approximation. *Journal of Economic Dynamics and Control*, 28(7):1229–1241.

[Gowda, 2017] Gowda, M. S. (2017). Polynomial complementarity problems. *Pac. J. Optim*, 3:227–241.

[Grubshtein et Meisels, 2012] Grubshtein, A. et Meisels, A. (2012). Finding a nash equilibrium by asynchronous backtracking. *In CP*, pages 925–940. Springer.

[Hansen *et al.*, 2004] Hansen, E. A., Bernstein, D. S. et Zilberstein, S. (2004). Dynamic programming for partially observable stochastic games. *In AAAI*, volume 4, pages 709–715.

[Harrenstein *et al.*, 2001] Harrenstein, P., van der Hoek, W., Meyer, J. et Witteveen, C. (2001). Boolean games. *In TARK*, pages 287–298.

[Harsanyi, 1967] Harsanyi, J. C. (1967). Games with incomplete information played by "Bayesian" players, I–III Part I. The basic model. *Management science*, 14(3): 159–182.

[Harsanyi, 1968a] Harsanyi, J. C. (1968a). Games with incomplete information played by "Bayesian" players, part III. The basic probability distribution of the game. *Management Science*, 14(7):486–502.

[Harsanyi, 1968b] Harsanyi, J. C. (1968b). Games with incomplete information played by "Bayesian" players part II. Bayesian equilibrium points. *Management Science*, 14(5):320–334.

[Herings et Peeters, 2010] Herings, J. J. et Peeters, R. (2010). Homotopy methods to compute equilibria in game theory. *Economic Theory*, 42:119–156.

[Herings et van den Elzen, 2002] Herings, P. J. et van den Elzen, A. (2002). Computation of the Nash equilibrium selected by the tracing procedure in n-person games. *Games and Economic Behavior*, 38(1):89–117.

[Howson et Rosenthal, 1974] Howson, J. et Rosenthal, R. (1974). Bayesian equilibria of finite two-person games with incomplete information. *Management Science*, 21(3):313–315.

[Howson, 1972] Howson, J. T. (1972). Equilibria of polymatrix games. *Management Science*, 18(5-part-1):312–318.

[Jiang et Leyton-Brown, 2010] Jiang, A. et Leyton-Brown, K. (2010). Bayesian action-graph games. *In NIPS*, volume 23, pages 991–999.

[Jiang *et al.*, 2011] Jiang, A., Leyton-Brown, K. et Bhat, N. (2011). Action-graph games. *Games and Economic Behavior*, 71(1):141–173.

[Jiang et Safari, 2010] Jiang, A. X. et Safari, M. (2010). Pure Nash equilibria : complete characterization of hard and easy graphical games. *In AAMAS*, pages 199–206. International Foundation for Autonomous Agents and Multiagent Systems.

[Johnson *et al.*, 1988] Johnson, D. S., Papadimitriou, C. H. et Yannakakis, M. (1988). How easy is local search ? *Journal of computer and system sciences*, 37(1): 79–100.

[Kaelbling *et al.*, 1998] Kaelbling, L. P., Littman, M. L. et Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134.

[Kearns *et al.*, 2001] Kearns, M., Littman, M. L. et Singh, S. (2001). Graphical Models for Game Theory. *UAI*, 1:253–260.

[Kiekintveld *et al.*, 2009] Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordónez, F. et T., M. (2009). Computing optimal randomized resource allocations for massive security games. *In Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 689–696.

[Koller et Milch, 2001] Koller, D. et Milch, B. (2001). Multi-agent influence diagrams for representing and solving games. *Games and economic behavior*, 45(1):181–221.

[Lemke, 1965] Lemke, C. E. (1965). Bimatrix equilibrium points and mathematical programming. *Management science*, 11(7):681–689.

[Lemke et Howson, 1964] Lemke, C. E. et Howson, J. T. (1964). Equilibrium points of bimatrix games. *Journal of the Society for industrial and Applied Mathematics*, 12(2):413–423.

[Lipton et Markakis, 2004] Lipton, R. et Markakis, E. (2004). Nash Equilibria via Polynomial Equations. *In LATIN*.

[Luce *et al.*, 1957] Luce, R. D., Raiffa, H. et Teichmann, T. (1957). Games and decisions.

[Mayr, 1997] Mayr, E. W. (1997). Some Complexity Results for Polynomial Ideals. *Journal of Complexity*, 13(3):303–325.

[McKelvey et McLennan, 1996] McKelvey, R. D. et McLennan, A. (1996). Computation of equilibria in finite games. *Handbook of computational economics*, 1:87–142.

[McKelvey *et al.*, 2014] McKelvey, R. D., McLennan, A. M. et Turocy, T. L. (2014). Gambit : Software Tools for Game Theory, Version 16.0.0.

[Meggido et Papadimitriou, 1989] Meggido, N. et Papadimitriou, C. H. (1989). A note on total functions, existence theorems, and computational complexity. *Tech. report, IBM, Tech. Rep.*

[Myerson, 1991] Myerson, R. B. (1991). *Game theory*. Harvard university press.

[Nash, 1950] Nash, J. F. (1950). Equilibrium points in n-person games. *PNAS*, 36(1): 48–49.

[Nebel, 2000] Nebel, B. (2000). On the compilability and expressive power of propositional planning formalisms. *Journal of Artificial Intelligence Research*, 12:271–315.

[Nisan, 2007] Nisan, N., éditeur (2007). *Algorithmic game theory*. Cambridge University Press, Cambridge ; New York. OCLC : ocn122526907.

[Nudelman *et al.*, 2004] Nudelman, E., Wortman, J., Shoham, Y. et Leyton-Brown, K. (2004). Run the GAMUT : A Comprehensive Approach to Evaluating Game-Theoretic Algorithms. *AAMAS*, 4:880–887.

[Ortiz et Kearns, 2003] Ortiz, L. et Kearns, M. (2003). Nash propagation for loopy graphical games. *In NIPS*, volume 15, pages 817–824.

[Osborne et Rubinstein, 1994] Osborne, M. J. et Rubinstein, A. (1994). *A course in game theory*. MIT press.

[Papadimitriou et Roughgarden, 2008] Papadimitriou, C. et Roughgarden, T. (2008). Computing correlated equilibria in multi-player games. *Journal of the ACM (JACM)*, 55(3):1–29.

[Papadimitriou, 1994] Papadimitriou, C. H. (1994). On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences*, 48(3):498–532.

[Pita *et al.*, 2008] Pita, J., Jain, M., Marecki, J., Ordóñez, F., Portway, C., Tambe, M., Western, C., Paruchuri, P. et Kraus, S. (2008). Deployed armor protection : the application of a game theoretic model for security at the los angeles international airport. *In AAMAS*, pages 125–132.

[Porter *et al.*, 2008] Porter, R., Nudelman, E. et Shoham, Y. (2008). Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior*, 63(2):664–669.

[Redpath *et al.*, 2018] Redpath, S. M., Keane, A., Andrén, H., Baynham-Herd, Z., Bunnefeld, N., Duthie, A. B., Frank, J., Garcia, C. A., Månsson, J., Nilsson, L., Pollard, C. R. J., Rakotonarivo, O. S., Salk, C. F. et Travers, H. (2018). Games as tools to address conservation conflicts. *Trends in Ecology & Evolution*, 33(6):415–426.

