Résolution exacte des Dec-POMDPs comme des MDPs continus

Jilles Steeve Dibangoye, Christopher Amato, Olivier Buffet, François Charpillet

Résolution exacte des Dec-POMDPs comme des MDPs continus

Jilles S. Dibangoye[1], Christopher Amato[2], Olivier Buffet[1], François Charpillet[1]

[1] INRIA – LORIA
Villers-lès-Nancy, France
firstname.lastname@loria.fr

[2] CSAIL
MIT
Cambridge, MA, USA
camato@csail.mit.edu

Résumé : Résoudre optimalement des processus décisionnels de Markov partiellement observables et décentralisés (Dec-POMDPs) est un problème combinatoire difficile. Les algorithmes actuels cherchent pour chaque agent à travers l'espace complet des politiques sur les historiques. A cause de la croissance doublement exponentielle de cet espace quand l'horizon de planification croît, ces méthodes deviennent rapidement insolubles. Toutefois, dans des problèmes réels, calculer des politiques sur l'espace des historiques complet est souvent inutile. L'extraction des informations pertinentes d'un historique permet de réduire le nombre d'historiques utiles. Nous montrons qu'en transformant un Dec-POMDP en un MDP continu, nous sommes capables de trouver et exploiter ces représentations à faible dimensionalité. En utilisant cette nouvelle transformation, nous pouvons appliquer des techniques efficaces pour la résolution de POMDPs et de MDPs continus. En combinant un algorithme de recherche générique et une réduction de la dimensionalité fondée sur la sélection de caractéristiques, nous introduisons une nouvelle approche pour résoudre de manière optimale des problèmes avec des horizons de planification significativement plus grands que les méthodes antérieures.

1 Introduction

Les processus décisionnels de Markov partiellement observables et décentralisés (Dec-POMDPs) ont été étudiés comme un modèle général pour la prise de décision stochastique dans les systèmes multi-agents coopératifs (Bernstein *et al.*, 2002; Szer *et al.*, 2005; Boularias & Chaib-draa, 2008; Amato *et al.*, 2009; Bernstein *et al.*, 2009; Aras & Dutech, 2010; Dibangoye *et al.*, 2011; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013). Alors que de multiples algorithmes ont été développés, les algorithmes optimaux actuels ne peuvent pas passer à l'échelle y compris pour des problèmes jouets. Ce n'est pas inattendu étant donnée la complexité NEXP au pire cas (Bernstein *et al.*, 2002), mais de nombreux problèmes réels ont une structure qui devrait permettre capacité de passage à l'échelle.

Les algorithmes optimaux actuels pour Dec-POMDPs cherchent dans l'espace des solutions (ou politiques), lesquelles associent des actions aux historiques des actions effectuées et des observations perçues (Hansen *et al.*, 2004; Szer *et al.*, 2005; Boularias & Chaib-draa, 2008; Amato *et al.*, 2009; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013). Ces approches procèdent typiquement en faisant croître itérativement ces politiques soit suivant une approche descendante (en utilisant une recherche heuristique) (Szer *et al.*, 2005; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013) soit suivant une approche ascendante (en utilisant la programmation dynamique) (Hansen *et al.*, 2004; Boularias & Chaib-draa, 2008; Amato *et al.*, 2009) jusqu'à ce que des politiques pour l'horizon complet du problème soient construites. L'horizon du problème augmentant, la croissance doublement exponentielle du nombre de politiques possibles rend les méthodes de résolution exactes inutilisables. Des méthodes pour améliorer la capacité de passage à l'échelle en compressant les politiques (Boularias & Chaib-draa, 2008) et les historiques (Oliehoek *et al.*, 2009) ont commencé à être explorées, mais la capacité de passage à l'échelle reste limitée pour de nombreux problèmes.

Au contraire, des progrès significatifs ont été faits dans la taille des problèmes résolus en tant que processus de décision markoviens complètement et partiellement observables (MDP et POMDP). Une raison des

progrès dans les MDP a été l'utilisation de la programmation dynamique approchée et d'approximations de fonctions (Powell, 2007; De Farias & Van Roy, 2003) pour représenter l'état du système (et la fonction de valeur) de manière plus concise. Pour les POMDP, des algorithmes efficaces ont été développés en reformulant les problèmes comme des belief MDP qui utilisent des distributions de probabilité sur des états du système,à savoir des *états de croyance* (*belief state*) (Smallwood & Sondik, 1973). Ce belief MDP est un MDP à espace d'états continu avec une fonction de valeur linéaire par morceaux et convexe, permettant aux algorithmes pour POMDP de résoudre de grands problèmes tout en gardant parfois des bornes de performance (Shani *et al.*, 2012).

Pour profiter de ces avancées dans les solveurs de POMDP et MDP, nous résolvons un Dec-POMDP en le ré-écrivant comme un MDP à espace d'états continu. L'espace d'états consiste en toutes les distributions de probabilité accessibles sur les états du système et les historiques des agents (que nous appelons *états d'occupation*) et l'espace d'actions consiste en toutes les règles de décision associant des historiques à des actions. Un premier résultat de cet article est la démonstration que l'état d'occupation est suffisant pour la planification optimale dans les Dec-POMDP. Ensuite, nous montrons que la fonction de valeur est une fonction linéaire par morceaux et convexe de l'état d'occupation. En conséquence, les algorithmes pour POMDP peuvent pour la première fois être appliqués directement aux Dec-POMDP. C'est une avancée théorique significative pour la planification dans les Dec-POMDPs. Toutefois, étant donné que les états d'occupation comme les règles de décision sont définis sur l'espace complet des historiques, la capacité de passage à l'échelle reste limitée. Pour l'accroître, nous remplaçons l'espace complet des historiques par un ensemble de caractéristiques de faible dimension en utilisant une technique de réduction de dimension sans perte. Cette réduction permet un passage à l'échelle en l'horizon significativement plus grand que les algorithmes de l'état de l'art antérieurs.

Le reste de ce papier est organisé comme suit. Nous présentons d'abord le contexte nécessaire dans la section 2. Nous discutons alors, en section 3, la transformation d'un Dec-POMDP en un MDP à espace d'états continu en fournissant des résultats théoriques. Ensuite, nous introduisons, en section 4, notre algorithme, *feature-based heuristic search value iteration* (FB-HSVI), lequel encode l'état d'occupation en utilisant un espace de caractéristiques de faible dimensionalité et peut générer une politique Dec-POMDP optimale. Nous concluons avec des résultats expérimentaux sur plusieurs problèmes de référence.

2 Contexte et travaux connexes

Nous passons d'abord en revue les modèles pertinents et fournissons un rapide aperçu des algorithmes Dec-POMDP optimaux.

