# Recherche heuristique pour jeux stochastiques (à somme nulle)

Olivier Buffet, Jilles S Dibangoye, Abdallah Saffidine, Vincent Thomas

# Recherche heuristique pour jeux stochastiques (à somme nulle)

Olivier buffet[1] Jilles Dibangoye[2] Abdallah Saffidine[3] Vincent Thomas[1]

[1] Université de Lorraine, INRIA, CNRS, LORIA, F-54000 Nancy
[2] Université de Lyon, INSA Lyon and Inria, CITI, F-69000 Lyon
[3] Australian National University, Canberra, Australie

prenom.nom@(inria.fr|anu.edu.au)

## Résumé

*Dans divers types de problèmes, par exemple de prise de décision séquentielle, les algorithmes de recherche heuristique permettent d'exploiter la connaissance d'une situation initiale et d'une heuristique admissible pour rechercher efficacement une solution optimale. De tels algorithmes existent y compris en cas de dynamique incertaine, d'observabilité partielle, de critères multiples, ou d'agents multiples collaborant. Nous proposons ici un algorithme de recherche heuristique pour jeux stochastiques à deux joueurs et à somme nulle, et avec critère décompté, algorithme reposant sur HSVI—donc sur la génération de trajectoires. Nous démontrons que, chaque joueur agissant de manière optimiste, et en employant des initialisations heuristiques simples, l'algorithme obtenu converge vers une solution $\epsilon$-optimale en temps fini.*

## Mots Clef

Recherche heuristique, jeux stochastiques, HSVI.

## Abstract

*In various types of problems, such as sequential decision-making, heuristic search algorithms allow exploiting the knowledge of the initial situation and of an admissible heuristic to efficiently search for an optimal solution. Such algorithms exist including in case of uncertain dynamics, of partial observability, of multiple criteria, or of multiple collaborating agents. Here we propose a heuristic search algorithm for two-player zero-sum stochastic games with discounted critrion. This algorithm relies on HSVI—hence on generating trajectories. We demonstrate that, each player acting in an optimistic manner, and employing simple heuristic initializations, the resulting algorithm converges in finite time to an $\epsilon$-optimal solution.*

## Keywords

Heuristic Search, Stochastic Games, HSVI.

# 1 Introduction

Les techniques de recherche heuristiques ont été introduites pour résoudre des problèmes d'optimisation à un seul agent et un seul critère dans des cadres déterministes (souvent, mais pas seulement, à des fins de planification) avec des algorithmes tels que A*, (L)RTA*, ou RBFS [20, 7]. Elles ont été étendues pour résoudre des problèmes avec des dynamiques stochastiques dans les cadres MDP et POMDP comme dans (L)RTDP, (L)AO*, ou HSVI [5, 9, 25], et même dans des cadres multi-agents collaboratifs dans MAA* ou FB-HSVI [27, 8]. Dans tous ces cas, un seul critère est considéré, de sorte que guider l'exploration en étant optimiste reste une chose assez triviale à faire, même en planifiant pour plusieurs agents. Quand plusieurs critères sont considérés pour un seul agent, ou éventuellement des agents collaborants, ce guidage peut être adapté aussi bien quand on agrège ces critères que quand on cherche un front de Pareto [18]. À notre connaissance, la recherche heuristique n'a pas été appliquée aux cadres multi-agents non-collaboratifs avec jeu simultané, c'est-à-dire dans des jeux (stochastiques) où les joueurs ont leur propres objectifs.

Cet article se concentre sur les jeux stochastiques décomptés à deux joueurs et somme nulle (zs-SG) avec information parfaite (à l'exception du mouvement courant de l'adversaire), lesquels ont été principalement abordés en utilisant des algorithmes tels que la programmation dynamique exacte [17] (de manière similaire aux algorithmes d'itération sur la valeur ou la politique pour les MDP), l'apprentissage par renforcement [12], ou la programmation dynamique approchée [10, 16]. Le critère de la récompense totale, commun dans les jeux de plateau, requière que toutes les trajectoires finissent dans des états terminaux, et a été abordé avec l'apprentissage par renforcement [12], l'induction arrière [21, 6] (c'est-à-dire en applicant la programmation dynamique sur les états accessibles quand on élague les branches non pertinentes) et MCTS avec mouvements simultanés (Simultaneous Move MCTS) [6]. Pour d'autres critères, se référer à [26]. Les seuls algorithmes exacts (ou plus précisément à erreur bornée) listés ci-dessus sont ceux reposant sur la programmation dynamique. Parmi ceux-ci, l'induction arrière, parce qu'elle élague typiquement des branches en exploitant des estimations de valeurs conservatives, peut être vue comme une forme de recherche heu-

ristique.
Dans nos zs-SG décomptés, nous faisons l'hypothèse qu'un état initial $s_0$ est donné, et que des majorants et minorants initiaux ($U$ et $L$) de la fonction de valeur optimale (qui serviront d'heuristiques admissibles) sont disponibles. Sous ces hypothèses, nous considérons des schémas algorithmiques apprenant en générant des trajectoires (à la LRTA*, (L)RTDP ou HSVI). Nous montrons que, dans ce cadre, les trajectoires peuvent être générées en laissant les deux joueurs agit de manière optimiste (l'un de manière gourmande par rapport au majorant $U(s,a)$ de la valeur optimale (en équilibre de Nash), l'autre par rapport au minorant $L(s,a)$) tout en assurant que ces encadrements vont converger vers l'optimum. Utiliser majorant et minorant amène naturellement vers des algorithmes ressemblant à Bounded-RTDP et HSVI [13, 25].
L'article est organisé comme suit. Il fournit d'abord un contexte sur les jeux en forme normale, les processus de décision markoviens (MDP), et les jeux stochastiques en section 2. Ensuite, quelques résultats préliminaires sur les jeux en forme normale sont présentés en section 3 quand on raisonne sur des jeux minorants ou majorants. Cela conduit à proposer les ingrédients clefs pour dériver une variante de l'algorithme HSVI (avec des preuves de convergence) en section 4. Finalement, la section 5 conclut avec une discussion sur de possibles travaux futurs.

# 2 Contexte

Dans ce qui suit, les exposants indiquent les joueurs, et les indices indiquent le pas de temps ou l'itération.