[Rosenthal, 1973] Rosenthal, R. W. (1973). A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2(1):65–67.

[Sandholm *et al.*, 2005] Sandholm, T., Gilpin, A. et Conitzer, V. (2005). Mixed-integer programming methods for finding Nash equilibria. *In AAAI*, pages 495–501.

[Simon et Wojtczak, 2017] Simon, S. et Wojtczak, D. (2017). Constrained pure nash equilibria in polymatrix games. *In AAAI*, pages 691–697.

[Singh *et al.*, 2004] Singh, S., Soni, V. et Wellman, M. (2004). Computing approximate bayes-nash equilibria in tree-games of incomplete information. *In ICEC*, pages 81–90.

[Soni *et al.*, 2007] Soni, V., Singh, S. et Wellman, M. P. (2007). Constraint satisfaction algorithms for graphical games. *In AAMAS*, pages 1–8.

[Tambe, 2011] Tambe, M. (2011). *Security and game theory : algorithms, deployed systems, lessons learned*. Cambridge university press.

[van der Laan *et al.*, 1987] van der Laan, G., Talman, A. et van der Heyden, L. (1987). Simplicial variable dimension algorithms for solving the nonlinear complementarity problem on a product of unit simplices using a general labelling. *Mathematics of Operations Research*, 12(3):377–397.

[Vickrey et Koller, 2002] Vickrey, D. et Koller, D. (2002). Multi-agent algorithms for solving graphical games. *In AAAI*, pages 345–351.

[Von Neumann et Morgenstern, 1944] Von Neumann, J. et Morgenstern, O. (1944). *Theory of games and economic behavior*. Princeton university press.

[Von Stackelberg, 1934] Von Stackelberg, H. (1934). *Marktform und gleichgewicht*. J. springer.

[Von Stengel, 2002] Von Stengel, B. (2002). Computing equilibria for two-person games. *Handbook of game theory with economic applications*, 3:1723–1759.

[Wahbi et Brown, 2016] Wahbi, M. et Brown, K. (2016). A distributed asynchronous solver for nash equilibria in hypergraphical games. *In ECAI*, pages 1291–1299. IOS Press.

[Wilson, 1971] Wilson, R. (1971). Computing equilibria of N-person games. *SIAM Journal on Applied Mathematics*, 21(1):80–87.

[Yang *et al.*, 2014] Yang, R., Ford, B., Tambe, M. et Lemieux, A. (2014). Adaptive resource allocation for wildlife protection against illegal poachers. *In AAMAS*, AAMAS '14, page 453–460, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Yanovskaya, 1968] Yanovskaya, E. B. (1968). Equilibrium points in polymatrix games. *Litovskii Matematicheskii Sbornik*, 8:381–384.

[Yin *et al.*, 2012] Yin, Z., Jiang, A. X., Johnson, M. P., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., Tambe, M. et Sullivan, J. P. (2012). Trusts : Scheduling randomized patrols for fare inspection in transit systems. *In IAAI*.

Annexes

Annexe A

Version matricielle de l'algorithme Lemke Howson

Nous allons décrire la version de l'algorithme de Lemke Howson décrite dans [McKelvey et McLennan, 1996] et l'appliquer sur l'exemple 12 du chapitre 1. En reprenant le problème de complémentarité il est aussi possible de le modéliser par l'équation suivante :

$$Ux + y = q$$

où $x \geqslant 0$, $y \geqslant 0$ et $x.y = 0$.

On peut donc écrire :

$$\begin{bmatrix} U & I_m \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix} = q$$

où I_m est la matrice identité de dimension m. Nous définissons $A = \begin{bmatrix} U & I_m \end{bmatrix}$ et $z = \begin{bmatrix} x \\ y \end{bmatrix}$

Nous utilisons une collection d'indices $\beta = \{b_1, ..., b_m\}$ avec $1 \leqslant b_i \leqslant 2m$ où chaque indice désigne une variable de la base (les indices des colonnes de A). Les colonnes 1 à m de A correspondent aux variables x (qui donneront les probabilités non normalisées des stratégies) tandis que les colonnes de $m + 1$ à $2m$ correspondent aux variables de y (les variables d'écart).

Une vecteur z est une solution du LCP s'il y a une base β telle que $z_i = 0, \forall i \notin \beta$. Pour toute base β il y a une solution z. Une base β est dite faisable si sa solution z est faisable. Une base β est complémentaire si pour $1 \leqslant j \leqslant m$, soit j soit $m + j$ est dans β. Une base β est i-presque complémentaire si pour $1 \leqslant j \leqslant m$ où $j \neq i$, soit $j \notin \beta$ soit $m + j \notin \beta$.

Pour appliquer l'algorithme, il faut partir d'une base β_0 faisable et complémentaire. En général cette base de départ sera la solution externe, elle comprendra les indices de toutes les variables d'écart ($m + 1$ à $2m$) correspondant à $x = \mathbf{0}$ et $y = \mathbf{1}$. Pour un indice i choisi, une séquence de bases i-presque complémentaires sera parcourue jusqu'à obtenir une nouvelle solution complémentaire. Pour la base β_0, avec $q^{\beta_0} = (B^{\beta_0})^{-1}q$ et $A^{\beta_0} = (B^{\beta_0})^{-1}A$, nous définissons :

$$T^{\beta_0} = \begin{bmatrix} -q^{\beta_0} & A^{\beta_0} \end{bmatrix}$$

Pour l'étape suivante, la base β_1 aura un élément diffèrent de β_0 : $\beta_1 = \beta_0 - \{r\} \cup \{s\}$ où s est l'indice ajouté à la base et r est l'indice retiré de la base.

Lorsque l'on part d'une base complémentaire β_0, l'indice s vérifie $s \notin \beta_0$. L'indice $r = b_h^{\beta_0}$ a remplacer doit être choisi en fonction de $s = b_h^{\beta_1}$, h est un indice $1 \leqslant h \leqslant m$.

Les valeurs de r et s sont les indices de colonnes dans A et h est un indice de ligne de A. La valeur de h est un élément de l'ensemble $S = \{j : a^{\beta}_{js} > 0\}$ où $a^{\beta_0} \in A^{\beta_0}$, c'est à dire les indices des lignes de la colonne s de A^{β_0} où l'élément est supérieur à zéro. L'indice h est choisi dans l'ensemble :

$$S_0 = argmin\{\frac{q^{\beta}_j}{a^{\beta}_{js}} : j \in S\}$$

Si S_0 a plusieurs éléments alors ceci signifie que le jeu est dégénéré et il faut ajouter une étape supplémentaires pour le choix de l'indice h. On calcule successivement les ensembles suivants, pour $k = 1, ..., 2m$ et on s'arrête dès que Sk est un singleton :

$$S_k = argmax\{\frac{a^{\beta}_{jk}}{a^{\beta}_{js}} : j \in S_{k-1}\}$$

Lorsque S_k est un singleton alors l'indice de S_k est sélectionné.