2.1 MDP, POMDP et Dec-POMDP

Un MDP est un tuple $(S, A, \boldsymbol{P}, \boldsymbol{R})$ où S est un ensemble d'états, A est un ensemble d'actions, $\boldsymbol{P}^a$ est une matrice stochastique $|S|\times|S|$ décrivant la probabilité de transiter de l'état s vers l'état s' en utilisant l'action a, $\boldsymbol{R}^a$ est un vecteur de récompense $|S| \times 1$ définissant la récompense pour avoir exécuté l'action a dans l'état s. En résolvant un MDP, le but est de produire une *politique* π (associant à chaque état une action) devant être exécutée à chaque pas de temps t et qui maximise une mesure des récompenses accumulées — ici nous nous focalisons du *la récompense totale espérée* sur l'horizon de planification T.

Un POMDP est un tuple $(S, A, Z, \boldsymbol{P}, \boldsymbol{O}, \boldsymbol{R}, b^0)$ où $S, A, \boldsymbol{P}, \boldsymbol{R}$ sont les mêmes que dans un MDP, Z est un ensemble d'observations et l'ensemble $\boldsymbol{O}$ consiste en des matrices d'observation $\boldsymbol{O}^{az}$ de taille $|S|\times|S|$, une pour chaque paire d'observation z et d'action a. Si l'agent est incapable d'observer le vrai état du monde, il peut toutefois maintenir un état de croyance $b^t[s] = P(s^t = s \mid b^0, a^0, z^1, \cdots, a^{t-1}, z^t)$. La croyance b^0 définit l'état de croyance initial. Nous ferons référence à la séquence passée d'actions et d'observations de l'agent jusqu'au pas de temps t comme $\theta^t = (a^0, z^1, \cdots, a^{t-1}, z^t)$, une historique de longueur t. Les POMDP peuvent être résolus comme des MDP à espace d'états continu, qui sont les espaces des états de croyances du POMDP. Dans cette formulation une politique est une fonction des états de croyance vers les actions.

Un Dec-POMDP est une tuple $(S, \otimes_i A_i, \otimes_i Z_i, \boldsymbol{P}, \boldsymbol{O}, \boldsymbol{R}, b^0)$. Ces quantités sont les mêmes que pour un POMDP, mais maintenant chaque agent a ses problèmes ensembles d'actions et d'observations. Les transitions, observations et récompensent dépendent des actions choisies par tous les agents. Parce que chaque agent n'a accès qu'à ces propres observations locales, l'objectif est de maximiser une fonction de

valeur commune en exécutant des politiques qui dépendent seulement de l'historique de chaque agent. Du fait de la nature décentralisée de l'information, les Dec-POMDP ne peuvent pas être transformés naïvement en POMDP (ou MDP). En transformant le modèle Dec-POMDP en un MDP continu avec une fonction de valeur linéaire par morceaux et convexe, comme nous le démontrons en section 3.2, nous fournissons la première preuve que les méthodes pour POMDP peuvent aussi être appliquées directement au Dec-POMDP. Parce qu'un état de croyance central ne peut être calculé par les agents, les travaux antérieurs ont typiquement représenté une politique à T-étapes pour chaque agent comme un *arbre-politique* qui associe une action à chaque historique. Les détails sont discutés en section 3.1.

2.2 Solutions optimales pour Dec-POMDP

Une classe de méthodes de résolution de Dec-POMDP est fondée sur la programmation dynamique (Howard, 1960). Ici, un ensemble d'arbres-politiques à T pas, un pour chaque agent, est généré de manière ascendante (Hansen *et al.*, 2004). A chaque étape, toutes les politiques de l'étape t sont générées sur la base des politiques de l'étape $t+1$. Toute politique qui a une valeur inférieure qu'une autre politique pour tous les états et politiques possibles des autres agents sont alors élaguées (par programmation linéaire). Cette génération et cet élagage continuent jusqu'à ce que l'horizon désiré soit atteint et que les arbres avec la plus haute valeur à l'état initial soient choisies. Des méthodes de programmation dynamique plus efficaces ont été développées, réduisant le nombre d'arbres générés (Amato *et al.*, 2009) ou compressant les représentations des politiques (Boularias & Chaib-draa, 2008).

Les arbres peuvent aussi être construits de manière descendante en utilisant la recherche heuristique (Szer *et al.*, 2005). Dans ce cas, un nœud de recherche est un ensemble de politiques partielles pour les agents jusqu'à un horizon donné, t. Ces politiques partielles peuvent être évaluées jusqu'à cet horizon, et ensuite une heuristique (telle que la valeur du MDP ou POMDP associé) peut être ajoutée. Les valeurs heuristiques sont des surestimations de la vraie valeur, ce qui permet des recherches de type A* à travers l'espace des politiques possibles pour les agents, en développant les nœuds de recherche prometteurs de l'horizon $t+1$ à l'horizon t. Un cadre plus général de recherche a aussi été développé (Oliehoek *et al.*, 2008). Les travaux récents incluent le regroupement d'historiques probabilistiquement équivalentes (Oliehoek *et al.*, 2009) et le développement incrémental de nœuds dans l'arbre de recherche (Spaan *et al.*, 2011), ce qui grandement amélioré la capacité de passage à l'échelle de l'algorithme original.

Alors que les méthodes actuelles tentent de limiter le nombre de politiques prises en compte, elles reposent sur des représentations explicites des politiques qui considèrent les historiques complètes des agents. De plus, même si ces algorithmes utilisent une phase de planification hors-ligne centralisée, ils n'ont pas été capables d'identifier une statistique suffisante concise qui permet de résoudre de plus grands problèmes.

3 Les Dec-POMDP comme des MDP à espace d'états continus

Nous discutons maintenant de la transformation d'un Dec-POMDP en un MDP à espace d'états continu, en commençant par quelques définitions utiles. Cette approche utilise la phase de planification hors-ligne courante pour centraliser l'information disponible comme une distribution sur les états et les historiques des agents du point de vue d'un planificateur centralisé. Notons que le planificateur central ne perçoit pas les actions ou observations des agents pendant l'exécution mais sait quelles politiques ils sont en train d'exécuter. Les actions peuvent être sélectionnées sous la forme de règles de décision qui conditionnent les actions sur des historiques spécifiques vues par chaque agent.

3.1 Définitions préliminaires

Soient Θ_i^t et Θ^t respectivement les ensembles des historiques de l'agent i et des historiques jointes au pas de temps $t = 0, 1, \cdots, T-1$. Une *politique locale* pour l'agent i est une séquence ordonnée de *règles de décision locales*, $\pi_i \equiv \pi_i^0 \pi_i^1 \cdots \pi_i^{T-1}$. Une règle de décision locale $\pi_i^t \colon \Theta_i^t \mapsto A_i$ est une application des historiques locales $\theta_i^t \in \Theta_i^t$ vers les actions locales $\pi_i^t(\theta_i^t) \in A_i$.

