Résolution de Dec-POMDP à horizon infini à l'aide de contrôleurs à états finis dans JESP

Yang You[1] Vincent Thomas[1] Francis Colas[1] Olivier Buffet[1]

[1] Université de Lorraine, INRIA, CNRS, LORIA, F-54000 Nancy, France

prénom.nom@loria.fr

Résumé

Cet article s'intéresse à la résolution de problèmes de planification collaborative formalisés comme des POMDP décentralisés (Dec-POMDP) en cherchant des équilibres de Nash, c'est-à-dire des situations dans lesquelles la politique de chaque agent est une meilleure réponse aux politiques (fixes) des autres agents. Alors que l'algorithme joint equilibrium-based search for policies *(JESP) fait ceci dans le cadre d'horizons finis en se reposant sur des arbres-politiques, nous proposons ici d'adapter JESP aux Dec-POMDP à horizon infini en représentant les politiques des agents par des contrôleurs à états finis. Dans cet article, nous (1) expliquons comment transformer un Dec-POMDP avec $N-1$ contrôleurs à états finis fixés en un POMDP à horizon infini dont la solution est une meilleure réponse du $N^{ième}$ agent ; (2) proposons une variante de JESP, appelée inf-JESP, reposant sur cette transformation pour résoudre des Dec-POMDP à horizon infini ; (3) introduisons des initialisations heuristiques pour JESP visant à conduire à de bonnes solutions ; et (4) conduisons une évaluation empirique de notre approche sur des bancs d'essais de l'état de l'art.*

Mots Clef

Dec-POMDP, JESP, contrôleurs à états finis, équilibre de Nash

Abstract

This paper looks at solving collaborative planning problems formalized as Decentralized POMDPs (Dec-POMDPs) by searching for Nash equilibria, i.e.*, situations where each agent's policy is a best response to the other agents' (fixed) policies. While the* joint equilibrium-based search for policies *(JESP) algorithm does this in the finite-horizon setting relying on policy trees, we propose here to adapt JESP to infinite-horizon Dec-POMDPs by using Finite State Controller policy representations. In this article, we (1) explain how to turn a Dec-POMDP with $N-1$ fixed finite state controllers into an infinite-horizon POMDP whose solution is a best response of the N^{th} agent ; (2) propose a JESP variant based on this transformation, called inf-JESP, for solving infinite-horizon Dec-POMDPs ; (3) introduce heuristic initializations for JESP aiming at deterministically leading to good solutions ; and (4) conduct experiments on state-of-the-art benchmark problems to evaluate our approach.*

Keywords

Dec-POMDP, JESP, finite state controllers, Nash equilibrium

1 Introduction

Les problèmes de décisions markoviens partiellement observables décentralisés (Dec-POMDP) visent à représenter des problèmes de décision séquentielle multi-agents dans lesquels l'objectif est de concevoir simultanément les politiques de plusieurs agents de sorte que leur exécution décentralisée collecte le plus de récompenses cumulées possibles. Pour être exécutable, chaque politique d'agent doit ainsi reposer sur son historique individuel, à savoir la séquence des ses actions et observations passées.

Il a été démontré que résoudre un Dec-POMDP à horizon fini est de complexité NEXP au pire cas [5], même pour deux agents, ce qui limite l'efficacité possible des solveurs optimaux pour des Dec-POMDP génériques. Les principales difficultés de la résolution de Dec-POMDP sont liées à deux faits : (i) l'état du système évolue selon les actions de tous les agents ; et (ii) l'action effectuée par chaque agent ne peut dépendre que de son propre historique. Ainsi, toutes les politiques sont interdépendantes et chaque agent a besoin de considérer les historiques potentiellement rencontrés par les autres agents, ainsi que leurs politiques, pour prendre une décision de manière optimale.

Pour contourner ces interdépendances pendant le processus d'optimisation, l'algorithme JESP (*joint equilibrium-based search for policies*) [17] recherche des solutions sous la forme d'équilibres de Nash, la politique de chaque agent étant une *meilleure réponse* aux politiques des autres agents. Il accomplit cela pour des problèmes à horizon fini (en employant des représentations arborescentes des politiques) en optimisant individuellement la politique de chaque agent l'un après l'autre, les politiques des autres

agents étant figées, jusqu'à convergence, c'est-à-dire en répétant le processus jusqu'à ce qu'aucune amélioration ne soit possible.

Cet algorithme fait toutefois face à deux inconvénients. Premièrement, les équilibres de Nash sont des optima locaux et non globaux. Deuxièmement, il ne traite que des problèmes à horizon fini et fournit des arbres politiques, lesquels ne peuvent pas être employés avec des horizons infinis.

Dans cet article, nous abordons ce second inconvénient et proposons une façon de résoudre les problèmes à horizon infini à travers une approche JESP appelée *inf-JESP* pour "*infinite-horizon JESP*". Son point de départ est de reposer non sur des arbres politiques, mais sur des contrôleurs à états finis (FSC pour "*Finite State Controler*"), une représentation courante pour les POMDP à horizon infini, ce qui, à notre connaissance n'a pas été fait jusque là. Dans ce but, nous proposons une manière de concevoir le POMDP rencontré par un agent quand on fige les politiques FSC des autres agents et qu'on les combine avec le modèle Dec-POMDP. De là, nous utilisons un solveur de POMDP pour trouver un FSC qui soit une meilleure réponse individuelle, et intégrons cette étape dans un schéma algorithmique de type JESP.

Pour être plus précis, nous étendons et améliorons la méthode JESP sur trois aspects : (1) les représentations arborescentes sont remplacées par des FSC pour résoudre les problèmes à horizon infini et pour concevoir chaque POMDP intermédiaire sans avoir à considérer des distributions sur les historiques possibles des autres agents (mais seulement sur les nœuds internes de leurs FSC) ; (2) des solveurs de POMDP de l'état de l'art sont utilisés à chaque étape (dans ce travail, nous utilisons SARSOP [14]) pour approximer la fonction de valeur optimale d'un POMDP et, sur la base du travail de GRZEŚ et coll. [10], un FSC est dérivé de l'approximation résultante ; (3) une nouvelle méthode d'initialisation heuristique pour JESP est proposée, dans laquelle des FSC individuels sont extraits d'une politique jointe obtenue en résolvant un POMDP multiagent plus simple (MPOMDP) [23] dans lequel tous les agents sont contrôlés par une entité unique qui a accès à toutes les observations reçues. Ces FSC individuels extraits peuvent être utilisés pour initialiser notre algorithme et fournir des solutions de façon déterministe.

En suivant ces directions, nous espérons que l'emploi de FSC aidera à concevoir des politiques à la fois dotées de représentations compactes, mais aussi plus faciles à exécuter et à comprendre (comme présenté dans [10]) que le JESP classique

Dans la section 2, nous discutons des travaux connexes sur les contrôleurs à états finis et les méthodes de résolution de Dec-POMDP existantes. La section 3 définit formellement les Dec-POMDP et les FSC. Dans la section 4, nous (1) expliquons comment combiner des FSC avec un Dec-POMDP pour générer le POMDP requis à chaque itération de inf-JESP ; (2) puis décrivons l'algorithme inf-JESP global dédié à la résolution de Dec-POMDP à horizon infini ; et (3) présentons une méthode d'initialisation heuristique pour les algorithmes de la famille de JESP. Enfin, des résultats expérimentaux et leur analyse sont présentés en section 5 avant de conclure en section 6.

2 Travaux connexes

Résolution de Dec-POMDP Un premier type d'approche pour résoudre des Dec-POMDP consiste à transformer le Dec-POMDP en un problème de plus court chemin déterministe, comme le fait Multi-Agent A* (MAA*) [25], ou même un processus de décision markovien déterministe, en utilisant une statistique suffisante contenant des informations pertinentes concernant l'état du processus. Ainsi, plusieurs approches ont été proposées pour résoudre des Dec-POMDP en utilisant les *états d'occupation* définis comme la distribution de probabilité sur l'état réel du système et les possibles historiques d'actions et observations passées des différents agents. FB-HSVI [9] emploie le schéma algorithmique HSVI [24], c'est-à-dire une recherche heuristique dans l'espace des états d'occupation, permettant ainsi de dériver une politique jointe ϵ-optimale en temps fini. Reposant sur la même idée, MACDERMED et ISBELL [15] ont proposé un algorithme appelé PBVI-BB pour transformer un Dec-POMDP en un POMDP avec un nombre fini d'états de croyance qui peut être résolu par des méthodes à base de points. Le principal intérêt de ces approches est qu'elles peuvent donner des solutions aussi proches de l'optimum que désiré, mais la taille de l'espace d'état pour le problème correspondant explose quand le nombre d'actions et d'observations croît, nécessitant d'importantes ressources computationnelles pour obtenir une solution proche de l'optimum.