## 2.1 Jeux en forme normale/matricielle

Un jeu à deux joueurs en *forme normale* est défini par un tuple $\Gamma \stackrel{\text{def}}{=} \langle \mathcal{A}^1, \mathcal{A}^2, v^1, v^2 \rangle$, où $\mathcal{A}^i$ est l'ensemble fini des stratégies pures du joueur $i$ ($i \in \{1,2\}$), et $v^i : \mathcal{A}^1 \times \mathcal{A}^2 \mapsto \mathbb{R}$ est la fonction de gain du joueur $i$. L'objectif est alors, pour chaque joueur, de maximiser son espérance de gain. À moins qu'ils aient des fonctions de gain identiques, il s'agit de trouver non pas l'optimum d'une fonction, mais u *équilibre*. Un concept solution classique est celui d'*équilibre de Nash* (NE) [15], défini comme une paire $(d^1, d^2) \in \Delta(\mathcal{A}^1) \times \Delta(\mathcal{A}^2)$ de stratégies mixtes (c'est-à-dire des distributions de probabilités sur les stratégies pures) telle qu'aucun joueur n'a intérêt de dévier de cette solution seul :

$$\forall d'^1 \in \Delta(\mathcal{A}^1), \quad v^1(d'^1, d^2) \leq v^1(d^1, d^2),$$
$$\forall d'^2 \in \Delta(\mathcal{A}^2), \quad v^2(d^1, d'^2) \leq v^2(d^1, d^2).$$

Dans un jeu à somme nulle, $v^1 + v^2 = 0$ (c'est-à-dire que le gain d'un joueur est la perte de l'autre), et nous noterons typiquement $v = v^1 = -v^2$. Aussi, comme démontré par von Neumann [28], un tel jeu $\Gamma$ a une unique valeur d'équilibre de Nash $\text{NEV}(\Gamma)$ égale à la fois aux valeurs minimax et maximin. En conséquence, les stratégies formant une stratégie en équilibre de Nash (joint) (NES) peuvent être trouvées en résolvant (typiquement en tant que programmes linéaires) :

$$d^{NES,1} = \arg\max_{d^1 \in \Delta(\mathcal{A}^1)} \min_{a^2 \in \mathcal{A}^2} v(d^1, a^2),$$
$$d^{NES,2} = \arg\min_{d^2 \in \Delta(\mathcal{A}^2)} \max_{a^1 \in \mathcal{A}^1} v(a^1, d^2).$$

## 2.2 Jeux stochastiques

Un jeu stochastique (ou de Markov) décompté à deux joueurs et somme nulle (zs-SG) [22, 23, 17] est spécifié par un tuple $\langle \mathcal{S}, \mathcal{A}^1, \mathcal{A}^2, P, r, \gamma, s_0 \rangle$ où $\mathcal{S}$ est un ensemble fini d'états, $\mathcal{A}^1$ et $\mathcal{A}^2$ sont des ensembles finis d'actions (une par joueur), $P_{a^1,a^2}(s'|s)$ est la probabilité de transiter de l'état $s$ à $s'$ quand les actions $a^1$ et $a^2$ sont effectuées ; $r(s,a^1,a^2)$ est une fonction de récompense (scalaire) ; $\gamma \in [0,1)$ est un facteur d'atténuation ; et $s_0$ est l'état initial. L'objectif du joueur 1 est de maximiser l'espérance de la somme atténuée des récompenses $E[\sum_t \gamma^t R_t | s_0]$ alors que l'objectif du joueur 2 est de minimiser cette quantité.
Par commodité, on notera :

- $\pi^i : \mathcal{S} \to \Delta(\mathcal{A}^i)$ une stratégie stochastique pour le joueur $i$ ;
- $\pi = (\pi^1, \pi^2)$ une stratégie jointe (ou une paire de stratégies) pour les deux joueurs ;
- $\boldsymbol{r}(\pi)$ le vecteur des récompenses immédiates espérées pour la stratégie (jointe) $\pi$ : $\boldsymbol{r}(\pi)(s) = \sum_{a^1,a^2} \pi^1(a^1|s)\pi^2(a^2|s) r(s,a^1,a^2)$ ;
- $P(\pi)$ la matrice de transition induite par la stratégie $\pi$.

Un tel jeu peut être ré-écrit en forme normale et résolu en cherchant alors un équilibre de Nash (une stratégie mixte). Toutefois, un concept solution plus satisfaisant est ici celui de *stratégie équilibre de Nash parfaite en sous-jeux* (stratégies SGPNE), c'est-à-dire de solutions consistant en un équilibre de Nash par état (rencontré). Définissons, pour tout $\boldsymbol{v}$, le *jeu de matrice de Shapley* $\Gamma^s(\boldsymbol{v}) \stackrel{\text{def}}{=} [r(s,\cdot,\cdot) + \gamma \sum_{s'} P_{\cdot,\cdot}(s'|s)\boldsymbol{v}(s')]$ pour tout $s$ (c'est-à-dire, une valeur par paire d'action $(a^1,a^2)$). Alors, *l'opérateur d'optimalité de Shapley* $\mathcal{H} : \boldsymbol{v} \mapsto \text{NEV}(\Gamma^{\cdot}(\boldsymbol{v}))$ est une fonction contractante, et son unique point fixe est la fonction de valeur des stratégies SGPNE, notée $\text{NEV}^*$. *L'algorithme de Shapley* pour résoudre les zs-SG, analogue à *l'itération sur la valeur (VI)* pour les MDP, consiste en l'application itérative de cet opérateur jusqu'à $\epsilon$-convergence, et alors, quand on est dans un état $s$, d'agir selon la NES du jeu de Matrice de Shapley induit en $s$. D'autres algorithmes exacts ont été proposés qui gèrent aussi des problèmes sans états initiaux connus [17].

## 2.3 MDP et HSVI

Les processus de décision markoviens (MDP) [2, 3] peuvent être vus comme des jeux stochastiques dans lesquels un joueur (sans perte de généralité, le joueur minimiseur 2) a une seule action disponible, et donc aucune décision à prendre. Le problème est alors de trouver une stratégie (aussi appelée *politique*) pour le joueur

maximiseur 1. Ici la NEV devient la *fonction de valeur optimale* $V^*$, et agir de manière gloutonne par rapport à cette fonction de valeur induit une politique optimale, de sorte qu'il existe toujours une politique optimale déterministe. Par commodité, pour une fonction de valeur $V$, nous introduisons la $Q^V$-function correspondante $Q^V(s,a) \stackrel{\text{def}}{=} r(s,a) + \gamma \sum_{s'} P_a(s'|s)V(s')$.