Une *politique séparable* π est un tuple de n politiques locales, $\pi \equiv (\pi_1, \pi_2, \cdots, \pi_n)$, une pour chaque agent $i = 1, 2, \cdots, n$. Nous notons π_i la politique locale de l'agent i, et $\pi_{-i} \equiv (\pi_1, \cdots, \pi_{i-1}, \pi_{i+1}, \cdots, \pi_n)$ celle des ses co-équipiers. Une *règle de décision séparable* au pas de temps t est un tuple de n règles de décision locales au pas de temps t, $\pi^t \equiv (\pi_1^t, \pi_2^t, \cdots, \pi_n^t)$, une règle de décision locales pour chaque

agent. On peut considérer une politique séparable comme une séquence de règles de décision séparables, $\pi \equiv (\pi^0, \pi^1, \cdots, \pi^{T-1})$, une pour chaque pas de temps. Nous utilisons la notation $\pi^{t:l}$ (où $l \geq t$) pour représenter une séquence ordonnée de règles de décision séparables $(\pi^t, \cdots, \pi^l)$, à savoir une politique (partielle).

L'*état d'occupation* du processus sous le contrôle d'une politique séparable $\pi^{0:t-1}$ commençant à la distribution de probabilité initiale b^0 est la distribution de probabilité sur les états du système et les historiques jointes, $\boldsymbol{\eta}^t = b^0 \boldsymbol{P}^{\pi^{0:t-1}}$. Comme $\pi^{t:l}$, nous utiliserons la notation $\boldsymbol{\eta}^{t:l}$ au lieu de $(\boldsymbol{\eta}^t, \cdots, \boldsymbol{\eta}^l)$. Ici, l'état d'occupation $\boldsymbol{\eta}^t$ est un vecteur stochastique $(|S| \times |\Theta^t|) \times 1$, et $\boldsymbol{P}^{\pi^{t:l}}$ est une matrice stochastique $(|S| \times |\Theta^t|) \times (|S| \times |\Theta^{l+1}|)$, dans laquelle chaque entrée $\boldsymbol{P}^{\pi^{t:l}}[s^t, \theta^t; s^{l+1}, \theta^{l+1}]$ est la probabilité conditionnelle d'être dans (s^{l+1}, θ^{l+1}) si l'état initial et l'historique sont (s^t, θ^t) et les agents suivent la politique séparable $\pi^{t:l}$. Par souci de simplicité, nous employons la notation $\boldsymbol{P}^{\pi^t}$ au lieu de $\boldsymbol{P}^{\pi^{t:t}}$. Notons que, étant données une politique et une distribution initiale sur les états, l'état d'occupation peut être calculé à partir de $\boldsymbol{P}$ et $\boldsymbol{O}$ dans le Dec-POMDP.

3.2 Formulation et propriétés

La première propriété de l'état d'occupation est qu'il résume toute l'information passée des agents. La preuve peut être dérivée en développant la représentation de l'état d'occupation $\boldsymbol{\eta}^t = b^0 \boldsymbol{P}^{\pi^{0:t-1}}$. Alors que nous utilisons des historiques d'actions et d'observations, il peut aussi être montré que les états d'occupation dépendant juste des historiques d'observation et des états sont aussi suffisantes quand on utilise des politiques déterministes (dont une au moins est optimale) (Oliehoek, 2013).

Theorem 1
Le prochain état d'occupation dépend seulement de l'état d'occupation présent et de la règle de décision séparable :

$$\boldsymbol{\eta}^t \quad = \quad b^0 \prod_{l=0}^{t-1} \boldsymbol{P}^{\pi^l} \quad = \quad \boldsymbol{\eta}^{t-1} \boldsymbol{P}^{\pi^{t-1}}. \tag{1}$$

Ce théorème montre que les états d'occupation décrivent un processus de décision markovien : le *modèle occupancy MDP*. L'équation (1) définit la fonction de transition de ce modèle. Sans surprise, le modèle occupancy MDP est déterministe. Une représentation complète de ce modèle est obtenue en spécifiant aussi la fonction de récompense sur les états d'occupation. Nous notons $\boldsymbol{R}^{\pi^t}$ le vecteur de récompense associé à la règle de décision séparable π^t, dont les entrées sont définies par : $\forall s^t \in S, \theta^t \in \Theta^t$, $\boldsymbol{R}^{\pi^t}[s^t, \theta^t] = \boldsymbol{R}^{\pi^t(\theta^t)}[s^t]$. C'est-à-dire que la récompense pour avoir choisi la règle de décision dans un état d'occupation est définie comme l'espérance de la valeur des récompenses sur les états sous-jacents et les historiques.

Nous sommes maintenant prêts à définir formellement le modèle occupancy MDP qui correspond au modèle Dec-POMDP. Si $\triangle$ est l'espace des états d'occupation, et $\boldsymbol{A}$ est l'espace des règles de décision séparables, alors nous définissons le tuple $(\triangle, \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{R}, b^0)$ comme étant le modèle occupancy MDP correspondant au modèle Dec-POMDP.

Résoudre l'occupancy MDP détermine une politique π^* qui maximise l'espérance de récompense cumulée sur l'horizon T :

$$\pi^* \quad = \quad \arg\max_\pi \sum_{t=0}^{T-1} (\boldsymbol{R}^{\pi^t})^\top (b^0 \boldsymbol{P}^{\pi^{0:t-1}}).$$

Comme dans le modèle MDP traditionel, la t-ième fonction de valeur υ^t sous le contrôle d'une politique est définie comme suit : $\forall t = 0, 1, \cdots, T-1, \forall \boldsymbol{\eta}^t \in \triangle$

$$\upsilon^t(\boldsymbol{\eta}^t) \quad = \quad \max_{\pi^t \in \boldsymbol{A}} (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{t+1}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t}), \tag{2}$$

où $\upsilon^T(\cdot) = 0$. La quantité du côté droit de l'opérateur max de l'équation (2) est désignée comme la fonction de Q-valeur définie par : $\forall \boldsymbol{\eta}^t \in \triangle$, et $\forall \pi^t \in \boldsymbol{A}$, $Q^{\upsilon^{t+1}}(\boldsymbol{\eta}^t, \pi^t) = (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{t+1}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t})$. La quantité du côté droit de l'opérateur max de l'équation (2) est appelé fonction de Q-valeur définie par : $\forall \boldsymbol{\eta}^t \in \triangle$, et $\forall \pi^t \in \boldsymbol{A}$, $Q^{\upsilon^{t+1}}(\boldsymbol{\eta}^t, \pi^t) = (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{t+1}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t})$.

Notons que l'équation (2) est l'équation d'optimalité de Bellman pour le modèle occupancy MDP. Cela conduit au résultat suivant.

Corollary 1
Optimiser le modèle occupancy MDP est équivalent à optimiser le Dec-POMDP.

Cela vient en développant de l'équation (2) pour chaque pas du problème et en notant que, à cause de la mise-à-jour déterministe de $\boldsymbol{\eta}$, la politique résultante est une séquence de règles de décision séparables qui maximise la valeur espérée pour le Dec-POMDP.

Nous pouvons aussi montrer que la fonction de valeur est une fonction linéaire par morceaux et convexe (PWLC) de l'état d'occupation.