Un second type d'approche consiste à explorer l'espace des politiques jointes en optimisant simultanément les politiques paramétrées de tous les agents. Pour des Dec-POMDP à horizon infini, ces approches représentent les politiques (à mémoire bornée) de chaque agent de manière compacte comme un FSC, rendant possible de rechercher directement dans l'espace des FSC à nombre de nœuds fini. Profitant de cette représentation, AMATO, BERNSTEIN et ZILBERSTEIN [2] proposent d'optimiser directement les paramètres des FSC en représentant le problème Dec-POMDP comme un problème de programmation non-linéaire (NLP), employant des outils de NLP pour calculer la politique jointe optimale. Cette approche a été améliorée en employant des FSC sous la forme de machines de Mealy à la place de machines de Moore, dérivant ainsi le solveur MealyNLP [3]. D'autres approches [13, 20, 21] utilisent des techniques *Expectation-Maximization* en reformulant le problème d'optimisation Dec-POMDP en un problème d'inférence consistant à estimer les meilleurs paramètres des FSC afin de maximiser la probabilité de générer des récompenses, ce qui conduit à des politiques jointes sous-optimales (du fait de limitations d'Expectation-Maximization). En particulier, PeriEM

[21] travaille avec des FSC périodiques (comme Peri, un algorithme d'amélioration de FSC). Comme les descentes de gradient pour POMDP (i) s'étendent naturellement aux Dec-POMDP [26], et (ii) peuvent servir à optimiser les paramètres d'un FSC [16, 1], ils permettraient aussi d'optimiser plusieurs FSC dans un cadre multi-agent.

Un troisième type d'approche se concentre sur des heuristiques pour concevoir une politique jointe. C'est le cas de JESP (joint equilibrium-based search for policies), proposé par NAIR et coll. [17]. Celui-ci repose sur l'observation qu'une politique jointe optimale est à l'évidence un équilibre de Nash, puisqu'aucun agent ne peut, seul, améliorer la valeur de cette politique jointe. Même si le contraire n'est pas vrai, JESP propose de chercher des équilibres de Nash en optimisant la politique d'un agent à la fois, considérant que les politiques des autres agents sont figées, et ce, jusqu'à ce qu'aucune amélioration ne soit possible. Cette optimisation est réalisée en résolvant à chaque itération un problème de décision mono-agent modélisé comme un processus de décision markovien partiellement observable (POMDP) combinant la dynamique du Dec-POMDP et les politiques, connues, des autres agents. Dans ce problème, l'agent à optimiser n'a accès qu'à ses propres observations et doit raisonner, à travers des distributions de probabilité, sur l'état caché du système et les historiques d'action et d'observation des autres agents. En transformant un problème de décision séquentiel multi-agent en une séquence de plusieurs problèmes de décision mono-agents à résoudre, JESP réduit fortement la complexité de la résolution, mais se restreint à trouver des optima locaux. De plus, dans l'algorihtme JESP original, la résolution de POMDP est accomplie soit par une recherche exhaustive, soit par une approche par programmation dynamique, les politiques étant représentées par des arbres politiques. Dans ce cas, les historiques des autres agents devraient être représentées comme une variable cachée du POMDP mono-agent, limitant JESP à des problèmes à horizon fini tout en requérant un espace d'états dont la taille croît exponentiellement avec l'horizon considéré. Nous proposons d'adapter JESP à des problèmes à horizon infini en utilisant une représentation FSC pour la politique de chaque agent.

Parmi les approches de résolution de Dec-POMDP, Dec-BPI (*Decentralized Bounded Policy Iteration*) [6, 7] est étroitement lié à notre proposition. Il repose sur une représentation par FSC stochastiques complétée par un mécanisme de corrélation permettant aux agents d'avoir accès aux mêmes nombres pseudo-aléatoires, et propose une approche similaire à JESP en se concentrant sur l'amélioration des paramètres d'un nœud de FSC (ou du mécanisme de corrélation) à la fois en utilisant un programme linéaire. À l'inverse, nous proposons de ne pas modifier qu'un FSC localement mais, à chaque itération d'inf-JESP, de dériver un FSC complètement nouveau solution induit par les FSC des autres agents. Notons que, comme JESP, Dec-BPI ne peut fournir qu'un équilibre de Nash sans aucune garantie de trouver une solution globalement optimale. Un des principaux avantages de Dec-BPI est, comme la version théorique de notre algorithme, de se concentrer sur des FSC de taille fixe, alors que la version que nous implémentons relâche cette contrainte, au risque d'obtenir de grands FSC (voir sections 3 et 5).

Représentation et évaluation des politiques FSC Puisque JESP repose sur la résolution de POMDP, cette section discute de contributions employées dans ce travail, en se concentrant sur les représentations FSC pour les politiques POMDP à horizon infini. De tels FSC sont compacts, et ont ouverts de nouvelles directions de recherche pour la résolution de POMDPs en optimisant directement leurs paramètres.

Une des contributions principales dans ce contexte est itération sur les politiques pour POMDP [11, 12]. Dans ces articles, HANSEN propose un algorithme d'itération sur les politiques reposant sur des politiques FSC (ainsi qu'une recherche heuristique). Suivant le schéma algorithmique d'itération sur les politiques, itération sur les politiques pour POMDPs repose sur deux étapes : (i) une étape d'évaluation consistant à estimer la valeur de la politique FSC courante, et (ii) une étape d'amélioration mettant à jour la politique FSC à l'aide de cette évaluation à travers l'addition de nouveaux nœuds et l'élagage ou le remplacement de nœuds dominés. On notera que, si l'étape d'amélioration de la politique peut être très coûteuse en temps de calcul, l'étape d'évaluation de politique peut être effectuée très efficacement en résolvant un système d'équations linéaires [11]. Dans le présent article, nous réutilisons la même technique d'évaluation des FSC obtenus à chaque itération de JESP.

Notons que des descentes de gradients peuvent aussi être employées pour optimiser les paramètres d'un FSC (stochastique) de taille donnée, comme le font MEULEAU et coll. ABERDEEN [16, 1].

Plus récemment, GRZEŚ et coll. [10] ont expliqué comment dériver un FSC à partir d'une fonction de valeur et fournit une méthode de compression de contrôleur par suppression des nœuds redondants ou dominés. Cela permet d'utiliser des solveurs de POMDP de l'état de l'art pour calculer une fonction de valeur tout en exprimant néanmoins la politique finale comme un FSC pour une exécution de politique moins gourmande en énergie. Dans le présent article, nous utilisons des variantes des mêmes techniques (i) pour calculer, à chaque itération d'inf-JESP, un FSC à partir de l'ensemble d'α-vecteurs solution d'un POMDP (voir section 4.3), ainsi que (ii) pour munir les agents du Dec-POMDP de politiques initiales (voir section 4.4).

3 État de l'art

3.1 Dec-POMDP

La recherche de comportements collaboratifs optimaux pour un groupe d'agents sous dynamique stochastique et observabilité partielle est typiquement formalisé comme un

processus de décision markovien partiellement observable décentralisé (Dec-POMDP).

Definition 1. *Un **Dec-POMDP** avec $|\mathcal{I}|$ agents est représenté par un tuple $M \equiv \langle \mathcal{I}, \mathcal{S}, \mathcal{A}, \Omega, T, O, R, b_0, H, \gamma \rangle$, où :*

- *$\mathcal{I} = \{1, \ldots, |\mathcal{I}|\}$ est un ensemble fini d'**agents** ;*
- *$\mathcal{S}$ est un ensemble fini d'**états** ;*
- *$\mathcal{A}^i$ est l'ensemble fini d'**actions** de l'agent i ; $\mathcal{A} = \bigtimes_i \mathcal{A}^i$ est l'ensemble fini des actions jointes ;*
- *Ω^i est l'ensemble fini des **observations** de l'agent i ; $\Omega = \bigtimes_i \Omega^i$ est l'ensemble fini des observations jointes ;*
- *$T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ dénote la **fonction de transition** ; $T(s, a, s')$ est la probabilité de transiter vers le prochain état s' depuis l'état s si l'action jointe a est effectuée ;*
- *$O : \mathcal{A} \times \mathcal{S} \times \Omega \to \mathbb{R}$ est la **fonction d'observation** ; $O(a, s', o)$ est la probabilité d'observer o si l'action jointe a est effectuée et l'état résultant est s' ;*
- *$R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ est la **fonction de récompense** ; $R(s, a)$ donne la récompense immédiate reçue pour l'exécution de l'action jointe a dans l'état s ;*
- *b_0 est la **distribution de probabilité initiale** (ou croyance initiale) sur les états ;*
- *$H \in \mathbb{N} \cup \{\infty\}$ est l'**horizon temporel** (potentiellement infini) ;*
- *$\gamma \in (0, 1)$ est le **facteur d'atténuation**, lequel définit l'atténuation appliquée aux récompenses futures.*

Chaque agent i peut être muni d'une politique d'action π^i qui associe à chaque historique d'action-observation possible une action à effectuer. L'objectif est alors de trouver une politique jointe $\pi = \langle \pi^1, \ldots, \pi^{|\mathcal{S}|} \rangle$ qui maximise le critère de performance, ici l'espérance du retour atténué à partir de b_0 :

$$V_H^\pi(b_0) \stackrel{\text{def}}{=} \mathbb{E}\left[\sum_{t=0}^{H-1} \gamma^{-t} r(S_t, A_t) \mid S_0 \sim b_0, \pi \right].$$