Bounded RTDP [13] et HSVI [25] sont deux algorithmes résolvant des MDP en reposant sur (i) un état initial $s_0$ pour se focaliser sur les parties les plus pertinentes de l'espace d'états, (ii) des majorants et minorants ($U$ et $L$) de la fonction de valeur optimale $V^*$, et (iii) des mises à jour ponctuelles de ces encadrements en les états rencontrés en suivant des trajectoires. Alors que HSVI a été initialement introduit pour résoudre des POMDP, nous nous concentrons sur sa version générique (MDP) étudiée dans [24] (présentée dans l'algorithme 1) plutôt que BRTDP, parce qu'il vient avec des garanties théoriques plus fortes (par exemple, il converge en temps fini plutôt qu'à la limite). HSVI et BRTDP sélectionnent tous deux des actions de manière gourmande par rapport au majorant, et arrêtent d'échantillonner des trajectoires quand l'écart $U(s_0) - L(s_0)$ est sous un seuil $\epsilon > 0$. Pour sélectionner le prochain état étant donnés $s$ et $a$, HSVI choisi un état $s'$ maximisant $excess(s') = P_a(s'|s)(U(s') - L(s') - \gamma^{-\delta}\epsilon)$, où $\delta$ est la profondeur de $s'$ dans la trajectoire courante (de manière à se focaliser sur les états dont les mises à jours vont plus vraisemblablement aider). [1]

**Algorithme 1 :** Heuristic Search Value Iteration

```
Fct HSVI (ε)
    Initialize L and U
    while (U(s_0) − L(s_0)) > ε do
        RecursivelyTry (s_0, δ = 0)
    return L
Fct RecursivelyTry (s, δ)
    if (U(s) − L(s)) > γ^{−δ} ε then
        Update (s)
        a* ∈ arg max_{a∈A} Q^U(s, a)
        s' ∈ arg max_{s"∈S} Pr(s"|s, a*)
          × (U(s") − L(s") − γ^{−δ} ε)
        RecursivelyTry (s', δ + 1)
        Update (s)
    return
Fct Update (s)
    L ← Update (L, s)
    U ← Update (U, s)
```

1. BRTDP échantillone un état selon une distribution qui ressemble à une version normalisée de cette fonction excès.

# 3 Encadrer un jeu en forme normale

Comme expliqué dans l'introduction, nous souhaiterions proposer des algorithmes de recherche heuristique (à la HSVI) pour des zs-SG à 2 joueurs, c'est-à-dire en concentrant l'effort computationel sur les parties pertinentes de l'espace d'état en exploitant (i) la connaissance de l'état initial $s_0$, et (ii) des heuristiques admissibles pour les deux joueurs, qui seront employées pour initialiser le majorant et le minorant de la fonction de valeur. Alors que la recherche heuristique classique (1 joueur contre la nature) requière seulement un majorant de la fonction de valeur pour guider la recherche de manière optimiste, les zs-SG à 2 joueurs requièrent à la fois un majorant et un minorant (de manière à être optimiste pour les deux joueurs). Nous allons donc calculer des équilibres de Nash (NE) à la fois pour les jeux de matrices de Shapley majorant $\Gamma^s(U)$ et minorant $\Gamma^s(L)$ dans chaque état rencontré $s$. Cela soulève une première question : Que se passe-t-il si, dans un état $s$, les deux NE sont égaux ? Comme démontré par les deux résultats suivants, dans ce cas la valeur obtenue est la valeur (SGP)NE pour cet état.

**Lemme 1** *Si le gain $v^{up}$ du jeu à somme nulle en forme normale $\Gamma^{up}$ majore le gain $v^{lo}$ du jeu à somme nulle en forme normale (et de même dimensions) $\Gamma^{lo}$, alors* $\text{NEV}(\Gamma^{up}) \geq \text{NEV}(\Gamma^{lo})$.

**Preuve 1**

$$\forall a^1, a^2 \quad v^{up}(a^1,a^2) \geq v^{lo}(a^1,a^2)$$

$$\forall a^1 \quad \min_{a^2} v^{up}(a^1,a^2) \geq \min_{a^2} v^{lo}(a^1,a^2)$$

$$\max_{d^1} \min_{a^2} \sum_{a^1} d^1(a^1) v^{up}(a^1,a^2) \geq \max_{d^1} \min_{a^2} \sum_{a^1} d^1(a^1) v^{lo}(a^1,a^2)$$

*c'est-à-dire,* $\text{NEV}(\Gamma^{up}) \geq \text{NEV}(\Gamma^{lo})$. □

**Corollaire 1** *Dans un jeu stochastique à somme nulle (zs-SG), si, dans un état $s$, les valeurs NE de $\Gamma^s(L)$ et $\Gamma^s(U)$ sont égales, alors* $\text{NEV}(\Gamma^s(L)) = \text{NEV}(\Gamma^s(U))$ *est la valeur NE,* $\text{NEV}^*(s)$*, du jeu de Shapley à somme nulle obtenu après convergence dans cet état (pour le SGPNE de ce zs-SG).*

**Preuve 2** *En appliquant le lemme précédent dans un état $s$, on a toujours dans notre cadre :* $\text{NEV}(\Gamma^s(L)) \leq \text{NEV}^*(s) \leq \text{NEV}(\Gamma^s(U))$*. Ainsi, quand* $\text{NEV}(\Gamma^s(L)) = \text{NEV}(\Gamma^s(U))$*, c'est aussi la valeur de* $\text{NEV}^*(s)$. □

La section suivante discute de la façon de dériver une variante de HSVI pour les jeux stochastiques à deux joueurs et somme nulle.

# 4 Algorithme et preuve de convergence

Cette section montre comment adapter HSVI au cadre zs-SG. Tant que $\text{NEV}(\Gamma^{s_0}(U)) - \text{NEV}(\Gamma^{s_0}(L)) > \epsilon$, HSVI

génère des trajectoires en appliquant, dans chaque état rencontré, (i) une règle de sélection d'action et (ii) une règle de sélection du prochain état.[2] Nous présentons maintenant ces règles et d'autres détails de l'algorithme, et prouvons qu'il trouve des stratégies $\epsilon$-optimale en $s_0$ en temps fini.