Theorem 2
La t-ième fonction de valeur $\upsilon^t \colon \triangle \mapsto \mathbb{R}$ solution de l'équation (2) est une fonction PWLC de l'état d'occupation.

Comme pour les belief MDP, la preuve se fait par récurrence étant donné que υ^T, $\boldsymbol{R}^{\pi^t}$ et $\boldsymbol{P}^{\pi^t}$ sont linéaires et que tous les opérateurs impliqués dans l'équation (2) préservent la propriété PWLC. Avec les théorèmes 1 et 2, on peut exploiter les avancées dans le domaine des belief MDP pour résoudre les occupancy MDP. Toutefois, étant donné qu'à la fois les états d'occupation et les règles de décision sont définis sur l'espace complet des historiques, la capacité de passer à l'échelle reste limitée.

Dans la suite de cet article, nous allons utiliser les Dec-POMDP et occupancy MDP de manière interchangeable.

4 Résoudre optimalement des occupancy MDP

Dans cette section, nous présentons d'abord les contours d'un nouvel algorithme pour résoudre les Dec-POMDP. Nous proposons aussi un cadre de réduction de la dimensionalité qui remplace l'espace –de haute dimensionalité– des historiques par un ensemble –de plus faible dimensionalité– de caractéristiques, sans risque de perdre (PWLC) la propriété PWLC de la fonction de valeur sur les états d'occupation dans l'ensemble de caractéristiques de basse dimensionalité ; et (LOSSLESS) la suffisance des états d'occupation dans l'ensemble de caractéristiques de basse dimensionalité.

4.1 L'algorithme FB-HSVI

Notre algorithme, appelé *feature-based heuristic search value iteration* (FB-HSVI), exploite la structure PWLC de la fonction de valeur dans un algorithme de recherche heuristique générique. De nombreux algorithmes pour résoudre des belief MDP tirent parti de la propriété PWLC de la fonction de valeur. Un tel algorithme de recherche heuristique qui préserve la capacité de converger théoriquement vers une solution ϵ-optimale est *heuristic search value iteration* (HSVI) (Smith & Simmons, 2004). Nous étendons HSVI pour résoudre l'occupancy MDP, mais d'autres algorithmes pour POMDP seraient aussi applicables.

HSVI, comme décrit dans l'algorithme 1, procède en créant des trajectoires, à l'aide de majorants et minorants de la fonction de valeur exacte, notés respectivement $\bar{\upsilon}$ et $\underline{\upsilon}$. Chaque telle trajectoire débute à la distribution de probabilité initiale b^0. HSVI exécute toujours la meilleure action d'après le majorant, et sélectionne ensuite le prochain état de croyance qui maximise l'écart entre minorant et majorant. Quand la trajectoire est terminée les états de croyance sont mis à jour dans l'ordre inverse. HSVI utilise les idées suivantes : *garanties sur la fonction de valeur* : les représentations pour le minorant $\underline{\upsilon}$ et le majorant $\bar{\upsilon}$ de la fonction de valeur maintiennent un encadrement valide de la fonction de valeur optimale υ^* ; *mises-à-jour ponctuelles* : les fonctions de valeurs sont améliorées en utilisant des opérations de mise-à-jour seulement sur un sous-ensemble du simplexe de croyance entier ; *recherche heuristique à base de trajectoires* : le majorant guide la recherche à travers des trajectoires d'états de croyance, en focalisant sur des régions où l'encadrement a besoin d'être amélioré.

FB-HSVI utilise le cadre algorithmique d'HSVI, héritant des caractéristiques ci-dessus. FB-HSVI diffère d'HSVI en de nombreux aspects, dont :

1. *mise-à-jour ponctuelle efficace* : nous utilisons une implémentation efficace de la mise-à-jour qui évite l'énumération systématique de toutes les règles de décision séparables (Dibangoye *et al.*, 2012, 2013) ;

Algorithm 1: L'algorithme FB-HSVI

initialiser $\underline{v}$, $\bar{v}$ et $\boldsymbol{\eta}^0 = b^0$
tant que $\bar{v}^0(\boldsymbol{\eta}^0) - \underline{v}^0(\boldsymbol{\eta}^0) > \epsilon$ **faire**
  Explorer($\boldsymbol{\eta}^0, \bar{v}, \underline{v}$)

Explorer($\boldsymbol{\eta}^t, \bar{v}, \underline{v}$)
si $\bar{v}^t(\boldsymbol{\eta}^t) - \underline{v}^t(\boldsymbol{\eta}^t) > \epsilon/T$ **alors**
  $\bar{\pi}^t \leftarrow \arg\max_{\pi^t \in \boldsymbol{A}} Q^{\bar{v}^{t+1}}(\boldsymbol{\eta}^t, \pi^t)$
  Mettre-à-jour ($\bar{v}^t, \boldsymbol{\eta}^t, \bar{\pi}^t$)
  Explorer(Compresser($\boldsymbol{\eta}^t \boldsymbol{P}^{\bar{\pi}^t}$), $\bar{v}, \underline{v}$))
  Mettre-à-jour ($\underline{v}, \boldsymbol{\eta}^t, \bar{\pi}^t$)

2. *représentations d'ensembles de caractéristiques de faible dimensionalité* : nous remplaçons l'espace d'historiques de forte dimensionalité par un ensemble de caractéristiques de basse dimensionalité (représentée avec l'opération Compresser dans l'algorithme) ; et
3. *mises-à-jour de l'encadrement* : nous tirons parti du déterminisme des occupancy MDP pour mettre à jour en avant le majorant, et en arrière le minorant.

Ces idées misent ensembles permettent à FB-HSVI de :

1. passer à l'échelle à des problèmes Dec-POMDP de tailles d'horizon sans précédent ;
2. de permettre une généralisation de la fonction de valeur sur des états d'occupation non visités ; et
3. d'accélérer la convergence de la fonction de valeur optimale.

4.2 Réduction de dimension linéaire et sans perte (*lossless*)

Dans ce qui suit, nous décrivons le cadre de réduction de dimension qui permet des représentations compactes. Il consiste en la définition de métriques pour déterminer si des historiques sont équivalentes. Ensuite, nous nous appuyons sur ce cadre pour introduire l'algorithme de sélection de caractéristiques que nous employons en pratique. Nous introduisons *finite-order feature selection* (FOFS) qui opère incrémentalement pendant l'exécution de l'algorithme. Les historiques de la prochaine étape peuvent alors être compressées par les méthodes discutées ci-dessous, fournissant un ensemble réduit d'historiques à utiliser dans l'algorithme. FOFS cherche à trouver des ensembles représentatifs d'historiques d'horizons plus courts pour remplacer les historiques possibles à une étape donnée sans aucune perte dans la qualité de la solution.