3.2 POMDP

Dans ce travail, nous nous intéresserons à des solutions de Dec-POMDP sous la forme d'*équilibres*, c'est-à-dire de situations dans lesquelles chaque agent i suit une politique qui est une *meilleure réponse* étant données les politiques fixes des autres agents, notées $\pi^{\neq i}$, ce qui induit un Dec-POMDP mono-agent, c'est-à-dire un POMDP. Dans le cas d'un POMDP, une politique optimale existe dont l'entrée est la croyance b, c'est-à-dire la distribution de probabilité sur les états étant donné l'historique d'action-observation courant. Pour un horizon h fini, la valeur de la politique π à l'état de croyance b est définie récursivement comme suit :

$$V_h^\pi(b) = r(b, \pi(b)) + \gamma \sum_o Pr(o \mid b, \pi(b)) V_{h-1}^\pi(b^{a,o}),$$

où (i) $r(b, a) = \sum_s b(s) \cdot r(s, a)$, (ii) $Pr(o \mid b, \pi(b))$ dépend de la dynamique, et (iii) $b^{a,o}$ est l'état de croyance mis-à-jour après exécution de a et perception de o.
Estimer V^* permet de dériver une politique quasi-optimale, et repose souvent sur l'équation d'optimalité de Bellman :

$$V_h^*(b) = \max_a \left[r(b, a) + \gamma \sum_o Pr(o \mid b, a) V_{h-1}^*(b^{a,o}) \right].$$

Pour un horizon fini h, V_h^* est linéaire par morceaux est convexe (PWLC) en b. Quand l'horizon est infini, $V^*(= V_\infty^*)$ peut ainsi être approché par une enveloppe supérieure d'hyperplans—appelés α-vecteurs $\alpha \in \Gamma$.

3.3 Contrôleurs à états finis

Dans les POMDP comme dans les Dec-POMDP, les politiques solutions peuvent aussi être cherchées sous la forme de *contrôleurs à états finis* (FSC) (aussi appelés *graphes politiques* [16]), c'est-à-dire des automates dont les transitions d'un nœud interne au suivant dépendent de l'observation reçue et génèrent les actions à accomplir.

Definition 2. *Un **FSC** est représenté par un tuple $fsc \equiv \langle N, \eta, \psi \rangle$, où :*

- *N est un ensemble fini de nœuds, avec n_0 le nœud initial ;*
- *$\eta : N \times \Omega \times N \to \mathbb{R}$ est la fonction de transition entre nœuds du FSC ; $\eta(n, o, n') \stackrel{\text{def}}{=} Pr(n' | n, o')$ est la probabilité de transiter vers le nœud $n' \in N$ depuis le nœud $n \in N$ si l'observation $o' \in \Omega_i$ est perçue ;la notation $n' = \eta(n, o)$ est aussi employée quand cette transition est déterministe ;*
- *$\psi : N \times \mathcal{A} \to \mathbb{R}$ est la fonction de sélection d'action du FSC ; $\psi(n, a) \stackrel{\text{def}}{=} Pr(a | n)$ est la probabilité de choisir l'action $a \in \mathcal{A}$ dans le nœud n ; la notation $a = \psi(n)$ est aussi employée quand cette fonction est déterministe.*

HANSEN [11] a établi que, dans un POMDP donné, calculer la fonction de valeur d'un FSC déterministe (η et ψ étant tous deux déterministes) nécessite de résoudre le système d'équations linéaires suivant, avec un α-vector par nœud interne :

$$\alpha_s^n = R(s, a_n) + \gamma \sum_{s', o} T(s, a_n, s') O(a_n, s', o) \alpha_{s'}^{\eta(n,o)}, \tag{1}$$

où n est l'index d'un nœud de fsc, et $a_n \stackrel{\text{def}}{=} \psi(n)$. Ce système d'équations linéaires peut être résolu par un processus itératif en tirant profit du théorème du point fixe. Ce processus est typiquement interrompu quand le résidu de Bellman (la plus grande variation de valeur) est en-dessous d'un seuil ϵ donné, de sorte que la valeur calculée est à $\frac{\epsilon}{1-\gamma}$ de la vraie valeur de la politique FSC.

4 Infinite-horizon JESP

inf-JESP repose sur une procédure de recherche locale principale, laquelle est typiquement relancée plusieurs fois avec des initialisations aléatoires différentes pour converger vers différents optima locaux. Cette recherche locale, présentée en section 4.1, de manière itérative (i) définit un POMDP meilleur-réponse pour chaque agent sur la base des politiques des autres agents (voir section 4.2), et (ii) le résout pour extraire et évaluer le FSC associé (voir section 4.3). Nous expérimentons aussi avec une initialisation non aléatoire de inf-JESP, laquelle est décrite en section 4.4.

4.1 Algorithme principal

La politique de chaque agent est représentée par un FSC déterministe. Pour contrôler le coût computationnel de chaque itération, nous bornons la taille des FSC solution avec un paramètre $K \in \mathbb{N}^*$ (de sorte que le nombre de FSC considérés est fini). La recherche locale commence ainsi avec $|\mathcal{I}|$ FSC de taille au plus K générés aléatoirement dans fsc. Ensuite, elle boucle sur les agents, chaque itération tentant d'améliorer la politique d'un agent i en trouvant (line 7) un FSC fsc'_i de taille K qui soit une meilleure réponse aux FSC $fsc_{\neq i}$ courants (figés) des $|\mathcal{I}|-1$ autres agents (dénotés $\neq i$). La section 4.2 détaille comment le problème auquel est confronté l'agent i est formalisé comme un POMDP et résolu. La ligne 8 repose sur l'équation 1 pour évaluer la solution $\langle fsc'_i, fsc_{\neq i}\rangle$ (en b_0), à moins que le solveur de POMDP en ligne 7 ne fournisse cette information. Ensuite, si une meilleure solution a été trouvée, fsc'_i remplace fsc_i dans fsc. Le processus s'arrête quand le nombre d'itérations consécutives sans améliorations, $\#ni$, atteint $|\mathcal{I}|$.

Algorithme 1 : ∞-horizon JESP

```
1  [Input :] K : FSC size | fsc : initial FSCs
2  Fct LocalSearch (K, fsc ≝ ⟨fsc_1, ..., fsc_|I|⟩)
3      v_bestL ← eval(fsc)
4      #ni ← 0          // #(iterations w/o improvement)
5      i ← 0                        // Id of current agent
6      repeat
7          fsc'_i ← Solve2FSC (fsc_≠i, K)
8          v ← eval(fsc'_i, fsc_≠i)
9          if v ≥ v_bestL then
10             fsc_i ← fsc'_i
11             v_bestL ← v
12             #ni ← 0
13         else
14             #ni ← #ni + 1
15         i ← (i + 1) mod |I|
16     until #ni = |I|
17     return ⟨fsc, v_bestL⟩
```

Propriétés Pour l'agent i, à chacun des K nœuds de son FSC est associée une action de $\mathcal{A}_i$, et à chacun de ses $K \times |\Omega|$ arcs est associé un nœud de FSC, de sorte que le nombre de FSC *déterministes* possibles $|\mathcal{FSC}_i|$ est majoré par $|\mathcal{A}_i|^K \cdot K^{K\times|\Omega|}$. [1] Nous pouvons ainsi supposer ici que chaque POMDP rencontré est résolu optimalement, ce qui conduit aux propriétés suivantes.

Proposition 1. *La recherche locale de inf-JESP converge en un nombre fini d'itérations à un équilibre de Nash.*

Démonstration. La recherche n'accepte que des solutions de qualité croissante, de sorte que le nombre d'itérations (sur tous les agents) est majoré par le nombre (fini) de solutions possibles : $|\mathcal{FSC}| = \prod_i |\mathcal{FSC}_i|$.
La recherche s'interrompt quand le FSC de chaque agent est une meilleure réponse aux FSC des autres agents, c'est-à-dire quand un équilibre de Nash est atteint. □

Notons que ces équilibres ne sont que des optima locaux. Permettre une infinité de redémarrages aléatoires garantit de converger vers un optimum local avec probabilité 1. Évidemment, l'ensemble des équilibres de Nash dépend de l'ensemble des politiques considérées, donc du paramètre K dans notre cadre. Augmenter K donne accès à plus de politiques (sans en enlever), et ainsi à des équilibres de Nash potentiellement meilleurs.
En pratique (voir section4.2), nous utiliserons un solveur de POMDP sous-optimal dans **Solve2FSC** (ligne 7) dans lequel les tailles des FSC ne sont pas contraintes par une borne K. Si ce solveur de POMDP retourne une solution fsc'_i moins bonne que fsc_i, elle sera ignorée, de sorte que les améliorations monotones sont préservées, et la recherche s'arrêtera toujours nécessairement en temps fini. Nous ne perdrons que la propriété que les solutions obtenues sont des équilibres de Nash. Elles peuvent être proches d'équilibres de Nash si le solveur de POMDP retourne des solutions ϵ-optimales. On parlera simplement d'*équilibre*.