## 4.1 Sélection de l'action jointe

Dans le cadre mono-joueur classique, agir de manière gourmande par rapport au majorant (une estimation optimiste) est une règle de sélection d'action valide pour garantir la convergence. Dans un zs-SG, une estimation optimiste pour le joueur 1 est pessimiste pour le joueur 2. Supposant que le joueur 1 n'a qu'une seule action possible, agir de manière gourmande par rapport au majorant est l'opposé de ce que devrait faire le joueur 2.
La règle d'exploration proposée est, étant dans l'état $s$,

— le joueur 1 choisit l'action stochastique $d^{U,1}$ de manière gourmande par rapport au majorant $U$, c'est-à-dire en jouant au jeu de matrice à somme nulle de Shapley $\Gamma^s(U)$ ; et
— le joueur 2 choisit l'action stochastique $d^{L,2}$ de manière gourmande par rapport au minorant $L$, c'est-à-dire en jouant au jeu de matrice à somme nulle de Shapley $\Gamma^s(L)$.

On appelle la règle de décision jointe résultante $d = (d^{U,1}, d^{L,2})$ une *tentative de NES* en $s$. La figure 1 représente la matrice de jeu pour un état $s$ avec les actions impliquées dans les stratégies en NE $\text{NES}(\Gamma^s(U))$ et $\text{NES}(\Gamma^s(L))$ qui ont été trouvées.

## 4.2 Sélection du prochain état

Après avoir choisit une stratégie jointe, le prochain état $s'$ peut être sélectionné de différentes manières (par exemple, en échantillonnant $a^1$ de $d^1(\cdot)$, $a^2$ de $d^2(\cdot)$, puis $s'$ de $P_{a^1,a^2}(\cdot|s)$ (à la (L)RTDP). Toutefois, pour préserver les garanties de convergence de HSVI, nous préférons sélectionner un prochain état $s'$ maximisant

$$\sum_{a^1,a^2} d^1(a^1)d^2(a^2)P_{a^1,a^2}(s'|s) \underbrace{\left[U(s') - L(s') - \gamma^{-\delta}\epsilon\right]}_{excess(s',\delta)},$$

où $\delta$ est la profondeur courante de la trajectoire.

## 4.3 Mise à jour de la valeur

Comme pour HSVI classique, nous utilisons des opérateurs *à base de points* pour mettre à jour les minorants et majorants (c'est-à-dire des opérateurs qui sont appliqués en des points particuliers $s$), et définissons :

— $L$ un minorant *uniformément améliorable* (UI) comme vérifiant $\mathcal{H}L \geq L$ ; et
— $\mathcal{K}^L_\cdot$ un opérateur de mise à jour à base de point *fort* (*strong*) pour le minorant comme vérifiant, pour chaque $s$ où il est appliqué et chaque $L$, (i) $(\mathcal{H}L)(s) = (\mathcal{K}^L_s L)(s)$ et (ii) $(\mathcal{H}L)(s') \geq (\mathcal{K}^L_s L)(s')$ dans tout autre point $s'$

2. Nous utilisons le terme « règle » plutôt que « stratégie » pour éviter de possibles confusions.

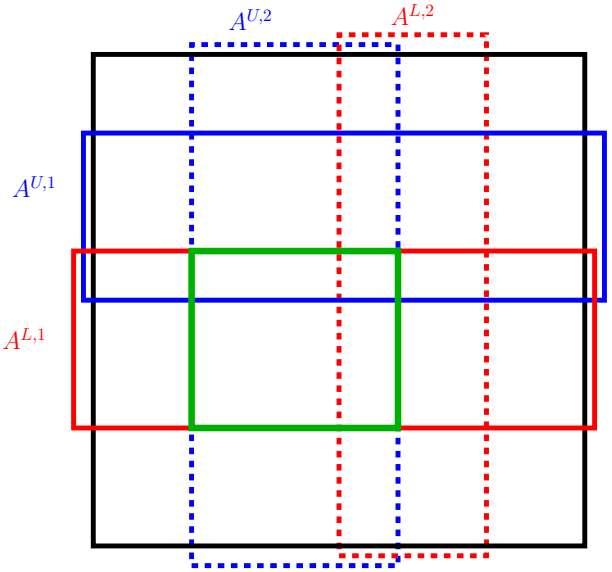


FIGURE 1 – La matrice de jeu pour un état $s$, avec les ensembles d'actions apparaissant dans les NES trouvées pour les jeux $\Gamma^s(U)$ et $\Gamma^s(L)$. Notes : (i) on suppose les actions ordonnées de manière à ce que celles intervenant dans les équilibres de Nash calculés soient regroupées ; (ii) le carré vert indique les actions jointes qui sont considérées par l'exploration ; (iii) les ensembles $A^{X,i}$ ne contiennent pas toutes les actions qui pourraient apparaître dans les NES.

(les mêmes définitions s'appliquent pour le majorant $U$ en inversant les opérateurs d'inégalité). L'améliorabilité uniforme est nécessaire pour garantir que le minorant (resp. le majorant) reste un minorant (resp. un majorant). La propriété forte est requise pour garantir la convergence vers la fonction optimale à la limite.
Dans notre cadre, nous utilisons pour le minorant comme pour le majorant des représentations tabulaires et un même opérateur de mise à jour $\mathcal{K}_s$, lequel, pour tout état $s$, ne met à jour que la valeur pour cet état, et ce, avec les mises à jour naturelles :

$$(\mathcal{K}_s L)(s) \stackrel{\text{def}}{=} \text{NEV}(\Gamma^s(L)),$$
$$(\mathcal{K}_s U)(s) \stackrel{\text{def}}{=} \text{NEV}(\Gamma^s(U)),$$

et laisse $L$ et $U$ inchangés pour les autres états.
Comme dans le cadre MDP : (i) l'opérateur proposé est fort, donc conservatif [24, Def. 3.24, 3.25] ; et (ii) tout opérateur de mise-à-jour conservatif préserve l'améliorabilité uniforme (UI) [24, Th. 3.29]. Il nous reste donc essentiellement à nous assurer que nous employons des initialisations induisant des encadrements UI (voir prochaine sous-section).