4.2.1 Utiliser des associations de caractéristiques

La *caractéristique* (locale) associée avec l'historique θ_i étant donné l'état d'occupation $\boldsymbol{\eta}^t$, notée ϕ_{θ_i}, est une fonction associant des réels à toutes les paires d'états et d'historiques des autres agents, de sorte que $\forall s, \forall \theta_{-i} \colon \phi_{\theta_i}[s, \theta_{-i}] = p(s, \theta_{-i}, \theta_i | \boldsymbol{\eta}^t)$. Chaque caractéristique ϕ_{θ_i} a un *label* l'associant avec une historique locale, ℓ_i ou $L(\theta_i)$. De plus, nous notons $\mathcal{C}_{\ell_i}$ le groupe d'historiques locales de label ℓ_i. Une façon de rendre explicite la relation entre les caractéristiques et les états d'occupation est de considérer l'occupation sous une forme matricielle, où les colonnes correspondent aux labels et les lignes correspondent des états et historiques des autres agents.

Pour sélectionner des caractéristiques, nous combinons deux critères. Le premier critère nous permet de regrouper ensemble des historiques locales qui sont *probabilistiquement équivalentes* et de les traiter comme une unique historique locale. Pour l'agent i, nous disons que deux historiques locales θ_i et θ_i' sont probabilistiquement équivalentes quand $\phi_{\theta_i} \propto \phi_{\theta_i'}$. Intuitivement, on peut voir que des historiques probabilistiquement équivalentes fournissent la même information sur l'environnement (y compris les autres agents) du point de vue d'un seul agent. Il a été montré que l'on ne perd rien à conditionner la sélection d'action sur le label $L(\theta_i)$ seul au lieu de θ_i (Oliehoek *et al.*, 2009). Cela permet une compression sans perte (lossless) des caractéristiques avec des valeurs générées comme suit : $\forall s, \forall \theta_{-i} \colon \phi_{\ell_i}[s, L(\theta_{-i})] = \sum_{\theta_i \in \mathcal{C}_{\ell_i}} \phi_{\theta_i}[s, \theta_{-i}]$.

Le second critère améliore la généralisation de la fonction de valeur sur les états d'occupation non visités tout en préservant les informations importantes. Cette approche peut être vue comme l'utilisation d'un "fenêtre glissante" sur l'historique d'information. On peut considérer les labels ℓ_i comme étant des historiques locales de longueurs quelconques dans l'ensemble Θ_i. Pour réduire l'espace des labels, nous

définissons un label d'ordre fini $\ell_{i,k}$ ou $L^k(\ell_i)$ comme étant les k actions et k observations les plus à droites (plus récentes) du label ℓ_i. De plus, nous notons $\mathcal{C}_{\ell_{i,k}}$ le groupe de labels avec le même label d'ordre fini $L^k(\ell_i)$. Cela permet une compression (éventuellement avec perte) de caractéristiques avec des valeurs définies comme suit : $\forall s, \forall \ell_{-i}$: $\phi_{\ell_{i,k}}[s, L^k(\ell_{-i})] = \sum_{\ell_i \in \mathcal{C}_{\ell_{i,k}}} \phi_{\ell_i}[s, \ell_{-i}]$. A l'évidence, il y a toujours un paramètre k entre 0 et $T-1$ tel que les caractéristiques $\phi_{\ell_{i,k}}$ sont sans perte (tel que $k = T-1$). Dans la suite, nous présentons un algorithme qui trouve un tel paramètre k.

4.2.2 Détails algorithmiques

Pour garantir que la réduction de dimension préserve la propriété PWLC, nous considérons des réductions de dimension linéaires (Boularias & Chaib-draa, 2008). Plus précisément, nous utilisons un regroupement fondé sur l'équivalence probabiliste et les labels d'ordre fini.

De manière à garantir la suffisance des états d'occupation, nous introduisons le concept de compression *sans perte selon Hajnal*. De manière informalle, cette propriété donne des indications sur comment sélectionner des caractéristiques qui préservent la capacité trouver au final une politique optimale. Pour définir cette propriété, nous avons besoin de majorer le regret de la recherche d'une politique optimale sur l'espace restreint au lieu de l'espace d'historiques de grande dimension original. Ce regret dépend de métriques de distance qui sont utilisées pour mesurer la *proximité* entre des états d'occupation exacts et approchés.

Soit $\mathbf{\Phi}$ la réduction de dimension linéaire $(|S| \times |\Theta|) \times (|S| \times |\tilde{\Theta}|)$ résultant des deux critères ci-dessus. Pour chaque état d'occupation $\boldsymbol{\eta}^t$, nous appelons état d'occupation approché $\widehat{\boldsymbol{\eta}}^t = \boldsymbol{\eta}^t \mathbf{\Phi} \mathbf{\Phi}^+$, où $\mathbf{\Phi}^+$ est un pseudo-inverse de $\mathbf{\Phi}$.

Définissons les métriques d et D comme suit : $\forall \boldsymbol{\eta}_x^t, \forall \boldsymbol{\eta}_y^t, d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t) = \sum_{s \in S} \sum_{\theta \in \Theta} \left(P(s, \theta | \boldsymbol{\eta}_x^t) - P(s, \theta | \boldsymbol{\eta}_y^t)\right)^+$, où $(o)^+ = o$ si $o > 0$ et zéro sinon, et $D(\boldsymbol{\eta}_x^{0:T-1}, \boldsymbol{\eta}_y^{0:T-1}) = \sum_{t=0}^{T-1} d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t)$. La métrique d, connue comme la *mesure de Hajnal*, a de nombreuses applications dans la théorie des chaînes ergodiques (Paz, 1971). De manière informelle, $d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t)$ est la *quantité* minimale de probabilité qui devrait être *ré-affectée* de manière à transformer l'état d'occupation exact $\boldsymbol{\eta}_x^t$ en l'état d'occupation $\boldsymbol{\eta}_y^t$. De manière similaire, $D(\boldsymbol{\eta}_x^{0:T-1}, \boldsymbol{\eta}_y^{0:T-1})$, appelé la *mesure de Hajnal variationelle* entre séquences d'états d'occupation, est la *quantité cumulée* de probabilité minimale par laquelle les deux séquences diffèrent.

Le théorème suivant majore le regret de la recherche d'une politique optimale sur l'espace restreint au lieu de l'espace d'historiques de grande dimension original. La preuve est similaire à la garantie de performance de l'algorithme de recherche de politique (théorème 1) (Bagnell *et al.*, 2003).

Theorem 3
Soient π et $\widehat{\pi}$ respectivement des politiques solutions du modèle d'occupancy MDP optimisé sur les états d'occupation exact et approché. Alors,

$$\upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \quad \geq \quad \upsilon^{\pi}(\boldsymbol{\eta}^0) - T \cdot D(\boldsymbol{\eta}^{0:T-1}, \widehat{\boldsymbol{\eta}}^{0:T-1}) ||r||,$$

où $||r|| = \max_{s,a} \boldsymbol{R}^a[s] - \min_{s,a} \boldsymbol{R}^a[s]$.