4.2 POMDP meilleure réponse pour l'agent i

Solution à horizon fini (JESP) Dans JESP, quand on raisonne sur la politique de l'agent i en considérant les politiques $\pi_{\neq i}$ des autres agents connues et figées, l'agent i doit maintenir une croyance b^t_{JESP} sur le tuple $e^t = \langle s^t, \vec{\omega}^t_{\neq i}\rangle$, où s^t dénote l'état courant et $\vec{\omega}^t_{\neq i}$ les historiques d'action-observation des autres agents. Cette croyance sur e^t est une statistique suffisante parce que, en l'utilisant, l'agent i peut déduire la distribution de probabilité sur l'état courant du système et inférer les actions des autres agents $a^t_{\neq i} = \pi_{\neq i}(\vec{\omega}^t_{\neq i})$.
Toutefois, dans les problèmes à horizon infini, l'agent i ne peut considérer les historiques d'action-observation complets $\vec{\omega}^t_{\neq i}$ des autres agents parce que leur nombre croît exponentiellement avec le temps, de sorte que l'espace d'états

1. Le nombre exact est plus petit du fait de symétries et parce que, dans certains FSC, tous les nœuds internes ne sont pas atteignables.

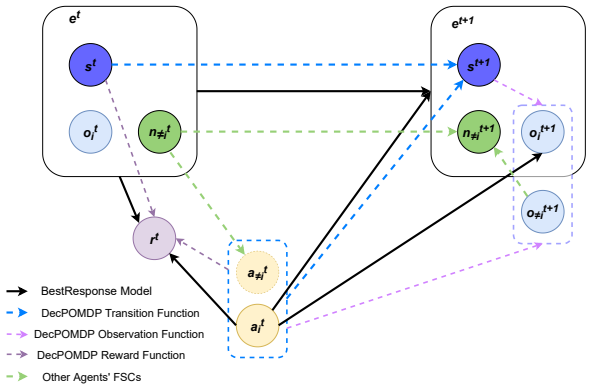

FIGURE 1 – Structure du POMDP meilleure-réponse : Les flèches noires décrivent un POMDP standard ; Les flèches bleues montrent la fonction de transition T du Dec-POMDP ; Les flèches roses montrent la fonction d'observation O du Dec-POMDP ; Les flèches violettes montrent la fonction de récompense R ; Les flèches vertes montrent l'évolution des nœuds internes des agents $\neq i$.

est infini. Avec des politiques représentées par des FSC, l'agent i peut raisonner sur les nœuds internes des autres agents au lieu des historiques d'action-observation. Mais combiner des FSC avec un Dec-POMDP n'est pas trivial (voir annexe A, page 12).

Solution à horizon infini Plusieurs POMDP meilleure-réponse peuvent être conçus. Les ensembles d'actions et d'observations sont imposés ($\mathcal{A}_i$ et Ω_i), mais plusieurs choix sont possibles pour l'ensemble des états *étendus*. Ici, l'état étendu $e^t \in \mathcal{E}$ contient (i) s^t, l'état courant du Dec-POMDP, (ii) $n^t_{\neq i} \equiv \langle n^t_1, \ldots, n^t_{i-1}, n^t_{i+1}, \ldots, n^t_n \rangle$, les nœuds des FSC des $|\mathcal{I}| - 1$ autres (agents $\neq i$) au pas de temps courant, et (iii) o^t_i, l'observation courante de l'agent i. Nous avons ainsi $\mathcal{E} \stackrel{\text{def}}{=} \mathcal{S} \times N_{\neq i} \times \Omega_i$. Étant donnée une action a^t_i, l'état étendu $e^t \equiv \langle s^t, n^t_{\neq i}, o^t_i \rangle$ évolue selon les étapes suivantes (voir figure 1) :

1. d'abord, chaque agent $j \neq i$ sélectionne son action a_j selon son nœud de FSC courant ;
2. ensuite l'état s^t transitionne vers s^{t+1} selon la fonction de transition du Dec-POMDP, l'action a^i de l'agent i et les actions des agents $\neq i$ ($a^t_{\neq i}$), c'est-à-dire l'action jointe $a^t \equiv \langle a^t_i, a^t_{\neq i} \rangle$; et
3. les nœuds FSC des agents j (y compris i) évoluent conjointement selon leurs observations (pas nécessairement indépendantes) o^{t+1}_j, lesquelles peuvent être inférées de s^{t+1} et de l'action jointe a^t.

Ce choix de conception pour l'état étendu induit un POMDP parce que les propriétés suivantes sont satisfaites : (i) il induit un processus markovien (contrôlé par l'action) ; (ii) l'observation dépend de l'action et du prochain état étendu ; et (iii) la récompense dépend de l'état et de l'action. En effet, dériver les fonctions de transition, d'observation et de récompense pour ce POMDP meilleure-réponse (voir annexe A) conduit à :

$$T_e(e^t, a^t_i, e^{t+1}) = Pr(e^{t+1}|e^t, a^t_i)$$
$$= \sum_{o^{t+1}_{\neq i}} T(s^t, \langle \psi_{\neq i}(n^t_{\neq i}), a^t_i \rangle, s^{t+1}) \cdot \eta_{\neq i}(n^t_{\neq i}, o^{t+1}_{\neq i}, n^{t+1}_{\neq i})$$
$$\cdot O(s^{t+1}, \langle \psi_{\neq i}(n^t_{\neq i}), a^t_i \rangle, \langle o^{t+1}_{\neq i}, o^{t+1}_i \rangle),$$
$$O_e(a^t_i, e^{t+1}_i, o^{t+1}_i) = Pr(o^{t+1}_i|a^t_i, e^{t+1}_i)$$
$$= Pr(o^{t+1}_i|a^t_i, \langle s^{t+1}, n^{t+1}_{\neq i}, \tilde{o}^{t+1}_i \rangle) = \mathbf{1}_{o^{t+1}_i = \tilde{o}^{t+1}_i},$$
$$r_e(e^t, a^t_i) = r(s^t, a^t_i, \psi_{\neq i}(n^t_{\neq i})),$$

où $\eta_{\neq i}(n^t_{\neq i}, o^{t+1}_{\neq i}, n^{t+1}_{\neq i}) = \prod_{j \neq i} \eta(n^t_j, \tilde{o}^{t+1}_j, n^{t+1}_j)$ et $\psi_{\neq i}(n^t_{\neq i}) = a^t_{\neq i}$.
Ce POMDP meilleure-réponse peut être vu comme un MDP à observabilité mixte (MOMDP [19, 4]), puisqu'une des variables d'état est complètement observable. Aussi, $\mathcal{E}$ peut être rendu plus petit en enlevant les états étendus impossibles. En particulier, comme on peut l'observer dans la fonction de transition T_e, une observation o^t_i peut être impossible sous un certain état s^t (du fait de la fonction d'observation).[2] L'efficacité de ce processus d'éliminination d'états (étendus) dépend évidemment beaucoup du Dec-POMDP considéré, le pire cas étant quand toutes les observations ont toujours une probabilité d'occurrence non nulle.

4.3 Obtenir et évaluer fsc_i

Après avoir construit le POMDP (meilleure-réponse) pour l'agent i, un solveur de POMDP est requis pour obtenir la politique de l'agent i. Il y a différents types de solveurs. Un premier type donne directement un FSC qui peut être utilisé pour construire le prochain POMDP meilleure-réponse [11]. Un second type approche la fonction de valeur optimale, comme les algorithmes à base de points tels que PBVI [22], HSVI [24], et SARSOP [14]. Pour tirer profit de ces solveurs modernes, nous choisissons cette seconde méthode pour résoudre le POMDP meilleure-réponse et obtenir un ensemble Γ d'α-vecteurs.

Algorithme Nous utilisons l'algorithme 2, qui est similaire à ce qui a été proposé par GRZEŚ et coll. [10], pour transformer une fonction de valeur approchée (sous la forme d'un ensemble d'α-vecteurs Γ) en un FSC. Il créé d'abord un nœud initial n_0 à partir de l'état de croyance initial b_0 et de Γ (ligne 2), et l'ajoute à la fois au nouveau FSC (N) et à une file (G). Ensuite, l'algorithme traite chaque nœud $n = \langle \alpha, b, a \rangle$ de G (premier entré d'abord), avec chaque observation possible o_i comme suit (les observations de probabilité nulle induisant des boucles (ligne 18)). Une mise-à-jour de l'état de croyance produit $b^{o_i}_{a_i}$ avant d'identifier $\alpha_i \in \Gamma$, l'α-vecteur dominant à $b^{o_i}_{a_i}$. Le nœud

2. Il peut aussi y avoir des restrictions dues aux FSC. En fait, une approche plus générale serait de ne considérer dans $\mathcal{E}$ que les états étendus atteignables depuis l'état de croyance initial.