La matrice T^{β_1} est définie de la manière suivante :

$$T^{\beta_1} = P^{\beta_1\beta_0}\begin{bmatrix} -q^{\beta_0} & A^{\beta_0}\end{bmatrix} = P^{\beta_1\beta_0}T^{\beta_0}$$

La matrice de pivot $P^{\beta_1\beta_0}$ est une matrice identité de dimension m où la colonne h est modifiée ainsi :

$$P^{\beta_1\beta_0} = \begin{bmatrix} 1 & & & -\frac{a^{\beta_0}_{1s}}{a^{\beta_0}_{hs}} & & \\ & \ddots & & \vdots & & \\ & & 1 & -\frac{a^{\beta_0}_{h-1,s}}{a^{\beta_0}_{hs}} & & \\ & & & \frac{1}{a^{\beta_0}_{hs}} & & \\ & & & -\frac{a^{\beta_0}_{h+1,s}}{a^{\beta_0}_{hs}} & 1 & \\ & & & & & \ddots \\ & & & -\frac{a^{\beta_0}_{m,s}}{a^{\beta_0}_{hs}} & & 1 \end{bmatrix}$$

Après avoir obtenu T^{β_1}, le prochain élément à ajouter dans la base sera j si $m + j$ est l'indice précédemment retiré de la base sinon ce sera $m + j$.

Ceci est effectué jusqu'à ce que la base β_k soit complémentaire. Lorsque c'est le cas, on aura $z_i = 0, \forall i \notin \beta_k$ et $z_j = q^{\beta_k}_j, \forall j \in \beta_k$. Nous avons donc y et x. En les normalisant on obtient p_1 et p_2 qui sera un équilibre de Nash.

Exemple 44 (Application de l'algorithme)**.** *Donc nous avons :*

$$U = \begin{bmatrix} 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 2 & 5 \\ 0 & 0 & 0 & 3 & 3 \\ 1 & 0 & 4 & 0 & 0 \\ 0 & 2 & 3 & 0 & 0 \end{bmatrix} A = \begin{bmatrix} 0 & 0 & 0 & 0 & 6 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 5 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} q = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

La base $\beta_0 = \{6, 7, 8, 9, 10\}$ est faisable et complémentaire, c'est la solution externe. Ici $A^{\beta_0} = A$ et $q^{\beta_0} = q$ donc $T^{\beta_0} = \begin{bmatrix} -q^{\beta_0} & A^{\beta_0}\end{bmatrix} = \begin{bmatrix} -q & A\end{bmatrix}$

$$T^{\beta_0} = \begin{array}{c} \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{array} \begin{array}{c} \begin{array}{cccccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & y_1 & y_2 & x_3 & y_4 & y_5 \end{array} \\ \begin{bmatrix} -1 & 0 & 0 & 0 & 6 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 2 & 5 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Nous choisissons d'entrer l'indice 5 *dans la base, la base* β_1 *sera* 5*-presque complémentaire. L'indice qui sera retiré de la base* β_0 *sera celui où* $h = 1$ *car* $argmin\{1 : \frac{1}{6}, 2 : \frac{1}{5}, 3 : \frac{1}{3}\} = 1$ *donc* $s = 6$*. A l'étape suivante l'indice* 1 *devra être ajouté à la base car* $6 > m$ *donc* $6 - m = 1$*. Nous avons donc* $\beta_1 = \{5, 7, 8, 9, 10\}$ *La matrice de pivot* P^{β_1,β_0} *est :*

$$P^{\beta_1,\beta_0} = \begin{bmatrix} \frac{1}{6} & 0 & 0 & 0 & 0 \\ -\frac{5}{6} & 1 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Le tableau de la base β_1 *est :*

$$T^{\beta_1} = \begin{array}{c} \\ x_5 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{array} \begin{array}{c} \begin{array}{cccccccccc} & & & & & & & & & \end{array} \\ \begin{bmatrix} -\frac{1}{6} & 0 & 0 & 0 & 0 & 1 & \frac{1}{6} & 0 & 0 & 0 & 0 \\ -\frac{1}{6} & 0 & 0 & 0 & 2 & 0 & -\frac{5}{6} & 1 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 & 3 & 0 & -\frac{1}{2} & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

L'indice 1 *rentre dans la base à la place de* 9 $(h = 4)$ *donc le prochain indice à ajouter dans la base sera* 4*. La base* $\beta_2 = \{5, 7, 8, 1, 10\}$ *est toujours* 5*-presque complémentaire. La matrice pivot est la suivante :*

$$P^{\beta_2,\beta_1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

C'est une matrice identité donc $T^{\beta_2} = T^{\beta_1}$*.*

L'indice 4 *entre dans la base à la place de* 7 $(h = 2)$ *donc le prochain indice à ajouter sera* 2*. La base* $\beta_3 = \{5, 4, 8, 1, 10\}$ *est toujours* 5*-presque complémentaire. La matrice pivot est la suivante :*

$$P^{\beta_3,\beta_2} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & -\frac{3}{2} & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Donc :

$$T^{\beta_3} = \begin{array}{c} \\ x_5 \\ x_4 \\ y_3 \\ x_1 \\ y_5 \end{array} \begin{array}{c} \begin{array}{cccccccccc} & & & & & & & & & \end{array} \\ \begin{bmatrix} -\frac{1}{6} & 0 & 0 & 0 & 0 & 1 & \frac{1}{6} & 0 & 0 & 0 & 0 \\ -\frac{1}{12} & 0 & 0 & 0 & 1 & 0 & -\frac{5}{12} & \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{3}{4} & -\frac{3}{2} & 1 & 0 & 0 \\ -1 & 1 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Nous ajoutons l'indice 2 *dans la base à la place de* 10 *(*$h = 5$*). La base* $\beta_4 = \{5, 4, 8, 1, 2\}$ *est complémentaire, il n'y aura pas besoin de continuer. La matrice pivot est :*

$$P^{\beta_4,\beta_3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

Ce qui nous donne le tableau :

$$T^{\beta_4} = \begin{array}{c} \\ x_5 \\ x_4 \\ y_3 \\ x_1 \\ x_2 \end{array} \begin{bmatrix} -\frac{1}{6} & 0 & 0 & 0 & 0 & 1 & \frac{1}{6} & 0 & 0 & 0 & 0 \\ -\frac{1}{12} & 0 & 0 & 0 & 1 & 0 & -\frac{5}{12} & \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{3}{4} & -\frac{3}{2} & 1 & 0 & 0 \\ -1 & 1 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -\frac{1}{2} & 0 & 1 & \frac{3}{2} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

Avec la base $\beta_4 = \{5, 4, 8, 1, 2\}$ *et le tableau* T^{β_4}*, nous avons* $p'_1 = \begin{bmatrix} 1 & \frac{1}{2} & 0 \end{bmatrix}$ *et* $p'_2 = \begin{bmatrix} \frac{1}{12} & \frac{1}{6} \end{bmatrix}$ *qui après normalisation nous donnent* $p_1 = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} & 0 \end{bmatrix}$ *et* $p_2 = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} \end{bmatrix}$

Annexe B

Exécution de l'algorithme de Wilson

Nous reprenons l'exemple 35 du chapitre 5, dont les utilités et désutilités sont :

ω_1	ω_2	ω_3	u_1	u_2	u_3	a_1	a_2	a_3
0	0	0	6	0	4	2	8	4
0	0	1	7	3	7	1	5	1
0	1	0	0	7	4	8	1	3
0	1	1	5	4	0	3	4	8
1	0	0	1	6	3	7	2	5
1	0	1	4	5	2	4	3	6
1	1	0	2	1	6	6	7	2
1	1	1	3	2	1	5	6	7

(B.1)