## 4.4 Majorants et minorants

Comme pour les POMDP, une façon d'initialiser les minorants et majorants est de calculer le point fixe d'un opérateur qui minore ou majore l'opérateur d'optimalité de Shapley $\mathcal{H}$. La table 1 montre des opérateurs candidats pour

les deux encadrements, lesquels nous décrivons aussi verbalement comme suit :

- $\mathcal{G}_{\text{SEQ}}^{\cdot}$ **:** une approximation par jeu séquentiel—aussi appelée *sérialisation*—, c'est-à-dire $U$ étant la valeur optimale if le joueur 1 agit avant le joueur 2, et $L$ l'opposé (le second joueur sachant le choix d'action du premier) ;
- $\mathcal{G}_{\text{MDP}}^{\cdot}$ **:** une approximation MDP, c'est-à-dire $U$ (respectivement $L$) étant la fonction de valeur optimale du MDP induit en affectant au joueur 2 (resp. au joueur 1) n'importe quelle stratégie fixe (par exemple des actions aléatoires) ;
- $\mathcal{G}_{\text{ORA}}^{\cdot}$ **:** une approximation MDP avec un oracle qui prédit l'action de l'adversaire randomisé (mais pas le mouvement de la nature) ; ou
- $\mathcal{G}_{\text{TRIV}}^{\cdot}$ **:** des approximations triviales, par exemple, $\forall s$, $U(s) = \frac{1}{1-\gamma}\max_{s',a^1,a^2} r(s',a^1,a^2)$ et $L(s) = \frac{1}{1-\gamma}\min_{s',a^1,a^2} r(s',a^1,a^2)$.

Dans la table 1, les opérateurs au sein d'une même boîte retourne des fonctions « plus grandes ou égales » à celles des opérateurs dans une boîte plus basse. À l'intérieur de chacune des deux boîtes contenant trois opérateurs, les seules relations d'ordre connues sont : $\mathcal{G}^U_{\text{ORA}}\boldsymbol{v} \geq \mathcal{G}^U_{\text{MDP}}\boldsymbol{v}$ et $\mathcal{G}^L_{\text{MDP}}\boldsymbol{v} \geq \mathcal{G}^L_{\text{ORA}}\boldsymbol{v}$. Évidemment, un problème est de savoir si dériver ces fonctions encadrantes en vaut l'effort de calcul. Les points fixes des opérateurs $\mathcal{G}^{\cdot}_{\text{TRIV}}$ peuvent être calculés en temps constant (voir formules ci-dessus), et servir d'initialisations pour calculer les points fixes des autres opérateurs, par exemple, à nouveau par recherche heuristique. Les opérateurs $\mathcal{G}^{\cdot}_{\text{ORA}}$ ne méritent pas d'être considérés parce qu'ils fourniraient de moins bons encadrements que d'autres opérateurs de même coût de calcul.

**Améliorabilité uniforme** Sans perte de généralité, discutons de l'améliorabilité uniforme (UI) seulement pour les majorants. Comme proposé ci-dessus, la majorant initial $U$ est obtenu comme l'unique point fixe d'un opérateur de mise-à-jour $\mathcal{K}^{up}$ qui « majore » toujours l'opérateur d'optimalité de Shapley $\mathcal{H}$. En conséquence, $\mathcal{H}U \leq \mathcal{K}^{up}U = U$, de sorte que $U$ est uniformément améliorable. Cette approche est similaire au théorème 3.20 dans [24, page 71], et permet aussi de démontrer que les minorants proposés sont UI.

**Politique par défaut** Finalement, nous allons voir en discutant de la politique à exécuter effectivement, que le $U$ initial (resp. $L$) doit venir avec une stratégie pour le joueur 2 (resp. pour le joueur 1) de manière à assurer que le pire résultat associé avec cette fonction de valeur encadrante est garantie (ce qui est le cas pour les trois approches proposées ci-dessus). Cette exigence nous empêche d'exploiter les NEV nulles des états auto-symétriques dans les zs-SG symétriques (c'est-à-dire quand les situations des deux joueurs sont en miroir l'une de l'autre). [3]

3. On pourrait éventuellement exploiter le fait que, si $s$ et $s'$ sont des états symétriques, alors $U(s) = -L(s')$.

TABLE 1 – Divers opérateurs de mise à jour pour la fonction de valeur, *presque* ordonnés selon les valeurs résultantes. Au milieu se trouve l'opérateur de Shapley optimal. Au dessus se trouvent les opérateurs pour l'initialisation du majorant $U$. En dessous se trouvent les opérateurs pour l'initialisation du minorant $L$.

| | | | |
|---|---|---|---|
| $(\mathcal{G}^U_{\text{TRIV}}\boldsymbol{v})(s)$ | $=$ | $\max_{s',a^1,a^2,s"} r(s',a^1,a^2)+\gamma\boldsymbol{v}(s")$ | |
| $(\mathcal{G}^U_{\text{ORA}}\boldsymbol{v})(s)$ | $=$ | $\mathbb{E}_{a^2\sim\text{Unif}(\mathcal{A}^2)}\max_{a^1}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{G}^U_{\text{MDP}}\boldsymbol{v})(s)$ | $=$ | $max_{a^1}\mathbb{E}_{a^2\sim\text{Unif}(\mathcal{A}^2)}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{G}^U_{\text{SEQ}}\boldsymbol{v})(s)$ | $=$ | $\min_{a^2}\max_{a^1}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{H}\boldsymbol{v})(s)$ | $=$ | $\min_{d^2}\max_{a^1}$ | $\Gamma^s(\boldsymbol{v})(a^1,d^2)$ |
| $=\text{NEV}(\Gamma^s(\boldsymbol{v}))$ | $=$ | $\max_{d^1}\min_{a^2}$ | $\Gamma^s(\boldsymbol{v})(d^1,a^2)$ |
| $(\mathcal{G}^L_{\text{SEQ}}\boldsymbol{v})(s)$ | $=$ | $\max_{a^1}\min_{a^2}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{G}^L_{\text{MDP}}\boldsymbol{v})(s)$ | $=$ | $\min_{a^2}\mathbb{E}_{a^1\sim\text{Unif}(\mathcal{A}^1)}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{G}^L_{\text{ORA}}\boldsymbol{v})(s)$ | $=$ | $\mathbb{E}_{a^1\sim\text{Unif}(\mathcal{A}^1)}\min_{a^2}$ | $\Gamma^s(\boldsymbol{v})(a^1,a^2)$ |
| $(\mathcal{G}^L_{\text{TRIV}}\boldsymbol{v})(s)$ | $=$ | $\min_{s',a^1,a^2,s"} r(s',a^1,a^2)+\gamma\boldsymbol{v}(s")$ | |