$n' \in N$ qui contient α_i est extrait s'il existe (ligne 16), sinon un nouveau nœud $n' \stackrel{\text{def}}{=} \langle \alpha_i, b^{o_i}_{a_i} \rangle$ est créé (ligne 12) et ajouté à la fois à N et G. Un arc est ajouté entre n et n', de label o_i (ligne 19).

Comme dans le travail de GRZEŚ et coll. et comme déjà mentionné, des boucles sont ajoutées quand une observation improbable $Pr(o_i|b, a_i) = 0$ est rencontrée. Cela peut arriver parce que, quand on construit le FSC, chaque nœud n_i est associé à un tuple $\langle \alpha_i, b_i \rangle$, et les arcs sortant de n_i seront créés seulement pour les observations qui ont une probabilité non-nulle dans b_i. Or, durant l'exécution, n_i peut être atteint via un état de croyance b' différent depuis lequel a_i (l'action attachée à α_i, donc à n_i) peut induire des observations "inattendues". [3] Ajouter des boucles est une façon de munir l'agent d'une stratégie par défaut quand une observation inattendue a lieu.

Il y a toutefois deux différences dans notre méthode par rapport au travail de GRZEŚ et coll.

— Premièrement, dans leur travail, le nombre de nœuds $|N|$ du FSC compilé est égal au nombre d'α-vecteurs dans $|\Gamma|$. Dans notre méthode, le nombre de nœuds est réduit puisque nous n'explorons que les états de croyance atteignables depuis l'état de croyance initial b_0, et ne stockons qu'un α-vector par état de croyance.

— Deuxièmement, dans notre cas, nous n'avons pas une mais deux raisons d'ajouter des boucles. La première raison est la même que dans les travaux de GRZEŚ et coll. La seconde raison est que, dans l'approche JESP, chaque FSC d'un agent i est obtenu en considérant les FSC des agents $\neq i$ figés. Toutefois, modifier les FSC des autres agents va conduire à un nouveau POMDP du point de vue de i, avec potentiellement de nouvelles observations rencontrées depuis certains nœuds de fsc_i.

Notons que cet algorithme ne borne pas le nombre résultant de nœuds internes. À la place, nous comptons sur (i) le fait que SARSOP retourne un nombre fini d'α-vecteurs, et (ii) la production de (significativement) moins de $|\Gamma|$ nœuds internes par l'extraction de FSC.

Une fois fsc_i obtenu, l'étape suivante est l'évaluation commune de toutes les politiques des agents dans le Dec-POMDP. Dans ce but, il est suffisant d'évaluer fsc_i dans le POMDP meilleure-réponse correspondant, puisque celui-ci combine dans un seul modèle le Dec-POMDP et les FSC des agents $\neq i$. Ici, nous employons la technique d'évaluation de FSC décrite en section 3.3.

4.4 Initialisations MPOMDP

Commme mentionné précédemment, l'initialisation d'inf-JESP est importante. Même si inf-JESP améliore la valeur du FSC joint à chaque itération, des initialisations aléatoires peuvent souvent conduire à de mauvais optima locaux. Nous souhaitons donc investiguer si des heuristiques d'initialisation non-aléatoires permettent de trouver de bonnes solutions vites et fiablement. Notre méthode part de l'hypothèse d'observations publiques dans le Dec-POMDP, de sorte que nous sommes face à un POMDP multi-agent (MPOMDP) [23], c'est-à-dire un problème formellement équivalent à un POMDP et donc résolu à l'aide d'un solveur de POMDP. Nous extrayons des FSC individuels de la politique MPOMDP résultante comme détaillé ci-dessous, et les employons pour initialiser inf-JESP.

3. Cela arrivera si un état s a une probabilité nulle dans b_i mais pas dans b' et peut induire cette observation quand a_i est effectuée.

Algorithme 2 : Compilation d'un ensemble Γ d'α-vecteurs en un contrôleur à états finis $\langle N, \eta, \psi \rangle$

```
1  [Input :] Γ : α-vector set
2  Start node n_0 ← node(arg max_{α∈Γ} α · b_0, b_0)
3  N ← {n_0}
4  G.pushback(n_0)
5  while |G| > 0 do
6      n ← G.popfront()
7      (b, a_i) ← (n.b, ψ(n))
8      forall o_i ∈ Ω_i do
9          if Pr(o_i|b, a_i) > 0 then
10             α_i ← argmax_{α∈Γ} α · b^{o_i}_{a_i}
11             if α_i ∉ N then
12                 n' ← node(α_i, b^{o_i}_{a_i})
13                 N ← N ∪ {n'}
14                 G.pushback(n')
15             else
16                 n' ← N(α_i)
17         else
18             n' ← n
19         η(n, o_i) ← n'
20 return ⟨N, η, ψ⟩
```

FSC initial basé-MPOMDP stochastique (M-S) – L'algorithme 3 extrait une politique fsc_i pour l'agent i d'une politique MPOMDP. Cette approche est similaire à l'algorithme 2, la différence principale étant que les observations et actions des agents $\neq i$ devraient être aussi considérées pour calculer le prochain état de croyance. Ce n'est toutefois pas vraiment faisable puisque l'agent i ne les connaît pas. Pour résoudre ce problème, étant donné l'agent i, considérons un nœud courant $n = \langle \alpha, b \rangle$ et une observation individuelle o_i (de probabilité d'occurrence non nulle). La solution MPOMDP spécifie une action jointe $a = \langle a_i, a_{\neq i} \rangle$ en b, a étant l'action associée à α. Nous supposons arbitrairement que (i) chaque transition stochastique possible du FSC (de n et associée à l'observation o_i) correspond à une observation jointe possible $o_{\neq i}$ des autres agents, (ii) une telle transition a lieu avec probabilité $Pr(o_{\neq i} \mid b, a, o_i)$, et (iii) elle "conduit" à une nouvelle croyance $b_a^{\langle o_i, o_{\neq i} \rangle}$ et ainsi à un nœud n' attaché à l'α-vecteur MPOMDP dominant (α_i, *cf.* ligne 12) en ce point. Comme seulement un nœud FSC devrait correspondre à un α-vecteur

donné, un nouveau n' attaché à α_i n'est créé que si nécessaire (lignes 13, 14 et 18). Comme de multiples observations jointes $o_{\neq i}$ peuvent conduitre au même n', les probabilités de transition correspondantes sont cumulées dans $\eta(n, o_i, n')$ (ligne 19). Les observations jointes $o_{\neq i}$ de probabilité nulle étant donnés b, a et o_i sont ignorées. Les observations individuelles o_i de probabilité nulle étant donnés b et a induisent la création d'une boucle (ligne 21).

Notons que les FSC résultants sont stochastiques, et que certains d'entre eux pourraient ne pas être améliorables, de sorte que, dans ce cas, la solution retournée par inf-JESP pourrait contenur des FSC stochastiques.

FSC initial basé-MPOMDP déterministe (M-D) – Une variante très simple de cette méthode est de supposer que la seule transition possible de n sous o_i correspond à l'observation jointe la plus probable $o_{\neq i}$ des autres agents. La transition entre nœuds est alors déterministe ($\eta(n, o_i, n') = 1$).

Notes : (1) Comme (i) ces méthodes d'initialisation heuristiques et (ii) la recherche locale d'inf-JESP sont déterministes, utiliser une procédure déterministe pour dériver des FSC meilleures-réponses induit un algorithme déterministe pour lequel les redémarrages sont inutiles. (2) Ces méthodes d'initialisation heuristiques peuvent aussi être adaptées au cadre à horizon fini de JESP en remplaçant la représentation de politique par des arbres à horizon fini.

Algorithme 3 : Extraction de fsc_i pour l'agent i à partir de l'ensemble Γ d'α-vecteurs solution du MPOMDP

```
1  [Input :] i : agent | Γ : MPOMDP α-vector set
2  Start node n_0 ← node(argmax_{α∈Γ} α · b_0, b_0)
3  N ← {n_0}
4  G.pushback(n_0)
5  while |G| > 0 do
6      n ← G.popfront()
7      (b, a) ← (n.b, ψ(n))
8      forall o_i ∈ Ω_i do
9          if Pr(o_i|b, a) > 0 then
10             forall o_{≠i} ∈ Ω_{≠i} do
11                 if Pr(o_{≠i}|o_i, b, a) > 0 then
12                     α_i ← argmax_α(b_a^{⟨o_i,o_{≠i}⟩}, Γ)
13                     if α_i ∉ N then
14                         n' ← node(α_i, b_a^{⟨o_i,o_{≠i}⟩})
15                         N ← N ∪ {n'}
16                         G.pushback(n')
17                     else
18                         n' ← N(α_i)
19                     η(n, o_i, n') ←
                         η(n, o_i, n') + Pr(o_{≠i}|o_i, b, a)
20         else
21             η(n, o_i, n) ← 1
22 return ⟨N, η, ψ⟩
```

5 Expérimentations

Dans cette section, nous appliquons l'algorithm inf-JESP sur divers bancs d'essai Dec-POMDP standards. Comme il y a différents choix d'initialisations, nous comparons différents réglages de inf-JESP. Nous incluons aussi les résultats de plusieurs algorithmes de l'état de l'art pour résoudre des Dec-POMDP à horizon infini.