Les polynômes $A_i^n(x^{-n})$ sont les suivants :

$$\begin{aligned}
A_0^1\left(x^2,x^3\right) &= 2x_0^2x_0^3 + x_0^2x_1^3 + 8x_1^2x_0^3 + 3x_1^2x_1^3 \\
A_1^1\left(x^2,x^3\right) &= 7x_0^2x_0^3 + 4x_0^2x_1^3 + 6x_1^2x_0^3 + 5x_1^2x_1^3 \\
A_0^2\left(x^1,x^3\right) &= 8x_0^1x_0^3 + 5x_0^1x_1^3 + 2x_1^1x_0^3 + 3x_1^1x_1^3 \\
A_1^2\left(x^1,x^3\right) &= x_0^1x_0^3 + 4x_0^1x_1^3 + 7x_1^1x_0^3 + 6x_1^1x_1^3 \\
A_0^3\left(x^1,x^2\right) &= 4x_0^1x_0^2 + 4x_0^1x_1^2 + 5x_1^1x_0^2 + 2x_1^1x_1^2 \\
A_1^3\left(x^1,x^2\right) &= x_0^1x_0^2 + 8x_0^1x_1^2 + 6x_1^1x_0^2 + 7x_1^1x_1^2
\end{aligned}$$

Lorsque nous choisissons la stratégie jointe $\omega^0 = (0,0,0)$, nous avons pour chacun des niveaux inférieurs à 3 les polynômes suivantes. Pour le niveau 2 (où l'on fixe $x_0^3 = 1$ et $x_1^3 = 0$) :

$$\begin{aligned}
A_0^1\left(x^2\right) &= 2x_0^2 + 8x_1^2 \\
A_1^1\left(x^2\right) &= 7x_0^2 + 6x_1^2 \\
A_0^2\left(x^1\right) &= 8x_0^1 + 2x_1^1 \\
A_1^2\left(x^1\right) &= x_0^1 + 7x_1^1
\end{aligned}$$

Pour le niveau 1 (on fixe de plus $x_0^2 = 1$ et $x_1^2 = 0$) :

$$\begin{aligned}
A_0^1 &= 2 \\
A_1^1 &= 7
\end{aligned}$$

Le parcours complet effectué avec la méthode de parcours de chemin est montré dans la figure B.1. Notez que le premier joueur/niveau est numéroté 0 et non pas 1 dans les figures. Le niveau 2 est numéroté 1 et le niveau 3 est numéroté 2.

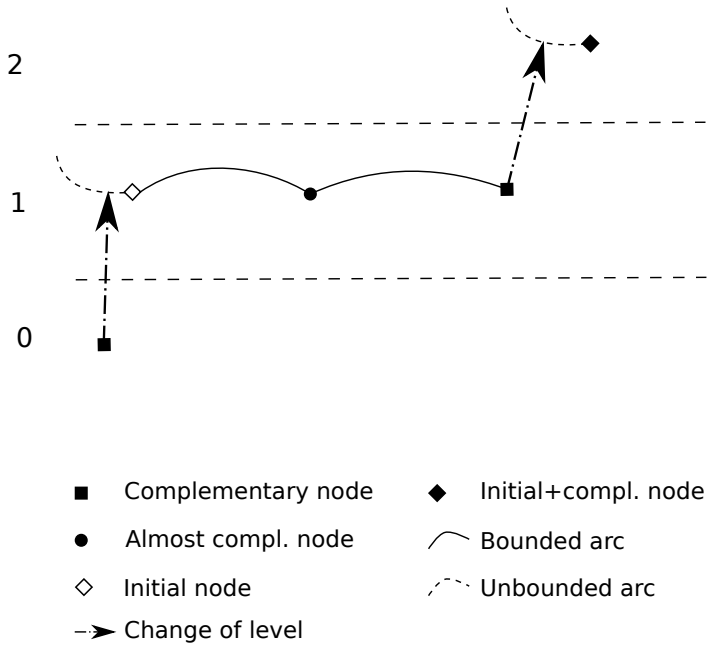


FIGURE B.1 – Parcours complet du jeu de l'exemple 35 pour $\omega^0 = (0, 0, 0)$

Nous commençons au premier niveau. A ce niveau les "polynômes" du premier joueur sont des constantes. Il est facile de calculer le noeud complémentaire à partir de ces constantes. En réutilisant le système suivant :

$$\left\{\begin{array}{rcl} x_i^1 & \geqslant & 0, \forall i \in S_1 \\ x_i^1 \cdot \left(\frac{a^1_{(i,\omega^0_{-1})}}{\min_{j\in\Omega_1} a^1_{(j,\omega^0_{-1})}} - 1\right) & = & 0, \forall i \in S_1 \quad (\mathcal{S}_1) \\ \sum_{i\in\Omega_1} x_i^1 & = & 1 \end{array}\right.$$

Nous "simplifions" les équations :

$$\begin{array}{rcl} A_0^{1,1} & = & \frac{2}{2} = 1 \\ A_1^{1,1} & = & \frac{7}{2} \end{array} \tag{B.2}$$

Le PCP à résoudre sera :

$$\left\{\begin{array}{rcl} x_0^1, x_1^1 & \geqslant & 0 \\ x_0^1 \cdot (0) & = & 0 \\ x_1^1 \cdot \left(\frac{5}{2}\right) & = & 0 \\ \sum_{i\in\Omega_1} x_i^1 & = & 1 \end{array}\right. \quad (\mathcal{S}^1)$$

La solution de ce système est $x_0^1 = 1$ et $x_1^1 = 0$, ce qui correspond à la paire (Z, W) qui suit :

— $Z = \{(1, 1)\}$
— $W = \{(1, 0)\}$

Nous sommes actuellement sur le premier noeud (au premier niveau), comme montré dans la figure B.2. Ce noeud est complémentaire étant donné que $Z \cup W = I_1$ et que $Z \cap W = \varnothing$.

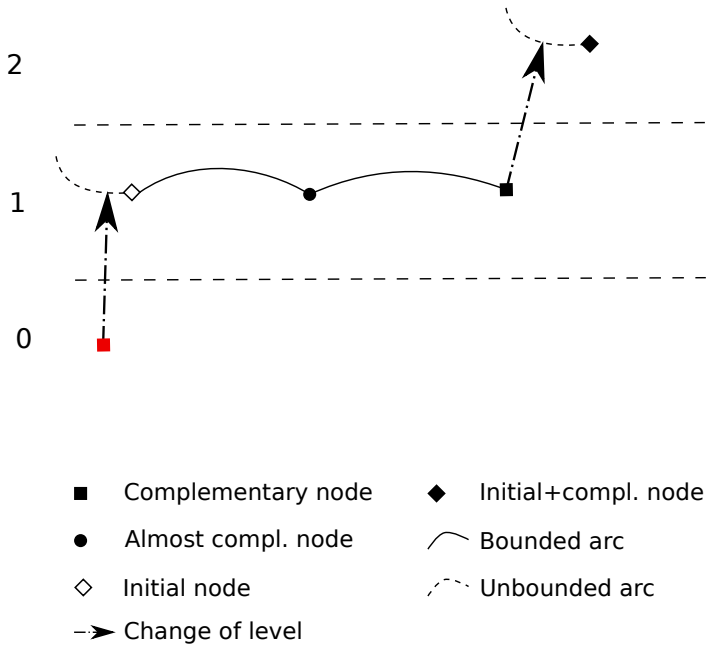


FIGURE B.2 – Noeud complémentaire au niveau 0.