Preuve Si $\beta = \upsilon^{\pi}(\boldsymbol{\eta}^0) - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0)$, alors nous avons que :

$$\begin{aligned}
\beta &= \textstyle\sum_{t=0}^{T-1} (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \\
&= \textstyle\sum_{t=0}^{T-1} [(\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)] - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \\
&= \textstyle\sum_{t=0}^{T-1} [(\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{\widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t}) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)] \\
&= \textstyle\sum_{t=0}^{T-1} [\upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)]
\end{aligned}$$

Maintenant, soit $\beta^t = \upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)$, et α^t l'hyperplan dans la fonction de valeur $\upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}$ qui est maximale en $\boldsymbol{\eta}^t$, et $\widehat{\alpha}^t$ l'hyperplan dans $\upsilon^{\widehat{\pi}^{t:T-1}}$ qui est maximal en $\widehat{\boldsymbol{\eta}}^t$. Alors, étant donné que $\widehat{\alpha}^t$ est maximal en $\widehat{\boldsymbol{\eta}}^t$, alors nous avons que :

$$\begin{aligned}
\beta^t &= (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t \\
&= (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t + (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t - (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \\
&\leq (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t + (\widehat{\alpha}^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t - (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \\
&= (\alpha^t - \widehat{\alpha}^t) \cdot (\boldsymbol{\eta}^t - \widehat{\boldsymbol{\eta}}^t) \;\leq\; T d(\boldsymbol{\eta}^t, \widehat{\boldsymbol{\eta}}^t) ||r||.
\end{aligned}$$

La première égalité est vraie parce que $(\widehat{\alpha}^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \geq (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t$, où $\widehat{\boldsymbol{\eta}}^t$ et $\boldsymbol{\eta}^t$ doit être (dé-)compressé sans perte pour correspondre respectivement à $\widehat{\alpha}^t$ et α^t. La dernière inégalité est démontrée en utilisant à la fois l'inégalité de Hölder et la mesure de Hajnal. En remplaçant β^t dans la dernière expression de β et la définition de D, on obtient le résultat. □

Ce majorant (théorème 3) dépend de la *proximité*, au sens de Hajnal, entre les états d'occupation exacts et approchés. Quand la réduction de dimension linéaire et sans perte, c'est-à-dire que $D(\boldsymbol{\eta}^{0:T-1}, \widehat{\boldsymbol{\eta}}^{0:T-1}) = 0$, le majorant implique qu'il existe une politique dans l'espace restreint qui atteint une performance aussi bonne que n'importe quelle politique dans l'espace d'historiques de grande dimension.

Il est aisé de vérifier si une réduction de dimension linéaire est avec ou sans perte de Hajnal. En fait, vérifier cette propriété sur la séquence entière $\boldsymbol{\eta}^{0:T-1}$ des états d'occupation consiste en vérifier que $d(\boldsymbol{\eta}^t, \widehat{\boldsymbol{\eta}}^t) = 0$ est vrai pour tout état d'occupation $\boldsymbol{\eta}^t$ de cette séquence. Nous appelons cela une *compression locale sans perte de Hajnal*. Nous sommes maintenant prêts à introduire notre algorithme de sélection de caractéristiques, à savoir *finite-order feature selection* (FOFS), qui sélectionne automatiquement les caractéristiques informatives sur la base de la compression sans perte de Hajnal.

FOFS compresse d'abord les états d'occupation à l'aide de l'équivalence probabiliste comme dans (Oliehoek *et al.*, 2009). Etant donné l'état d'occupation $\boldsymbol{\eta}^t$, pour chaque agent i, nous remplaçons les historiques avec les labels associés, remplaçant ainsi les caractéristiques ϕ_{θ_i} par la caractéristique correspondante ϕ_{ℓ_i}. Soit $\tilde{\boldsymbol{\eta}}^t$ l'état d'occupation correspondant. Clairement, il n'y a pas de risque de perdre soit la propriété (PWLC) soit la propriété (LOSSLESS) en planifiant sur $\tilde{\boldsymbol{\eta}}^t$ au lieu de $\boldsymbol{\eta}^t$ puisque cette compression est sans perte et linéaire. Toutefois, $\tilde{\boldsymbol{\eta}}^t$ est toujours défini sur un ensemble de labels de grande dimension. Cela limite de manière significative la généralisation de la fonction de valeur sur des états d'occupation non visités et souligne la nécessité d'une transformation réduisant la dimensionalité.

FOFS compresse ensuite l'état d'occupation $\tilde{\boldsymbol{\eta}}^t$ en $\widehat{\boldsymbol{\eta}}^t$ en utilisant des labels d'ordre fini, en commençant par $k = 0$. Cette compression est linéaire de sorte qu'elle préserve la propriété (PWLC), mais peut générer des pertes. Quand la compression locale est sans perte de Hajnal, c'est-à-dire que $\forall s, \forall \ell\colon P(s, \ell|\tilde{\boldsymbol{\eta}}^t) - P(s, L^k(\ell)|\widehat{\boldsymbol{\eta}}^t) = 0$, alors FOFS retourne $\widehat{\boldsymbol{\eta}}^t$ avec des labels ℓ remplacés par des labels d'ordre fini $L^k(\ell)$. Sinon il recalcule $\widehat{\boldsymbol{\eta}}^t$ avec le paramètre $k = k + 1$. L'algorithme itère jusqu'à ce que la compression locale soit sans perte de Hajnal. Dans le pire cas, FOFS se termine avec le paramètre $k = t - 1$, et retourne une représentation sans perte $\widehat{\boldsymbol{\eta}}^t$ de l'état d'occupation original $\boldsymbol{\eta}^t$. When $k < t - 1$, l'état d'occupation $\widehat{\boldsymbol{\eta}}^t$ est défini comme la distribution de probabilité sur les états et les historiques de longueur l avec $l \leq k < t - 1$, de sorte que la compression est une réduction de dimension.

Dans les deux cas, pour compresser un état d'occupation pour un critère donné, FOFS procède en regroupant des historiques (ou labels) d'un agent à la fois en supposant que les labels des autres agents sont fixés (initialement comme des extensions complètes des historiques des étapes précédentes, et ensuite comme des versions compressées). Il répète cette procédure pour chaque alternativement jusqu'à ce qu'aucun regroupement ne puisse avoir lieu.

5 Expérimentations

Nous avons exécuté FB-HSVI sur une machine Mac OSX avec un Intel Dual-Core 2,4GHz et 2Go de RAM disponible. Nous avons résolu les problèmes de mise-à-jour par optimisation sous contraintes avec le solveur toulbar2. Nous avons initialisé le majorant avec la fonction de valeur du MDP sous-jascent, et le minorant comme la fonction de valeur de la politique aveugle. Nous évaluons trois variantes de FB-HSVI(ρ) : la première variante, FB-HSVI(0), est FB-HSVI sans compression ni mise-à-jour efficace ; la seconde variante, FB-HSVI(1) utilise $k = \infty$ (historiques complètes), mais incorpore les mises-à-jour efficaces ; et FB-HSVI(2) utilise à la fois les mises-à-jour efficaces et les historiques compressées avec FOFS.