5.1 Méthode de comparaison

Nous utilisons des problèmes Dec-POMDP standards pour évaluer notre algorithme et ses variantes : DecTiger, Recycling, Meeting in a 3×3 grid (*aka* Grid), Box Pushing, et Mars Rover. Tous ces problèmes sont disponibles en ligne [4].

Nous comparons les différentes variantes d'inf-JESP avec des solveurs Dec-POMDP de l'état de l'art, à savoir : FB-HSVI [9], Peri [21], PeriEM [21], PBVI-BB [15] et MealyNLP [3].

Nous ignorons l'algorithme JESP original puisqu'il ne peut pas gérer des Dec-POMDPs à horizon infini. Nous comparons les résultats avec Dec-BPI séparément parce que nous ne pouvons qu'estimer les valeurs à partir de graphes empiriques sur quelques bancs d'essai [6].

5.2 Réglages des algorithmes

Nous avons utilisé SARSOP [14] comme solveur POMDP fournissant un ensemble solution d'α-vecteurs. Nous fixons un temps limite de 5 s à SARSOP, et un résidu de Bellman $= 0,001$ à la fois pour l'évaluation de FSC et la précision de SARSOP.

Nous avons testé quatre différentes implémentations d'inf-JESP. Les première et deuxième, IJ(M-D) et IJ(M-S), sont inf-JESP initialisé avec les heuristiques M-D et M-S (*cf.* section 4.4) et sans redémarrage parce que, sans options de randomisation et malgré le temps limite de 5 s, SARSOP se conduit de manière déterministe (comme observé empiriquement). Les troisième et quatrième, IJ(R-1_{runs}) et IJ(R-100_{runs}), sont inf-JESP avec des initialisations aléatoires et différents nombres de redémarrages (1 (re)démarrage et 100 redémarrages).

Par exemple, "IJ(R-100_5)" signifie 5 exécutions d'inf-JESP, chacun avec 100 redémarrages. Toutes les initialisations aléatoires sont bornées avec 5 nœuds pour chaque FSC. Pour chaque redémarrage, nous fixons une limite de temps de 7200 s. À cause du temps limité, nous n'avons pas effectué de tests avec redémarrage aléatoire sur les domaines Box-Pushing and Mars Rover.

Les expérimentations avec inf-JESP ont été conduites sur un portable doté d'un CPU i5-1.6 GHz et 8 GB de RAM.

5.3 Résultats

Comparaison avec les algorithmes de l'état de l'art. La table 1 présente les résultats obtenus sur les 5 problèmes Dec-POMDP standards employés comme bancs d'essai. Pour inf-JESP avec initialisation aléatoire et redémarrages,

4. `http://masplan.org/problem`

TABLE 1 – Comparaison de différents algorithmes en termes de taille finale des FSC, nombre d'itérations requis, temps et valeur, sur 5 bancs d'essai Dec-POMDP à horizon infini, avec $\gamma = 0.9$ dans chaque cas. R-R_N : initialisations aléatoires avec R redémarrages (évalué avec N exécutions). M-S and M-D : initialisations FSC basées-MPOMDP stochastiques et déterministes.

Alg.	FSC size	#Iterations	Time (s)	Value
DecTiger ($\|\mathcal{I}\| = 2, \|\mathcal{S}\| = 2, \|\mathcal{A}^i\| = 3, \|\mathcal{Z}^i\| = 2$)				
FB-HSVI			153,7	13,448
PBVI-BB				13,448
Peri			220	13,45
IJ(R-100_5)	$34_{\pm 34} \times 34_{\pm 34}$	$22 \ _{\pm 3}$	7361,8	13,30
IJ(M-D)	75×81	27	157	13,10
IJ(M-S)	75×79	27	164	13,10
PeriEM			6450	9,42
MealyNLP			29	$-1,49$
IJ(R-1_{500})	$52_{\pm 61} \times 53_{\pm 65}$	$12 \ _{\pm 10}$	73,62	$-67,81$
Recycling ($\|\mathcal{I}\| = 2, \|\mathcal{S}\| = 4, \|\mathcal{A}^i\| = 3, \|\mathcal{Z}^i\| = 2$)				
FB-HSVI			2,6	31,929
PBVI-BB				31,929
MealyNLP			0	31,928
Peri			77	31,84
IJ(R-100_{10})	$3_{\pm 0} \times 3_{\pm 0}$	$3 \ _{\pm 0}$	22,3	31,92
PeriEM			272	31,80
IJ(R-1_{1000})	$5_{\pm 8} \times 5_{\pm 8}$	$3 \ _{\pm 0}$	0,223	29,70
IJ(M-S)	8×8	3	0	26,57
IJ(M-D)	4×4	3	0	25,65
Grid3*3 ($\|\mathcal{I}\| = 2, \|\mathcal{S}\| = 81, \|\mathcal{A}^i\| = 5, \|\mathcal{Z}^i\| = 9$)				
IJ(M-D)	8×8	5	13	5,81
IJ(M-S)	8×8	7	88	5,81
IJ(R-100_5)	$9_{\pm 1} \times 12_{\pm 4}$	$3 \ _{\pm 0}$	3788,4	5,81
FB-HSVI			67	5,802
IJ(R-1_{500})	$11_{\pm 14} \times 14_{\pm 24}$	$4 \ _{\pm 1}$	37,88	5,40
Peri			9714	4,64
Box-pushing ($\|\mathcal{I}\| = 2, \|\mathcal{S}\| = 100, \|\mathcal{A}^i\| = 4, \|\mathcal{Z}^i\| = 5$)				
FB-HSVI			1715,1	224,43
PBVI-BB				224,12
IJ(M-S)	158×197	6	2443	220,26
IJ(M-D)	78×288	4	1564	202,94
Peri			5675	148,65
MealyNLP			774	143,14
PeriEM			7164	106,65
Mars Rover ($\|\mathcal{I}\| = 2, \|\mathcal{S}\| = 256, \|\mathcal{A}^i\| = 6, \|\mathcal{Z}^i\| = 8$)				
FB-HSVI			74,31	26,94
IJ(M-D)	19×49	7	1432	26,91
IJ(M-S)	41×41	6	891	24,20
Peri			6088	24,13
MealyNLP			396	19,67
PeriEM			7132	18,13

à chaque exécution nous avons gardé la plus haute valeur parmi les redémarrages, et ensuite calculé la moyenne de ces valeurs sur les exécutions effectuées (1 exécution = 100 redémarrages). Les cinq colonnes de la table fournissent : (*Alg.*) les différents algorithmes comparés, (*FSC size*) les tailles finales des FSC après convergence (pour les approches inf-JESP), (*#Iterations*) le nombre d'itérations requises pour atteindre un équilibre (pour les approches inf-JESP), (*Time*) le temps de calcul, (*Value*) la valeur finale de la politique jointe. Les résultats donnés pour les variantes de inf-JESP sont des minorants, de sorte que la valeur réelle des FSC obtenus pourrait être plus élevée (avec un différence maximum de $0,001$).

En termes de valeur finale atteinte, inf-JESP peut avoir des solutions raisonnablement bonnes dans la plupart des domaines avec différentes méthodes d'initialisation. Même si les valeurs obtenues avec inf-JESP ne peuvent pas concurrencer celles atteintes par FB-HSVI, dans la plupart des domaines l'écart est relativement faible. Nous avons aussi comparé nos résultats avec Dec-BPI en estimant les valeurs qu'il peut atteindre à travers les figures présentées dans [7] (page 123). Même si Dec-BPI n'a été testé que sur trois problèmes (à savoir, DecTiger, Grid, et Box-Pushing), inf-JESP surpasse Dec-BPI dans les trois cas. Il faut aussi noter que les résultats de Dec-BPI dépendent fortement de la taille des FSC considérés et que Dec-BPI rencontre des difficultés quand les FSC ont beaucoup de nœuds, comme on peut le voir sur le problème DecTiger [7].

Concernant le temps de résolution, IJ(M-S), IJ(M-D) et IJ(R-1_{500}/R-1_{1000}) ont des comportements différents selon le problème considéré. Par exemple, dans le problème DecTiger, IJ(M-S) et IJ(M-D) ont presque le même temps de résolution, et tous deux ont besoin de plus de temps que IJ(R-1_{500}). Par contre, pour le problème Grid, IJ(M-D) a besoin de significativement moins de temps que IJ(M-S) et IJ(R-1_{500}). Dans l'ensemble, IJ(M-D) et IJ(M-S) peuvent donner de bonnes solutions en un temps acceptable.