Le noeud étant complémentaire, nous passons au niveau supérieur, avec deux joueurs. Les polynômes sont les suivants :

$$\begin{aligned} A_0^1\left(x^2\right) &= 2x_0^2 + 8x_1^2 \\ A_1^1\left(x^2\right) &= 7x_0^2 + 6x_1^2 \\ A_0^2\left(x^1\right) &= 8x_0^1 + 2x_1^1 \\ A_1^2\left(x^1\right) &= x_0^1 + 7x_1^1 \end{aligned}$$

Lorsque nous passons au niveau supérieur, avant de trouver le noeud initial associé au noeud complémentaire selon ω^0, nous passons par l'arc $\gamma^2(Z, W)$ où :

— $Z = \{(1,1),(2,1)\}$
— $W = \{(1,0)\}$

Nous avons $(2,1) \in Z$ car $\omega_2^0 = 0$ (le joueur 2 joue l'action 0 dans ω^0). Nous sommes sur l'arc représenté dans la figure B.3. Il correspond au système d'équations suivant :

$$\left\{ \begin{array}{rcl} x_1^1 & = & 0 \\ x_1^2 & = & 0 \\ A_0^1(x) - 1 & = & 2x_0^2 + 8x_1^2 - 1 = 0 \end{array} \right. \qquad \left(\mathcal{S}_2^{Z,W}\right)$$

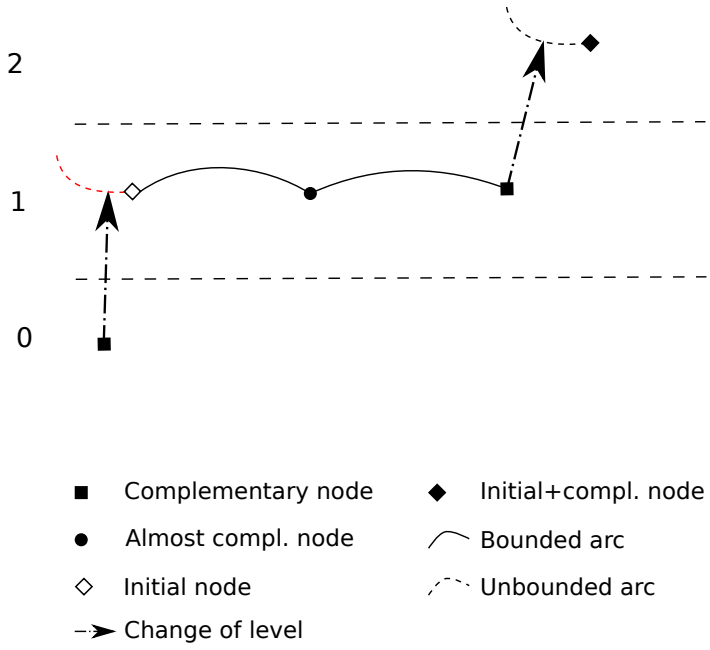


FIGURE B.3 – Arc non borné au niveau 1

Pour obtenir un noeud presque complémentaire il faut ajouter un couple $(2, i)$ où $i \in \Omega_2$ a l'ensemble W (sinon le noeud ne sera pas $(2,0)$-presque complémentaire), il y a deux possibilité, $(2,0)$ ou $(2,1)$. Nous devons donc vérifier quel est le couple à ajouter qui donne un système avec une solution.

Pour $W' = \{(1,0),(2,0)\}$

$$\begin{aligned} x_1^1 &= 0 \\ x_1^2 &= 0 \\ A_0^1(x) - 1 &= 2x_0^2 + 8x_1^2 - 1 = 0 \\ A_0^2(x) - 1 &= 8x_0^1 + 2x_1^1 - 1 = 0 \end{aligned}$$

Pour $W' = \{(1,0),(2,1)\}$

$$\begin{aligned} x_1^1 &= 0 \\ x_1^2 &= 0 \\ A_0^1(x) - 1 &= 2x_0^2 + 8x_1^2 - 1 = 0 \\ A_1^2(x) - 1 &= x_0^1 + 7x_1^1 - 1 = 0 \end{aligned}$$

Ici la solution du système avec $W' = \{(1,0),(2,0)\}$ correspond à une solution qui n'est pas faisable. Les coordonnées calculées sont $\{x_0^1 = 0.125, x_1^1 = 0, x_0^2 = 0.5, x_1^2 = 0\}$ ce qui donne donc $A_1^{2,2} = x_0^1 + 7x_1^1 = 0.125 < 1$ qui fait que la solution n'est pas faisable.

Les coordonnées de l'autre noeud ($W' = \{(1,0),(2,1)\}$) sont $\{x_0^1 = 1, x_1^1 = 0, x_0^2 = 0.5, x_1^2 = 0\}$ qui est faisable (tous les polynômes sont positifs), c'est donc le noeud initial obtenu après avoir effectué la montée.

Le prochain noeud $\rho^2(Z', W')$ est :

— $Z' = \{(1,1),(2,1)\}$
— $W' = \{(1,0),(2,1)\}$

Il a les "coordonnées" $x = \{x_0^1 = 1, x_1^1 = 0, x_0^2 = 0.5, x_1^2 = 0\}$, il est presque complémentaire nous devons donc continuer le parcours pour ce niveau.

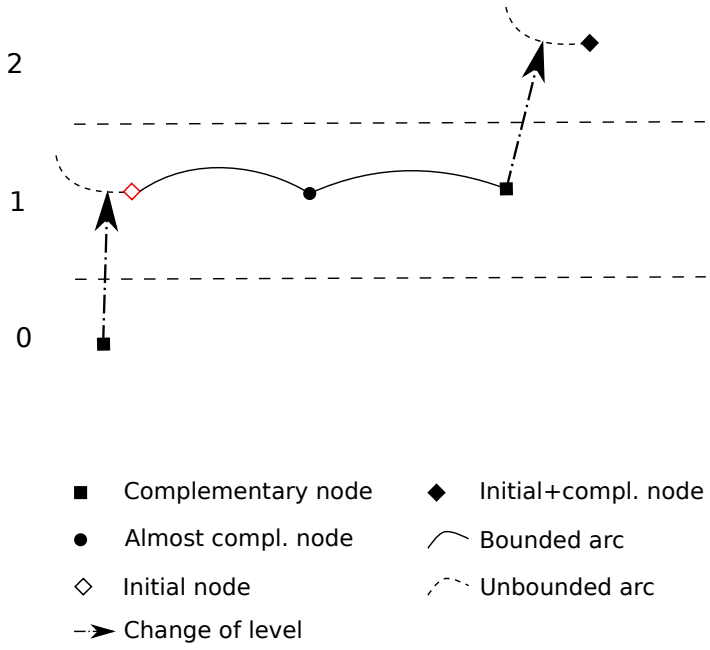


FIGURE B.4 – Noeud initial pour deux joueurs.