Nous avons évalué nos algorithmes sur cinq problèmes de référence de la littérature : *multi-agent tiger*, *recycling-robot*, *grid-small*, *cooperative box-pushing*, et *mars rovers*. Ce sont les plus grands et plus difficiles problèmes de la littérature. Pour chacun d'eux, nous avons comparé nos algorithmes avec des solveurs exacts de l'état de l'art : GMAA*-ICE (Spaan *et al.*, 2011), IPG (Amato *et al.*, 2009), MILP (Aras & Dutech, 2010), et LPC (Boularias & Chaib-draa, 2008). IPG et LPC effectue une programmation dynamique ; GMAA*-ICE effectue une recherche heuristique ; et MILP est une méthode de programmation linéaire entière mixte. Les résultats pour GMAA*-ICE (fournis par Matthijs Spaan), IPG, MILP et LPC ont été obtenus sur différentes machines. A cause de cela, les résultats en termes de temps de calcul ne sont pas directement comparables entre méthodes, mais ne diffèrent probablement que par un petit facteur constant.

Les résultats peuvent être vus dans la table 1. Pour chaque algorithme nous rapportons le temps de calcul, ce qui inclue le temps pour calculer les éventuelles valeurs heuristiques. Nous rapportons aussi l'espérance de la récompense cumulée ϵ-optimale $v_\epsilon(\boldsymbol{\eta}^0)$ (où $\epsilon = 0.01$) à l'état d'occupation initial. En outre, nous rap-

T	MILP	LPC	IPG	ICE	**FB-HSVI**(ρ)			$v_\epsilon(\boldsymbol{\eta}^0)$
					0	1	2	
	The multi-agent tiger problem ($\|S\|=2, \|Z\|=4, \|A\|=9, K=3$)							
2	–	0.17	0.32	0.01	0.05	0.03	**0.03**	−4.00
3	4.9	1.79	55.4	0.01	2.17	0.06	**0.40**	5.2908
4	72	534	2286	108	9164	2.66	**1.36**	4.8027
5				347		22.2	**9.65**	7.0264
6						171.3	**24.42**	10.381
7							**33.11**	**9.9935**
8							**41.21**	**12.217**
9							**58.51**	**15.572**
10							**65.57**	**15.184**
	The recycling-robot problem ($\|S\|=4, \|Z\|=4, \|A\|=9, K=1$)							
2	–	–	0.30	36	0.03	0.02	**0.01**	7.000
3	–	–	1.07	36	0.05	0.47	**0.10**	10.660
4	–	–	42.0	72	0.85	0.65	**0.30**	13.380
5	–	–	1812	72	1.52	0.87	**0.34**	16.486
10					5.06	2.83	**0.52**	31.863
30					62.8	37.9	**1.13**	93.402
70						78.1	**2.13**	216.47
100						259	**2.93**	**308.78**
	The grid-small problem ($\|S\|=16, \|Z\|=4, \|A\|=25, K=4$)							
2	0.65	–	0.18	36	116.1	0.1	**0.1**	0.9100
3	1624	–	4.09	1512	3024	6.09	**0.73**	1.5504
4		–	77.4	242605		12.85	**1.39**	2.2415
5						148.2	**2.40**	2.9704
6						319.8	**7.12**	3.7171
7						645.1	**58.25**	**4.4657**
8							**65.12**	**5.2319**
9							**71.38**	**5.9878**
	The cooperative box-pushing problem ($\|S\|=100, \|Z\|=16, \|A\|=25, K=3$)							
2	–	–	1.07	36	0.1	0.1	**0.1**	17.600
3	–	–	6.43	540	2026	0.64	**0.457**	66.081
4	–	–	1138	2596		3.16	**0.622**	98.593
5						16.72	**5.854**	**107.72**
6						274.6	**70.67**	**120.67**
7						462.4	**74.40**	**156.42**
8						751.5	**95.38**	**191.22**
9							**105.7**	**208.19**
10							**168.5**	**220.45**
	The mars-rovers problem ($\|S\|=256, \|Z\|=81, \|A\|=36, K=3$)							
2	–	–	83	1.0	0.21	0.09	**0.10**	5.80
3	–	–	389	1.0	2.84	0.21	**0.23**	9.38
4				103	104.2	1.73	**0.47**	10.18
5						6.38	**0.82**	**13.26**
6						8.16	**3.97**	**18.62**
7						11.13	**5.81**	**20.90**
8						35.49	**22.8**	**22.47**
9						57.47	**26.5**	**24.31**
10						316.2	**62.7**	**26.31**

TABLE 1 – Expériences comparant les temps de calcul (en secondes) pour tous les solveurs exacts. Les dépassements de temps limite ($1000s$) sont indiqués par " ", et "–" indique des valeurs inconnues.

portons aussi K, la valeur maximale de k trouvée par FOFS pour chaque problème (représentant la longueur d'historique maximale utilisée).La table 1 montre clairement que FB-HSVI(2) amène une amélioration significative sur les solveurs de l'état de l'art ; pour tous les problèmes nous fournissons des résultats pour des horizons plus longs que tous ceux résolus dans le passé (entrée en gras).

Il y a plusieurs raisons pour les performances de FB-HSVI(2) :

– d'abord, nous cherchons dans l'espace des politiques associations des actions aux caractéristiques en basse dimension, alors que les autres solveurs cherchent dans l'espace des politiques associant des actions aux historiques ;
– nous utilisons une fonction de valeurs associant des réels aux états d'occupation permettant de généraliser cette fonction de valeur sur des états d'occupation non visités alors que les autres solveurs utilisent des fonctions de valeur associant des réels à des politiques partielles ;
– enfin, nous remplaçons la mise-à-jour en force brute par une approche efficace par optimisation sous contraintes.

Pour mieux comprendre FB-HSVI, nous comparons les trois variantes $\rho = 0, 1, 2$ présentées plus tôt. Pour $\rho = 0$, nous dépassons vite le temps autorisé parce que nous énumérons explicitement tous les règles de décision séparables. Pour $\rho = 1$, nous passons à l'échelle pour de plus grands horizons mais la convergence ralentit parce que la fonction de valeur est définie sur l'espace des historiques. Pour $\rho = 2$, nous améliorons la généralisation de la fonction de valeur sur des états d'occupation non visités à l'aide des labels d'ordre fini. Cela évite des mises-à-jour inutiles et accélère la convergence.

6 Conclusion

Cet article explore une nouvelle théorie et de nouveaux algorithmes pour résoudre des Dec-POMDP. Cette théorie ce construit sur une nouvelle transformation des Dec-POMDP vers des MDP déterministes sur un espace d'états continu, généralisant des résultats théoriques et algorithmiques antérieurs. En particulier, nous démontrons deux résultats importants :

1. la distribution sur à la fois les états et les historiques, à savoir les états d'occupations, sont suffisantes pour la planification optimale dans les Dec-POMDP ;
2. la fonction de valeur est linéaire par morceaux et convexe sur les états d'occupation.