## 4.5 Critère de terminaison des trajectoires et preuve de convergence

Interrompons une trajectoire quand on atteint un état $s$ à profondeur $\delta$ s'il est *NEV-fini*, c'est-à-dire quand l'excès, $excess(s,\delta) \stackrel{\text{def}}{=} U(s) - L(s) - \gamma^{-\delta}\epsilon$, est négatif ou nul. L'utilisation de ce critère de terminaison des trajectoires est le dernier élément nécessaire pour garantir que la version zs-SG de HSVI converge en temps fini. Nous avons essentiellement besoin de prouver le lemme clef suivant avant d'exploiter des résultats de [24].

**Lemme 2** (Cf. *[24, Lemme 6.1]*)
*Soient $\mathcal{K}$. un opérateur de mise-à-jour fort, $L$ et $U$ des fonctions de valeurs uniformément améliorables, $s$ un état, et $d^* = (d^{U,1}, d^{L,2})$ la NES tentative courante en $s$. Alors*

$$width(\mathcal{K}_s\hat{V}(s)) \leq \gamma \sum_{s'} P_{d^{U,1},d^{L,2}}(s'|s)\, width(\hat{V}(s')),$$

*où $\hat{V}$ désigne la paire $(L,U)$ et $P_{\cdot,\cdot}$ est naturellement étendue pour les stratégies mixtes (/actions stochastiques).*

**Preuve 3** *Nous avons*

$$\begin{aligned}
&width(\mathcal{K}_s\hat{V}(s)) \\
&= width(\mathcal{H}\hat{V}(s)) && (\mathcal{K}_s \textit{ est fort}) \\
&= \mathcal{H}U(s) - \mathcal{H}L(s) && (\textit{déf. de } width(\cdot)) \\
&= \text{NEV}(\Gamma^s(U)) - \text{NEV}(\Gamma^s(L)) && (\textit{déf. de } \mathcal{H}) \\
&= \max_{d^1}\min_{a^2} v^U(s,d^1,a^2) - \min_{d^2}\max_{a^1} v^L(s,d^1,a^2) \\
&\leq v^U(s,d^{U,1},d^{L,2}) - v^L(s,d^{U,1},d^{L,2})
\end{aligned}$$

$$= \gamma \sum_{s'} P_{d^{U,1},d^{L,2}}(s'|s)(U(s') - L(s'))$$
$$= \gamma \sum_{s'} P_{d^{U,1},d^{L,2}}(s'|s) width(\hat{V}(s')). \qquad \square$$

Ce résultat est important parce qu'il majore la largeur en $s$ après mise-à-jour par une combinaison linéaire des largeurs des états immédiatement atteignables depuis $s$ avec $d^*$. Ainsi, on peut réduire la largeur en $s$ si l'on est capable de réduire suffisamment les largeurs de ces états suivants (dont la liste peut changer d'une visite de $s$ à l'autre).

**Corollaire 2** *La variante de HSVI pour zs-SG spécifiée par les processus de sélection, les opérateurs de mise-à-jour, et le critère de terminaison de trajectoires ci-dessus converge en temps fini.*

**Preuve 4** *(ébauche inspirée de [24]) D'abord, le lemme précédent reste valable si l'on remplace la largeur (width) par l'excès (excess) [24, Lemma 6.2]. En conséquence, si tous les successeurs $s'$ de $s$ atteignables par $d^*$ sont NEV-finis, alors $s'$ est aussi NEV-fini [24, Lem. 6.3]. Aussi, HSVI sélectionne toujours un successeur non NEV-fini s'il en existe un [24, Lem. 6.4], et tous les états au-delà de la profondeur $\delta_{\max} \stackrel{\text{def}}{=} \lceil \log_\gamma(\epsilon/\|U - L\|_\infty) \rceil$ sont NEV-finis [24, Lem. 6.4] (par exemple, avec $U_0$ et $L_0$ les encadrements initiaux). Une conséquence est que toute trajectoire se termine et l'avant-dernier état visité devient NEV-fini [24, Lem. 6.7]. De là, l'arbre des états atteignables depuis $s_0$ a une profondeur bornée $\delta_{\max}$ et un facteur de branchement fini $\beta$, de sorte que $s_0$ sera NEV-fini (à profondeur 0) après au plus $\frac{\beta^{\delta_{\max}}-1}{\beta-1}$ trajectoires.* $\square$

L'algorithme complet est détaillé dans l'algorithme 2.

## 4.6 Exécution de la politique

Dans le cadre HSVI (ou BRTDP), un joueur doit agir de manière gourmande par rapport (i) à une estimation optimiste de la fonction de valeur optimale pour que la recherche heuristique converge (*cf.* la sélection d'action présentée ci-dessus), et (ii) à une estimation pessimiste lors de l'exécution de manière à garantir le pire retour espéré. Plus précisément, lors de l'exécution, si l'on est dans un état $s$, le joueur 1 (resp. 2) doit trouver une stratégie en équilibre de Nash $d^L$ pour $\Gamma^s(L)$ (resp. $d^U$ pour $\Gamma^s(U)$) et agir suivant $d^{L,1}$ (resp. $d^{U,2}$). Ainsi, le joueur 1 s'assure d'obtenir la valeur du « niveau de sécurité » associée à $\Gamma^s(L)$. L'exemple suivant illustre pourquoi agir de manière optimiste lors de l'exécution peut être préjudiciable.