Compte tenu que nous venons d'ajouter la paire $(2,1)$ dans W', pour maintenir la presque complémentarité, pour le prochain arc que nous allons parcourir, nous devons retirer le couple correspondant $(2,1)$ de Z'. Ce qui fait que nous avons un arc $\gamma^2(Z,W)$:

— $Z = \{(1,1)\}$

— $W = \{(1,0),(2,1)\}$

Si nous venions d'ajouter la paire$(2,1)$ dans Z', alors nous l'aurions retirée de W'. Nous sommes "sur" l'arc présenté dans la figure B.5

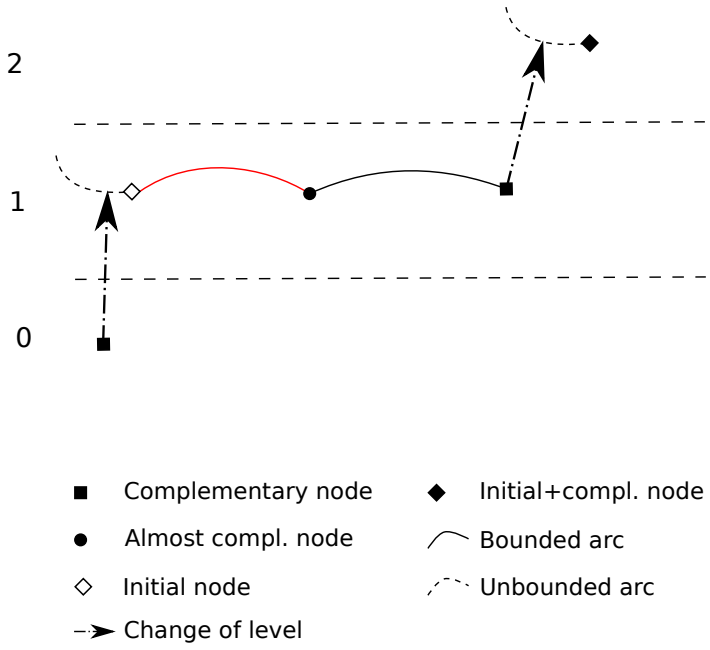


FIGURE B.5 – Arc au niveau à deux joueurs

Nous allons maintenant parcourir tous les paires de I_2 qui permettent d'avoir un nouveau noeud presque complémentaire et vérifier sa faisabilité. Ici nous avons $I_2 \backslash Z' =$

$\{(1,0),(2,0)\}$ que nous pouvons ajouter dans Z, cependant si nous ajoutons $(1,0)$, nous nous retrouvons dans une situation impossible où un joueur ne joue pas, donc seule la paire $(2,0)$ est envisageable comme ajout dans Z. Nous avons donc le noeud suivant qui est à envisager : Ajout de $(2,0)$ dans Z :

— $Z_1'' = \{(1,1),(2,0)\}$
— $W_1'' = \{(1,0),(2,1)\}$

Nous pouvons ajouter dans W l'une des paires suivantes $I_2 \backslash W' = \{(1,1),(2,0)\}$.

Donc il y a deux autres noeuds possibles :

Ajout de $(1,1)$ dans W :
— $Z_2'' = \{(1,1)\}$
— $W_2'' = \{(1,0),(2,1),(1,1)\}$

Ajout de $(2,0)$ dans W :
— $Z_3'' = \{(1,1)\}$
— $W_3'' = \{(1,0),(2,1),(2,0)\}$

Pour (Z_1'', W_1''), nous avons une variété $\mathcal{V}\left(\mathcal{S}_k^{Z_1'',W_1''}\right)$ qui sera de dimension 0 qui comprend une solution $x = \{x_0^1 = 1, x_1^1 = 0, x_0^2 = 0, x_1^2 = 0.125\}$ qui est cependant non faisable car $A_1^{1,2} = 7x_0^2 + 6x_1^2 = 0.75 < 1$.

Pour (Z_2'', W_2''), nous avons une variété $\mathcal{V}\left(\mathcal{S}_k^{Z_2'',W_2''}\right)$ qui sera de dimension 0, qui comprend une solution $x = \{x_0^1 = 1, x_1^1 = 0, x_0^2 = \frac{1}{22}, x_1^2 = \frac{5}{44}\}$ qui est faisable. Les polynômes ont les valeurs suivantes :

$$\begin{aligned} A_0^{1,2}\left(x^2\right) &= 2x_0^2 + 8x_1^2 = 1 \\ A_1^{1,2}\left(x^2\right) &= 7x_0^2 + 6x_1^2 = 1 \\ A_0^{2,2}\left(x^1\right) &= 8x_0^1 + 2x_1^1 = 8 \\ A_1^{2,2}\left(x^1\right) &= x_0^1 + 7x_1^1 = 1 \end{aligned}$$

Ce noeud est faisable, nous avons donc trouvé le prochain noeud.

Nous allons quand même observer le troisième noeud possible : Pour (Z_3'', W_3''), nous avons une variété $\mathcal{V}\left(\mathcal{S}_k^{Z_3'',W_3''}\right)$ qui vide, cela signifie qu'il n'y a pas de solution.

Notre noeud est le noeud $\rho^2(Z', W')$ où :

— $Z' = \{(1,1)\}$
— $W' = \{(1,0),(2,1),(1,1)\}$

Il est toujours presque complémentaire, nous devons continuer le parcours.

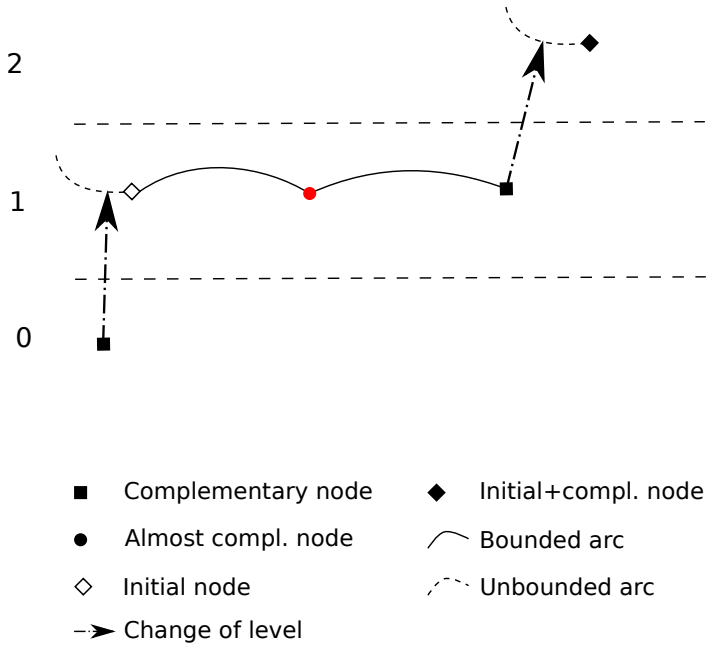


FIGURE B.6 – Noeud presque complémentaire au niveau à deux joueurs

Nous venons d'ajouter $(1, 1)$ dans W, nous retirons donc ce même couple de Z pour obtenir l'arc $\gamma^2(Z, W)$ qui suit :