Étude d'inf-JESP. La figure 2 (gauche) présente l'évolution des valeurs des FSC pendant l'exécution de JESP en fonction du nombre d'itérations. La courbe bleue présente cette évolution en moyenne (ainsi que l'écart-type correspondant) pendant des exécutions initialisées aléatoirement, alors que les courbes rouges et vertes présentent les résultats déterministes avec les initialisation M-S et M-D. Pour les initialisations aléatoires, il est à noter que, à chaque itération, seuls les FSC n'ayant pas convergé à un optimum local sont considérés dans la moyenne. Cela explique pourquoi cette moyenne peut (i) être plus élevée que la distribution sur les valeurs finales (à gauche), et (ii) décroître, puisqu'une exécution avec une solution de haute valeur peut s'être arrêtée juste avant. Néanmoins, cette figure souligne clairement que la recherche locale d'inf-JESP avec initialisations aléatoires améliore effectivement la valeur du FSC à chaque itération jusqu'à convergence à une solution localement optimale. Elle montre aussi que les initialisations MPOMDP sont des heuristiques plutôt bonnes comparées aux initialisations aléatoires, mais, comme attendu, inf-JESP avec initialisation MPOMDP n'atteint pas toujours un optimum global (comme pour le problème Recycling).

La partie droite de la figure 2 présente la distribution des valeurs des FSC finaux pour différentes exécutions de la recherche locale d'inf-JESP avec initialisations aléatoires. Elle souligne donc la fréquence d'accès à des valeurs spécifiques après convergence d'inf-JESP(R-1_1). Il faut d'abord noter que, dans certaines exécutions, inf-JESP(R-1_1) a été capable d'atteindre un optimum global même si les FSC

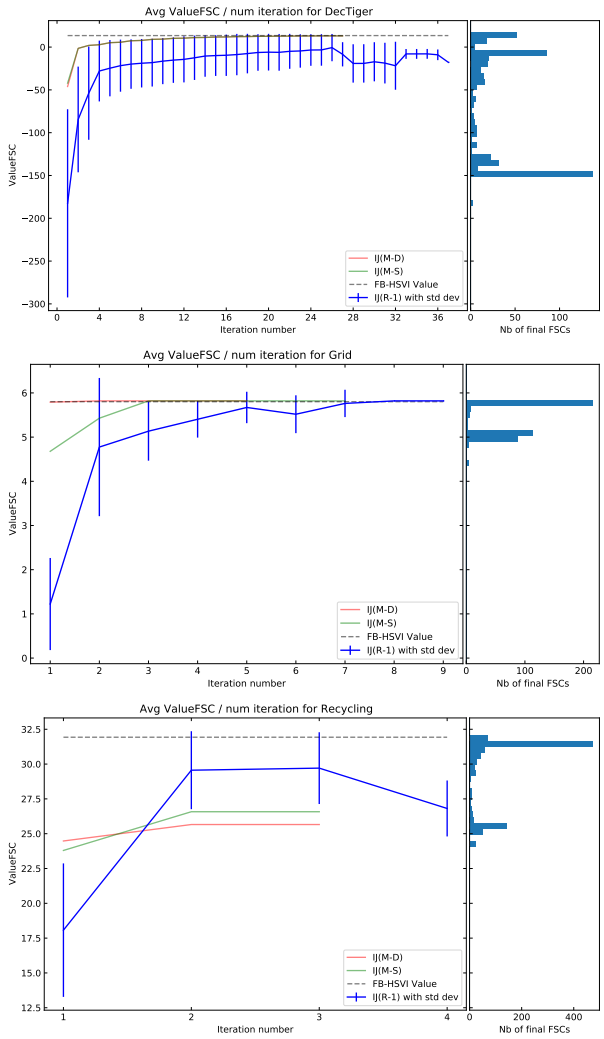

FIGURE 2 – Valeurs des politiques FSC jointes pour les bancs d'essai considérés (de haut en bas) : DecTiger, Grid and Recycling. La partie gauche de chaque figure présente l'évolution (pendant une exécution) de la valeur du FSC joint à chaque itération de inf-JESP(R-1_1) (moyenne et écart-type en bleu), et des algorithmes déterministes inf-JESPM-D (en rouge) et inf-JESPM-S (en vert). La partie droite présente la distribution des valeurs des FSC joints obtenus après convergence de inf-JESP(R-1_1). La ligne tiretée représente la valeur obtenue avec FB-HSVI.

initiaux sont limités à seulement 5 nœuds. Si cela arrive dans chacun de ces 3 problèmes, cela pourrait ne pas être le cas pour d'autres Dec-POMDP. Toutefois, cela démontre quand même que, dans certains contextes, inf-JESP peut tirer parti de petits FSC pour produire des politiques jointes utiles. En fonction du problème, les FSC joints avec des valeurs comparables à la politique optimale peuvent être difficiles à atteindre, comme pour les problèmes DecTiger ou Recycling où il semble facile d'atteindre une solution quasi-optimale sans avoir une forte probabilité d'atteindre un FSC joint globalement optimal. Cette distribution donne aussi une idée générale du nombre de redémarrages nécessaires pour atteindre un optimum global avec forte probabilité. Par exemple, inf-JESP aurait probablement besoin de moins de redémarrages pour trouver un FSC joint optimal pour le problème Grid que pour DecTiger.

Résultats complémentaires. Les expérimentations conduites ont aussi montré qu'inf-JESP exhibe des comportements différents selon le problème abordé. Les résultats obtenus sont décrits en détails dans l'annexe B. Il est à noter que l'élimination d'états (comme expliqué en section 4.2) se comporte différemment selon le problème considéré. Elle peut être très efficace, comme pour le problème Grid, où elle réduit le nombre d'états du POMDP meilleure-réponse par un facteur 10 à chaque itération. Par contre, comme attendu, l'élimination d'états est sans effet sur le problème DecTiger parce que n'importe quelle observation peut être reçue depuis n'importe quel état de ce Dec-POMDP. Nous avons aussi examiné les tailles des FSC finaux obtenus par inf-JESP avec initialisations aléatoires. Nous observons que ceux dont les valeurs sont élevées ne sont pas ceux qui requièrent le plus grand nombre d'itérations ou ayant le plus grand nombre de nœuds. Des FSC de petite taille semblent aussi suffisants pour induire de hautes valeurs, ce qui ouvre une direction de recherche intéressante pour combiner inf-JESP avec de la compression de FSC.

6 Conclusion

Cet article apporte deux contributions principales.
Premièrement, nous avons proposé un nouveau solveur de Dec-POMDP à horizon infini appelé inf-JESP. Ce solveur repose sur l'approche JESP et une nouvelle formalisation qui combine un problème Dec-POMDP avec des FSC connus pour construire un POMDP mono-agent meilleure-réponse. Nous avons traité la limitation de JESP aux horizons finis en tirant profit de solveurs à base de points modernes tels que SARSOP. Nous avons comparé les résultats d'inf-JESP avec plusieurs solveurs de Dec-POMDP de l'état de l'art à travers des expérimentations sur cinq bancs d'essai. Même si, comme JESP, inf-JESP ne converge que vers des optima locaux, les résultats obtenus montrent qu'inf-JESP avec des initialisations aléatoires ou basées-MPOMDP peut quand même fournir des politiques jointes utiles pour les cinq problèmes considérés. Notre nouvelle formalisation de POMDP meilleure-réponse pourrait aussi être utile en elle-même (hors d'inf-JESP), dans des cadres où les comportements d'autres agents sont définis indépendamment, comme dans des jeux [18] ou pour l'interaction homme-robot [8].
Deuxièmement, nous avons fourni une méthode pour extraire des politiques individuelles (FSC) de l'ensemble solution Γ d'un MPOMDP pour initialiser inf-JESP. Cette approche peut être facilement adaptée à JESP pour des horizons finis. Des résultats empiriques ont montré que cette méthode d'initialisation peut, dans certains cas, atteindre des solutions de bonne qualité avec une valeur plus éle-

vée que la solution moyenne d'inf-JESP avec initialisation aléatoire. Cette approche ne marche toutefois pas toujours. Dans le problème Recycling, elle est pire que la valeur moyenne d'inf-JESP avec initialisations aléatoires. Comment dériver de meilleures heuristiques (éventuellement randomisées) de politiques MPODMP reste une question ouverte.
Les travaux futurs incluent des expérimentations avec (i) des FSC de taille bornée ; (ii) différentes formalisations des POMDP meilleures-réponses ; (iii) différentes durées limites pour SARSOP ; (iv) des redémarrages parallèles.

Références

[1] D. ABERDEEN. « Policy-Gradient Algorithms for Partially Observable Markov Decision Processes ». Thèse de doct. Canberra, Australia : The Australian National University, mar. 2003.

[2] C. AMATO, D. S. BERNSTEIN et S. ZILBERSTEIN. « Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs ». *Journal of Autonomous Agents and Multi-Agent Systems* 21.3 (2010), p. 293-320. DOI : 10.1007/s10458-009-9103-z.

[3] C. AMATO, B. BONET et S. ZILBERSTEIN. « Finite-State Controllers Based on Mealy Machines for Centralized and Decentralized POMDPs ». Dans : *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*. Sous la dir. de M. FOX et D. POOLE. AAAI Press, 2010.

[4] M. ARAYA-LÓPEZ, V. THOMAS, O. BUFFET et F. CHARPILLET. « A Closer Look at MOMDPs ». Dans : *Proceedings of the Twenty-Second IEEE International Conference on Tools with Artificial Intelligence (ICTAI-10)*. Arras, France, 2010.