**Exemple 1** *Prenons le jeu de matrix $\Gamma^\alpha$ décrit en table 2 avec ses majorant et minorant $U$ et $L$. $U$ et $L$ ont la même NEV, 0. Toutefois, selon $U$, n'importe quelle stratégie $(p, 1-p)$ du joueur (ligne) 1 ($p \in [0,1]$) fait partie d'une NES. Si le joueur 1 joue $(p, 1-p)$ et le joueur 2 joue $(0,1)$ (par exemple, parce que ce dernier sait qu'$\alpha$ est négatif), alors la valeur obtenue est $p\alpha$ ($< 0$), ce qui représente une perte pour le joueur 1.*

**Algorithme 2 :** zsSG-HSVI

```
1  Fct HSVI (ε)
2      Initialize L and U
3      while (U(s_0) − L(s_0)) > ε do
4          RecursivelyTry (s_0, δ = 0)
5      return L
6  Fct RecursivelyTry (s, δ)
7      if (U(s) − L(s)) > γ^{−δ} ε then
8          Update (s)
9          d^U ← NES(Γ^s(U))
10         d^L ← NES(Γ^s(L))
11         s' ∈ arg max_{σ∈S} Σ_{a^1,a^2} d^{U,1}(a^1) d^{L,2}(a^2)
               × T(s, a^1, a^2, σ) [U(σ) − L(σ) − γ^{−δ} ε]
12         RecursivelyTry (s', δ + 1)
13         Update (s)
14     return
15 Fct Update (s)
16     L ← Update (L, s) /* uses NEV(Γ^s(L))   */
17     U ← Update (U, s) /* uses NEV(Γ^s(U))   */
```

TABLE 2 – Un exemple de jeu $\Gamma^\alpha$ (où $\alpha \in [-1, +1]$) avec des exemples de jeux majorant et minorant $U$ et $L$, leurs NES et leurs valeurs (à l'équilibre de Nash). Le joueur 1 est le joueur ligne.

| jeu | matrice | NES | valeur |
|---|---|---|---|
| $U$ | $\begin{bmatrix} 0 & +1 \\ 0 & 0 \end{bmatrix}$ | $((p, 1-p), (1, 0))$ | 0 |
| $\Gamma_\alpha$ | $\begin{bmatrix} 0 & \alpha \\ 0 & 0 \end{bmatrix}$ | [voir texte] | 0 |
| $L$ | $\begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix}$ | $((0, 1), (q, 1-q))$ | 0 |

*Notes :*

- *Cet exemple montre aussi que l'égalité des NEV de $U$ et $V$ ne nécessite pas leur égalité sur le support de la tentative d'équilibre de Nash (c'est-à-dire sur les valeurs de la matrice impliquées).*
- *$\Gamma^\alpha$ a toujours une NEV de 0, mais ses NES possibles dépendent du signe ou de la nullité d'$\alpha$.*

L'initialisation de $L$ devrait ainsi venir avec une stratégie par défaut pour le joueur 1 (pré-calculée ou calculable en ligne) de manière à garantir que le joueur 1 a une action appropriée à effectuer même dans les états non rencontrés par l'algorithme (et de même pour $U$ et le joueur 2).

# 5 Discussion

Cet article propose un algorithme de recherche heuristique pour les jeux stochastiques à (deux joueurs et) somme nulle

avec un état intial. Les critères opposés des joueurs amène naturellement à des algorithmes avec majorant et minorant tels que HSVI et BRTDP. Le principe de l'optimisme face à l'incertain est maintenu, ainsi que celui d'agir (lors de la phase d'exécution) suivant l'estimation pessimiste. Comme le HSVI original, il est prouvé que l'algorithme obtenu converge vers une solution $\epsilon$-optimale en temps fini.
Nous énumérons maintenant quelques directions pour des travaux futurs.

**Expérimentations** zs-HSVI a encore besoin d'être implémenté et évalué empiriquement sur des bancs d'essai tels que un jeu de football à deux joueurs sur plateau [11, 10] ; le problème de contrôle de flux routeur/serveur [1, 10] ; ou des Garnets (c'est-à-dire des problèmes générés automatiquement) [16]. D'autres jeux de plateau pourraient être considérés, tels qu'Alesia [14] ou Goofspiel/GOPS [19], même si leurs critères ne sont habituellement pas atténués. L'analyse devrait comparer (i) zs-HSVI avec d'autres solveurs de zs-SG, et (ii) différentes méthodes d'initialisation des majorant et minorant.

**Améliorations** D'abord, il est commun d'améliorer le comportement anytime de HSVI en appelant la procédure de génération de trajectoires avec un paramètre $\epsilon$ de calcul de l'écart égal à une grande fraction de la largeur courant en $s_0$, par exemple $0,9(U(s_0) - L(s_0))$. Cela empêche les premières trajectoires d'être trop longues, et ressemble aux approches par approfondissement itérative (*iterative deepening*).
Par ailleurs, un problème est que de nombreux programmes linéaires (LP), potentiellement grands, devront être résolus. Comme dans DO$\alpha\beta$ [6], une première idée est de réduire les coûts de calcul en utilisant des méthodes avec oracles, c'est-à-dire des méthodes itératives qui permettent de résoudre de grands jeux en forme normale sans considérer toutes les stratégies possibles. [4]
Une autre idée est d'exploiter le fait que, quand un état est revisité, les LP qui doivent être résolus seront souvent très similaires à ceux qui ont déjà été résolus lors de la dernière visite. Cela suggère d'employer des techniques de *bootstrapping* (amorçage), c'est-à-dire d'initialiser les solveurs LP (si possible avec double oracle) en utilisant les dernières solutions.

**Autre schémas algorithmiques** Nous nous sommes concentrés jusqu'ici sur un algorithme reposant sur HSVI. Il serait intéressant de regarder si d'autres schémas algorithmiques pour la recherche heuristique, tels que LAO*, RTDP et HDP, pourraient être employés pour des jeux stochastiques. Comme souligné par Bonet et Geffner [4], tous les trois correspondent au schéma cherche-et-RÉVISE (FIND-and-REVISE), de sorte que les principaux résultats de convergence devraient pouvoir être abordés conjointement.

4. Des techniques apparentées ont été utilisées pour améliorer la procédure d'élagage dans les POMDP [29].

**Problèmes sans atténuation** Le présent travail fait l'hypothèse qu'un critère atténué est utilisé (comme requis par le HSVI de base). Une question qui se pose naturellement est comment adapter ce travail à un critère non atténué : Quels schémas algorithmiques pourraient être employés ? Sous quelles hypothèses ?