— $Z' = \{\}$

— $W' = \{(1, 0), (2, 1), (1, 1)\}$

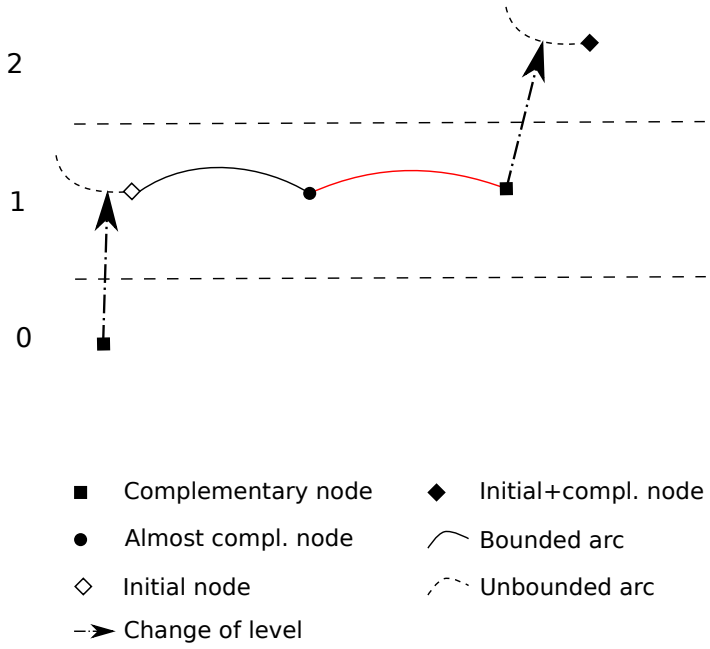


FIGURE B.7 – Arc au niveau au niveau à deux joueurs.

Nous pouvons ajouter 3 paires dans Z, $\{(1, 0), (2, 0), (2, 1)\}$ ou bien une paire dans W, $(2, 0)$. Parmi ces cas de figure possibles le seul permettant d'avoir un noeud presque complémentaire et faisable est l'ajout de $(2, 0)$ dans W, qui nous donne le noeud $\rho^2(Z'', W'')$ suivant

— $Z'' = \{\}$

— $W'' = \{(1,0),(2,1),(1,1),(2,0)\}$

Ce noeud aura les coordonnées suivantes : $x = \{x_0^1 = \frac{5}{54}, x_1^1 = \frac{7}{54}, x_0^2 = \frac{1}{22}, x_1^2 = \frac{5}{44}\}$

$$\begin{aligned} A_0^{1,2}\left(x^2\right) &= 2x_0^2 + 8x_1^2 = 1 \\ A_1^{1,2}\left(x^2\right) &= 7x_0^2 + 6x_1^2 = 1 \\ A_0^{2,2}\left(x^1\right) &= 8x_0^1 + 2x_1^1 = 1 \\ A_1^{2,2}\left(x^1\right) &= x_0^1 + 7x_1^1 = 1 \end{aligned}$$

Le noeud est faisable et complémentaire, par conséquent nous avons trouvé une solution pour le niveau avec deux joueurs.

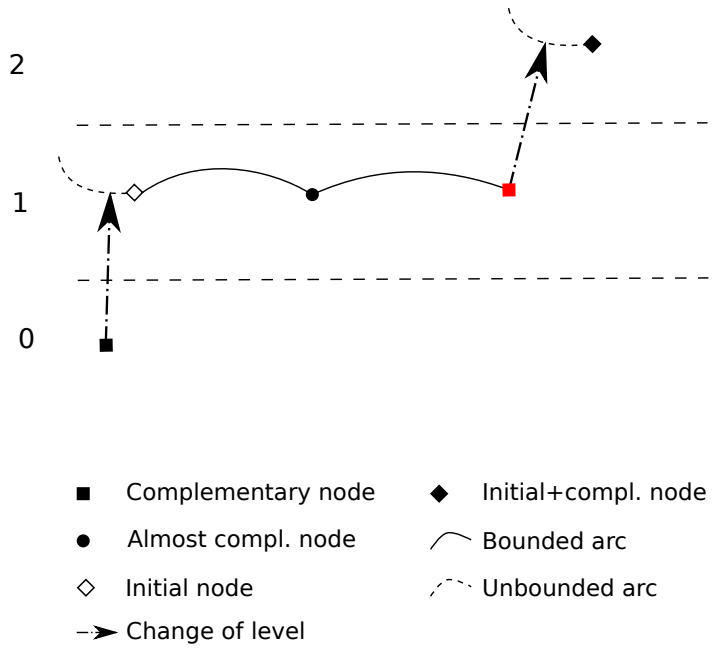


FIGURE B.8 – Noeud complémentaire au niveau à deux joueurs.

Nous devons donc passer du niveau avec deux joueurs (niveau 1 dans les figures) au niveau avec trois joueurs (niveau 2 dans les figures). En tenant compte de ω^0, l'arc nous permettant de calculer le noeud initial sera l'arc $\gamma^3(Z,W)$ où

— $Z = \{(3,1)\}$
— $W = \{(1,0),(2,1),(1,1),(2,0)\}$

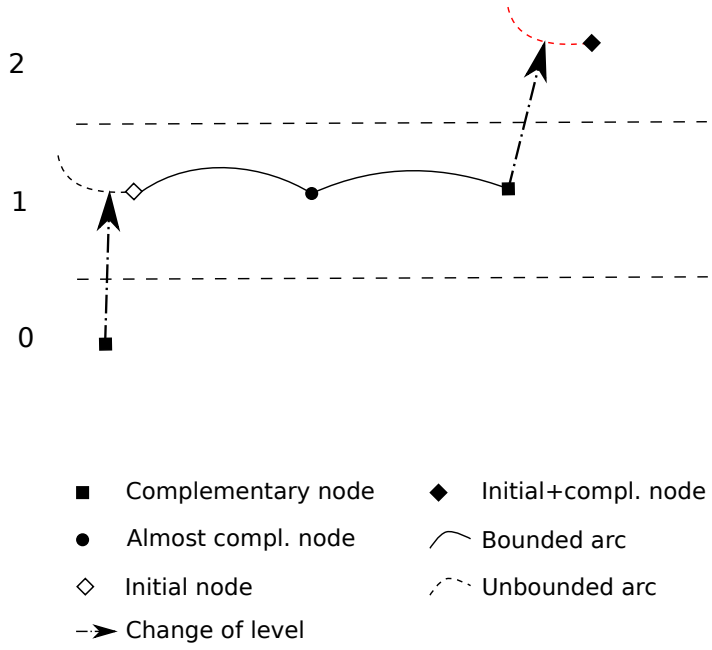


FIGURE B.9 – Arc non borné au niveau à trois joueurs.

Il y a deux paires pouvant être ajoutées dans W pour obtenir un noeud $(3,0)$ et $(3,1)$. C'est l'ajout de $(3,0)$ dans W qui nous amène a un noeud faisable. Le noeud initial $\rho^3(Z,W)$ est :

— $Z = \{(3,1)\}$

— $W = \{(1,0),(2,1),(1,1),(2,0),(3,0)\}$

qui a les coordonnées $x = \{x_0^1 : 0.2697, x_1^1 : 0.3776, x_0^2 : 0.1324, x_1^2 : 0.3310, x_0^3 : 0.3432, x_1^3 : 0\}$.