Avec cette nouvelle formulation, les méthodes POMDP peuvent, pour la première fois, être directement appliquées aux Dec-POMDP. Cela nous permet de ne plus maintenir une représentation de politique explicite et d'exploiter de la structure dans la fonction de valeur. Nous décrivons aussi un nouvel algorithme pour cette représentation, l'étendant à des grands espaces d'états et d'action à l'aide d'une réduction de dimension et d'optimisation sous contraintes. Cet algorithme, appelé *feature-based heuristic search value iteration* ou FB-HSVI, s'avère passer à l'échelle pour de grands horizons de planification, réduisant les temps de calcul de plusieurs ordres de grandeur par rapport aux approches précédentes.

Dans le futur, nous prévoyons d'explorer l'application de cette théorie et de cet algorithme à des sous-classes de Dec-POMDP et de plus grands groupes d'agents. Par exemple, nous avons déjà montré que les états d'occupation sur les états seuls (et non les historiques de agents) peuvent être utilisés dans les Dec-MDP à transitions et observations indépendantes de manière à grandement augmenter la capacité de passage à l'échelle (Dibangoye *et al.*, 2012, 2013). Nous souhaitons explorer l'utilisation d'états d'occupation sur des historiques d'observation (plutôt que des historiques action-observation), lesquelles ont été démontrée suffisantes (comme les historiques action-observation) simultanément à ce travail Oliehoek (2013). De plus, nous pensons que les états d'occupation avec des modèles graphiques pourraient aider à exploiter la localité de l'interaction entre agents statistiquement (comme dans les ND-POMDP (Nair *et al.*, 2005; Kumar & Zilberstein, 2009)) ou dynamiquement (comme dans (Canu & Mouaddib, 2011)). Par ailleurs, la capacité de passage à l'échelle de FB-HSVI est encourageante et nous chercherons des méthodes supplémentaires pour calculer automatiquement des représentations compactes.

7 Remerciements

Nous souhaitons remercier Matthijs Spaan d'avoir fourni les résultats pour GMAA*-ICE, ainsi que Frans Oliehoek et les relecteurs pour leurs commentaires constructifs. Recherche soutenue en partie par le projet AFOSR MURI #FA9550-09-1-0538.

Références

AMATO C., DIBANGOYE J. S. & ZILBERSTEIN S. (2009). Incremental policy generation for finite-horizon DEC-POMDPs. In *ICAPS*.

ARAS R. & DUTECH A. (2010). An investigation into mathematical programming for finite horizon decentralized POMDPs. *JAIR*, **37**, 329–396.

BAGNELL J. A., KAKADE S., NG A. & SCHNEIDER J. (2003). Policy search by dynamic programming. In *NIPS*, volume 16.

BERNSTEIN D. S., AMATO C., HANSEN E. A. & ZILBERSTEIN S. (2009). Policy iteration for decentralized control of Markov decision processes. *JAIR*, **34**, 89–132.

BERNSTEIN D. S., GIVAN R., IMMERMAN N. & ZILBERSTEIN S. (2002). The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, **27**(4).

BOULARIAS A. & CHAIB-DRAA B. (2008). Exact dynamic programming for decentralized POMDPs with lossless policy compression. In *ICAPS*, p. 20–27.

CANU A. & MOUADDIB A.-I. (2011). Collective decision under partial observability - a dynamic local interaction model. In *IJCCI (ECTA-FCTA)*, p. 146–155.

DE FARIAS D. P. & VAN ROY B. (2003). The linear programming approach to approximate dynamic programming. *Operations Research*, **51**(6), 850–865.

DIBANGOYE J. S., AMATO C. & DONIEC A. (2012). Scaling up decentralized MDPs through heuristic search. In *UAI*, p. 217–226.

DIBANGOYE J. S., AMATO C., DONIEC A. & CHARPILLET F. (2013). Producing efficient error-bounded solutions for transition independent decentralized MDPs. In *AAMAS*.

DIBANGOYE J. S., MOUADDIB A.-I. & CHAIB-DRAA B. (2011). Toward error-bounded algorithms for infinite-horizon DEC-POMDPs. In *AAMAS*, p. 947–954.

HANSEN E. A., BERNSTEIN D. S. & ZILBERSTEIN S. (2004). Dynamic programming for partially observable stochastic games. In *AAAI*, p. 709–715.

HOWARD R. A. (1960). *Dynamic Programming and Markov Processes*. The M.I.T. Press.

KUMAR A. & ZILBERSTEIN S. (2009). Constraint-based dynamic programming for decentralized POMDPs with structured interactions. In *AAMAS*, p. 561–568.

NAIR R., VARAKANTHAM P., TAMBE M. & YOKOO M. (2005). Networked distributed POMDPs : A synthesis of distributed constraint optimization and POMDPs. In *AAAI*, p. 133–139.

OLIEHOEK F. A. (2013). Sufficient plan-time statistics for decentralized POMDPs. In *IJCAI*.

OLIEHOEK F. A., SPAAN M. T. J., AMATO C. & WHITESON S. (2013). Incremental clustering and expansion for faster optimal planning in Dec-POMDPs. *JAIR*, **46**, 449–509.

OLIEHOEK F. A., SPAAN M. T. J. & VLASSIS N. A. (2008). Optimal and approximate Q-value functions for decentralized POMDPs. *JAIR*, **32**, 289–353.

OLIEHOEK F. A., WHITESON S. & SPAAN M. T. J. (2009). Lossless clustering of histories in decentralized POMDPs. In *AAMAS*, p. 577–584.

PAZ A. (1971). *Introduction to probabilistic automata (Computer science and applied mathematics)*. Orlando, FL, USA : Academic Press, Inc.

POWELL W. B. (2007). *Approximate Dynamic Programming : Solving the Curses of Dimensionality (Wiley Series in Probability and Statistics)*. Wiley-Interscience.

SHANI G., PINEAU J. & KAPLOW R. (2012). A survey of point-based POMDP solvers. *JAAMAS*, p. 1–51.

SMALLWOOD R. D. & SONDIK E. J. (1973). The optimal control of partially observable Markov decision processes over a finite horizon. *Operations Research*, **21**(5), 1071–1088.

SMITH T. & SIMMONS R. (2004). Heuristic search value iteration for POMDPs. In *UAI*, p. 520–527.

SPAAN M. T. J., OLIEHOEK F. A. & AMATO C. (2011). Scaling up optimal heuristic search in Dec-POMDPs via incremental expansion. In *IJCAI*, p. 2027–2032.

SZER D., CHARPILLET F. & ZILBERSTEIN S. (2005). MAA* : A heuristic search algorithm for solving decentralized POMDPs. In *UAI*, p. 568–576.