**SG à somme générale** Une direction pour des recherches futures est l'extension du présent travail à la résolution de jeux stochastiques à somme générale (avec si possible plus de deux joueurs). On peut s'attendre à ce que cela implique le maintient de majorants et minorants pour les critères de chaque joueur. Mais un premier problème est de choisir un concept solution approprié, puisqu'un jeu matriciel donné n'a plus nécessairement une valeur d'équilibre de Nash unique. Si on planifie pour tous les joueurs en même temps (en faisant l'hypothèse que chacun s'engagera à suivre la stratégie qui lui est affectée), alors n'importe quel NE peut être une solution valide. Mais que faire si on planifie pour un seul joueur sans avoir de contraintes sur les stratégies des autres joueurs ?

**POSG / Occupancy zs-SG** Un objectif à plus long terme de ce travail est la résolution de jeux stochastiques partiellement observables (POSG) en utilisant une variante de HSVI et une approximation de la fonction de valeur comme cela a été fait pour les Dec-POMDP [8], c'est-à-dire dans le cas complètement collaboratif. Dans le cas d'un POSG à 2 joueurs et somme nulle, cela supposerait de résoudre le problème comme un *occupancy zs-SG*, où l'état d'occupation est une statistique suffisante pour le planificateur. Le problème de planification devient alors déterministe, mais l'espace d'états est un simplexe (un espace de distributions de probabilités). Dans les occupancy MDP induits par des Dec-POMDP, le facteur de branchement reste fini parce que seules des règles de décision déterministes sont considérées, lesquelles sont en nombre fini. À l'inverse, dans les occupancy zs-SG, des règles de décision stochastiques doivent être considérées, lesquelles sont en nombre infini. On pourrait vraisemblablement obtenir une convergence $\epsilon$-optimale en temps fini en utilisant un partitionnement récursif (qui induirait une discrétisation de l'espace d'actions, et donc de l'espace des états) si des approximations Lipschitz-continues (plutôt que PWLC) de la fonction de valeur peuvent être employées (comme nous le supposons pour l'instant).

# Références


[1] E. Altman : Flow control using the theory of zero-sum Markov games. *IEEE Trans. on Auto. Control*, (39), 1994.

[2] R. Bellman : The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6), 1954.

[3] D. Bertsekas et J. Tsitsiklis : *Neurodynamic Programming*. 1996.

[4] B. Bonet et H. Geffner : Faster heuristic search algorithms for planning with uncertainty and full feedback. Dans *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'03)*, 2003.

[5] B. Bonet et H. Geffner : Labeled RTDP : Improving the convergence of real-time dynamic programming. Dans *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS'03)*, 2003.

[6] B. Bošanský, V. Lisý, M. Lanctot, J. Čermák et M. H. Winands : Algorithms for computing strategies in two-player simultaneous move games. *Artificial Intelligence*, 237, 2016.

[7] V. Bulitko et G. Lee : Learning in real-time search : A unifying framework. *Journal of Artificial Intelligence Research (JAIR)*, 25, 2006.

[8] J. Dibangoye, C. Amato, O. Buffet et F. Charpillet : Optimally solving Dec-POMDPs as continuous-state MDPs. *Journal of Artificial Intelligence Research*, 55, 2016.

[9] E. Hansen et S. Zilberstein : LAO* : A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, 129, 2001.

[10] M. G. Lagoudakis et R. Parr : Value function approximation inf zero-sum Markov games. Dans *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI'02)*, 2002.

[11] M. Littman : Markov games as a framework for multi-agent reinforcement learning. Dans *Proceedings of the Eleventh International Conference on Machine Learning (ICML'94)*, 1994.

[12] M. Littman et C. Szepesvári : A generalized reinforcement learning model : Convergence and applications. Dans *Proceedings of the International Conference on Machine Learning (ICML'96)*, 1996.

[13] H. B. McMahan, M. Likhachev et G. J. Gordon : Bounded real-time dynamic programming : RTDP with monotone upper bounds and performance guarantees. Dans *Proceedings of the twenty-second International Conference on Machine Learning*, 2005.

[14] C. Meyer, J. Ganascia et J. Zucker : Learning strategies in games by anticipation. Dans *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, (IJCAI'97)*, 1997.

[15] J. Nash : Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36, 1950.

[16] J. Pérolat, B. Piot, M. Geist, B. Scherrer et O. Pietquin : Softened approximate policy iteration for Markov games. Dans *Proceedings of the International Conference on Machine Learning (ICML 2016)*, 2016.

[17] T. E. S. Raghavan et J. A. Filar : Algorithms for stochastic games – a survey. *Zeitschrift für Operations Research*, 35(6), Nov 1991.

[18] D. M. Roijers, P. Vamplew, S. Whiteson et R. Dazeley : A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research (JAIR)*, 48, 2013.

[19] S. M. Ross : Goofspiel - the game of pure strategy. *Journal of Applied Probability*, (8), 1971.

[20] S. Russell et P. Norvig : *Artificial Intelligence : A Modern Approach*. 2010.

[21] A. Saffidine, H. Finnsson et M. Buro : Alpha-Beta pruning for games with simultaneous moves. Dans *Proceedings of the 26th AAAI Conference (**AAAI**)*, Toronto, Canada, juil. 2012.

[22] L. S. Shapley : Stochastic games. *Proceedings of the National Academy of Sciences (PNAS)*, 39, 1953.

[23] L. S. Shapley : Some topics in two person games. *Annals of Mathematical Studies*, 5, 1964.

[24] T. Smith : *Probabilistic Planning for Robotic Exploration*. Thèse de doctorat, The Robotics Institute, Carnegie Mellon University, 2007.

[25] T. Smith et R. Simmons : Point-based POMDP algorithms : Improved analysis and implementation. Dans *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

[26] E. Solal : Stochastic games. *Encyclopedia of Database Systems*, 2009.

[27] D. Szer, F. Charpillet et S. Zilberstein : MAA* : A heuristic search algorithm for solving decentralized POMDPs. Dans *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI'05)*, 2005.

[28] J. von Neumann : Zur Theorie der Gesellschaftsspiele. *Math. Annalen*, 100, 1928.

[29] E. Walraven et M. T. J. Spaan : Accelerated vector pruning for optimal POMDP solvers. Dans *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, 2017.