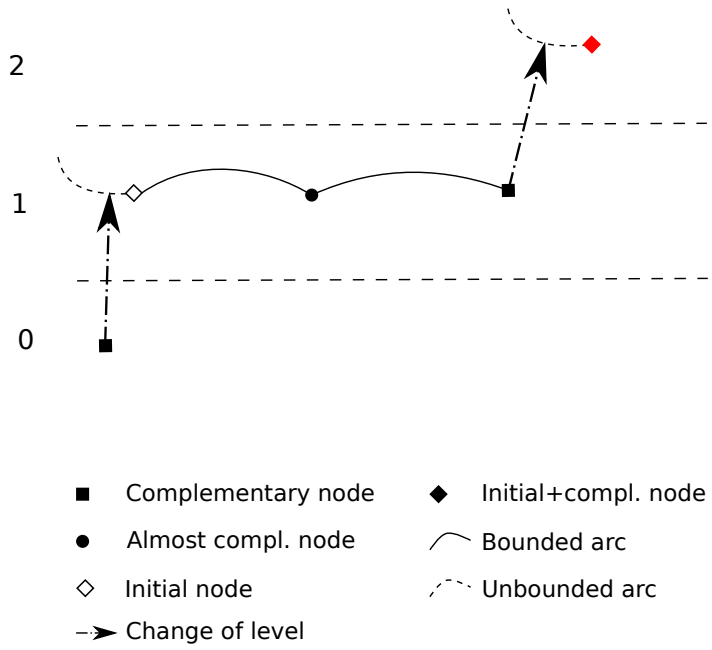


FIGURE B.10 – Noeud complémentaire au niveau maximum.

Le noeud étant complémentaire, nous avons trouvé un équilibre de Nash (mixte). En normalisant les coordonnées nous avons la distribution de probabilité suivante : $\xi = \{\xi_0 = \{\frac{5}{12}, \frac{7}{12}\}, \xi_1 = \{\frac{2}{7}, \frac{5}{7}\}, \xi_2 = \{1,0\}\}$.

Annexe C

Ajout de variables auxiliaires dans le PCP

Lorsque nous avons un jeu hypergraphique, nous proposons aussi une écriture du PCP différente de celle présentée dans le chapitre 5. Ce modèle de PCP inclut des variables additionnelles obtenues a partir de la structure de l'hypergraphe. Cela permet d'obtenir des polynômes de degré moins élevé. L'idée est que le calcul de Gröbner est théoriquement simplement exponentiel en le nombre de variables et doublement exponentiel en le degré maximum des polynômes du système à résoudre. Cette modélisation pouvait permettre un calcul plus rapide des solutions d'un PCP (puisque résoudre un PCP nécessite de résoudre plusieurs systèmes d'équations polynomiales).

Définition 51 (Variables auxiliaires d'un PCP)**.** *En plus des variables* $\{x_i^n\}$*, considérons un ensemble de variables supplémentaire,* $\left\{y_g^n\right\}_{g=1..G,n\in S}$.

Ces variables sont définies par :

$$y_g^n = \prod_{\nu=1}^{n} \alpha_g^\nu, \ \text{où } \alpha_g^\nu = 1 \ si \ \nu \in S_g \ et \ \alpha_\nu^g = \sum_{\omega_\nu \in \Omega_\nu} x_{\omega_\nu}^\nu \ si \ \nu \in S \backslash S_g.$$

Notez qu'il y a exactement $E \times N$ variables additionnelles y_g^n et leur valeurs sont définies par des équations de degré 1 quand $n \in S_g$ et de degré 2 lorsque $n \in S\backslash S_g$. Cela donne l'expression suivante pour A_i^n :

$$A_i^n\left(x^{-n}, \left\{y_g^N\right\}_{g=1,..,G}\right) = \sum_{g,n\in S_g} y_g^N \sum_{\substack{\omega_{S_g} \\ \omega_n = i}} a_{\omega_{S_g}}^{g,n} \prod_{\nu \in S_g \backslash \{n\}} x_{\omega_\nu}^\nu. \tag{C.1}$$

Remarquez que les polynômes $A_i^n\left(x^{-n}, \left\{y_g^N\right\}_{g=1,..,G}\right)$ on maintenant un degré au plus de $max_{g=1,..,G}|P_g|$ et que leur nombre de monômes est de l'ordre de la taille du jeu hypergraphique (du nombre de valeurs des matrices des jeux locaux).

Mon algorithme peut maintenant être étendu pour gérer ces nouvelles expressions avec des variables supplémentaires. La seule modification dans l'algorithme est la définition des sous systèmes polynomiaux qui maintenant doivent comprendre les variables y_g^n. La définition du systèmes polynomial Les systéme polynomial $\overline{\mathcal{S}}_k^{Z,W}$ sont définis, pour les paires $Z, W \subseteq I_k$

$$\begin{cases} & x \in \mathcal{D}, \\ x^n_{\omega^0_n} = 1 \text{ et } x^n_i = 0, & \forall n > k, \forall i \neq \omega^0_n \\ x^n_i = 0, & \forall (n,i) \in Z, \\ y^1_g = \alpha^1_g, & \forall g = 1, \ldots, G \\ y^n_g = y^{n-1}_g \times \alpha^n_g, & \forall g = 1..G, \forall n = 2..k. \\ A^{n,k}_i \left(x^{\{1..k\}\setminus\{n\}}, \left\{ y^k_g \right\}_{g=1,..,G} \right) = 1, & \forall (n,i) \in W, \end{cases}$$

Cette généralisation aux jeux hypergraphiques conduit à une réduction du nombre de termes dans les polynômes du système et à un degré borné par le nombre de joueurs du plus grand sous jeu. Ceci est important pour le calcul des bases de Gröbner. Le *Ideal Membership Problem*, qui est l'étape principale du problème de calcul de base de Gröbner est connu pour être ExpSpace-complet. Cependant, lorsque l'idéal est de dimension 0 (ce qui est le cas quand $\left(\overline{\mathcal{S}}^{Z,W}_k\right)$ a un nombre fini de solutions), il est soluble en temps exponentiel [Dickenstein *et al.*, 1991]. De plus, être capable d'avoir une borne supérieur sur le degré des polynômes dans le PCP à un impact positif sur la complexité de l'algorithme :

Proposition 29 (Complexité de l'algorithme de parcours de chemin)**.** *La complexité en temps de l'algorithme de parcours de chemin pour un jeu graphique/hypergraphique non degénéré est exponentiel en $|I_N|$ (le nombre total de variables) et doublement exponentiel en le nombre maximum de joueurs dans tout jeu local.*

Preuve (Proposition 29)**.** *En effet, dans le pire cas, la complexité en temps du meilleur algorithme de calcul de base de Grobner est doubblement exponentielle en le degré maximum des polynômes (voir [Mayr, 1997]). Le nombre de traversées d'arc est borné par le nombre de noeuds presque complémentaire pour tout niveau. Par conséquent, le nombre de traversées d'arc au niveau k est borné par :*

$$\#Nodes(k) = |\left\{(Z,W), |Z| + |W| = |I_k|, |Z \cap W| \leqslant 1\right\}|$$

Notez que, pour tout jeu hypergraphique, dont les jeux sous forme normale : $\#Nodes(k) \leqslant |I_k| 2^{|I_k|}$. En effet, les noeuds potentiels peuvent être construits en choisissant arbitrairement $Z \subseteq I_k$ et il y a au plus $|I_k|$ ensembles W potentiels amenant à un noeuds ω^0-presque complémentaires au niveau k. Donc le nombre total de noeud presque complémentaire pour tout niveau est $O\left(N|I_N|2^{|I_N|}\right)$. Puisque chaque traversée d'arc demande au plus $O(|I_N|)$ calculs de base de Grobner, nous obtenons le résultat[1]

1. Cette borne de la complexité est théorique. En pratique, la longueur des chemin est généralement courte.