[5] D. BERNSTEIN, R. GIVAN, N. IMMERMAN et S. ZILBERSTEIN. « The Complexity of Decentralized Control of Markov Decision Processes ». *Mathematics of Operations Research* 27.4 (2002), p. 819-840.

[6] D. S. BERNSTEIN, C. AMATO, E. A. HANSEN et S. ZILBERSTEIN. « Policy Iteration for Decentralized Control of Markov Decision Processes ». *Journal of Artificial Intelligence Research* 34 (2009), p. 89-132. DOI : 10.1613/jair.2667.

[7] D. S. BERNSTEIN, E. A. HANSEN et S. ZILBERSTEIN. « Bounded Policy Iteration for Decentralized POMDPs ». Dans : *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 2005.

[8] A. BESTICK, R. BAJCSY et A. DRAGAN. « Implicitly Assisting Humans to Choose Good Grasps in Robot to Human Handovers ». Dans : *2016 International Symposium on Experimental Robotics*. Mar. 2017, p. 341-354. ISBN : 978-3-319-50114-7. DOI : 10.1007/978-3-319-50115-4_30.

[9] J. DIBANGOYE, C. AMATO, O. BUFFET et F. CHARPILLET. « Optimally Solving Dec-POMDPs as Continuous-State MDPs ». *Journal of Artificial Intelligence Research* 55 (2016), p. 443-497.

[10] M. GRZEŚ, P. POUPART, X. YANG et J. HOEY. « Energy Efficient Execution of POMDP Policies ». *IEEE Transactions on Cybernetics* 45 (2015), p. 2484-2497. DOI : 10.1109/TCYB.2014.2375817.

[11] E. HANSEN. « An Improved Policy Iteration Algorithm for Partially Observable MDPs ». Dans : *Advances in Neural Information Processing Systems*. Sous la dir. de M. JORDAN, M. KEARNS et S. SOLLA. T. 10. MIT Press, 1998, p. 1015-1021.

[12] E. HANSEN. « Solving POMDPs by Searching in Policy Space ». Dans : *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*. 1998.

[13] A. KUMAR, S. ZILBERSTEIN et M. TOUSSAINT. « Probabilistic Inference Techniques for Scalable Multiagent Decision Making ». *Journal of Artificial Intelligence Research* 53 (2015), p. 223-270. DOI : 10.1613/jair.4649.

[14] H. KURNIAWATI, D. HSU et W. LEE. « SARSOP : Efficient point-based POMDP planning by approximating optimally reachable belief spaces ». Dans : *Robotics : Science and Systems IV*. 2008.

[15] L. C. MACDERMED et C. ISBELL. « Point Based Value Iteration with Optimal Belief Compression for Dec-POMDPs ». Dans : *Advances in Neural Information Processing Systems 26*. 2013.

[16] N. MEULEAU, K.-E. KIM, L. KAELBLING et A. CASSANDRA. « Solving POMDPs by searching the space of finite policies ». Dans : *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*. 1999, p. 417-426.

[17] R. NAIR, M. TAMBE, M. YOKOO, D. PYNADATH et S. MARSELLA. « Taming decentralized POMDPs : Towards efficient policy computation for multiagent settings ». Dans : *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*. 2003.

[18] F. OLIEHOEK, M. SPAAN et N. VLASSIS. « Best-response play in partially observable card games ». *Benelearn 2005 : Proceedings of the 14th Annual Machine Learning Conference of Belgium and the Netherlands* (jan. 2005).

[19] S. ONG, S. PNG, D. HSU et W. LEE. « POMDPs for robotic tasks with mixed observability ». Dans : *Proceedings of Robotics : Science and Systems V (RSS'09)*. 2009.

[20] J. PAJARINEN et J. PELTONEN. « Efficient Planning for Factored Infinite-Horizon DEC-POMDPs ». Dans : *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. AAAI Press, juil. 2011, p. 325-331. DOI : 10.5591/978-1-57735-516-8/IJCAI11-064.

[21] J. PAJARINEN et J. PELTONEN. « Periodic Finite State Controllers for Efficient POMDP and DEC-POMDP Planning ». Dans : *Advances in Neural Information Processing Systems 24*. Sous la dir. de J. SHAWE-TAYLOR, R. ZEMEL, P. BARTLETT, F. PEREIRA et K. Q. WEINBERGER. T. 24. Curran Associates, Inc., 2011.

[22] J. PINEAU, G. GORDON et S. THRUN. « Point-Based Value Iteration : An Anytime Algorithm for POMDPs ». Dans : *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. IJCAI'03. Acapulco, Mexico : Morgan Kaufmann Publishers Inc., 2003, p. 1025-1030.

[23] D. V. PYNADATH et M. TAMBE. « The Communicative Multiagent Team Decision Problem : Analyzing Teamwork Theories and Models ». *Journal of Artificial Intelligence Research* 16 (juin 2002), p. 389-423. ISSN : 1076-9757. DOI : 10.1613/jair.1024.

[24] T. SMITH et R. SIMMONS. « Heuristic Search Value Iteration for POMDPs ». Dans : *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*. UAI '04. Banff, Canada : AUAI Press, 2004, p. 520-527. ISBN : 0974903906.

[25] D. SZER, F. CHARPILLET et S. ZILBERSTEIN. « MAA* : A Heuristic Search Algorithm for Solving Decentralized POMDPs ». Dans : *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*. 2005, p. 576-583.

[26] N. TAO, J. BAXTER et L. WEAVER. « A Multi-Agent, Policy-Gradient approach to Network Routing ». Dans : *Proceedings of the Eighteenth International Conference on Machine Learning*. 2001.

A Notes sur les formalisations POMDP candidates

Nous regardons plus en détails comment, étant donnés le Dec-POMDP et les FSC de tous les agents sauf i, on peut dériver un POMDP meilleure-réponse valide du point de vue de l'agent i. Notons d'abord que, dans la formalisation POMDP :

- nous n'avons **aucun choix** pour l'observation (o_i^t) et pour l'action (a_i^t), qui sont des pré-requis ;
- le **seul choix** qui peut être fait est celui des variables mises dans l'état e_i^t ;
- les fonctions de transition, d'observation et de récompense sont des **conséquences** de ce choix.

Notons aussi que, dans une formalisation POMDP, les variables d'état et d'observation doivent être distinguées. On ne peut pas écrire que X est une variable d'état observée (contrairement au formalisme MOMDP). Dans ce qui suit, il est ainsi important de distinguer les différents types de variables aléatoires. En particulier,

- O_i^t dénote toujours (évidemment) une *variable d'observation*, mais
- $\tilde{O}_i^t$ dénote soit une *variable intermédiaire*[5], soit une *variable d'état* (qui, dans les deux cas, est fortement dépendante de la variable d'observation O_t^i puisque leurs valeurs sont toujours égales).

Le principal problème dans la conception de notre POMDP meilleure-réponse est de vérifier que les dépendances dans les fonctions de transition, d'observation, et de récompense sont appropriées (voir figure 3 (haut gauche)). Dans les paragraphes qui suivent, nous considérons 3 choix pour l'*état étendu* du POMDP, vérifions s'ils induisent effectivement des POMDP valides, et dérivons les fonctions de transition, d'observation et de récompense induites le cas échéant.

$e^t = \langle s^t, n_{\neq i}^t \rangle$ **? –** Pour montrer que $e^t = \langle s^t, n_{\neq i}^t \rangle$ n'induit pas un POMDP convenable, écrivons la fonction de transition :

$$
\begin{aligned}
&T_e(e^t, a_i^t, e^{t+1}) \stackrel{\text{def}}{=} Pr(e^{t+1}|e^t, a_i^t) \\
&= Pr(s^{t+1}, n_{\neq i}^{t+1}|s^t, n_{\neq i}^t, a_i^t) \\
&= \sum_{\tilde{o}^{t+1}} Pr(s^{t+1}, n_{\neq i}^{t+1}, \tilde{o}^{t+1}|s^t, n_{\neq i}^t, a_i^t) \\
&= \sum_{\tilde{o}^{t+1}} Pr(n_{\neq i}^{t+1}|s^t, n_{\neq i}^t, a_i^t, s^{t+1}, \tilde{o}^{t+1}) \\
&\quad \cdot Pr(s^{t+1}, \tilde{o}^{t+1}|s^t, n_{\neq i}^t, a_i^t) \\
&= \sum_{\tilde{o}^{t+1}} Pr(n_{\neq i}^{t+1}|s^t, n_{\neq i}^t, a_i^t, s^{t+1}, \tilde{o}^{t+1}) \\
&\quad \cdot Pr(\tilde{o}^{t+1}|s^t, n_{\neq i}^t, a_i^t, s^{t+1}) \cdot Pr(s^{t+1}|s^t, n_{\neq i}^t, a_i^t) \\
&= \sum_{\tilde{o}^{t+1}} \left(\prod_{j \neq i} \eta(n_j^t, \tilde{o}_j^{t+1}, n_j^{t+1}) \right) \\
&\cdot O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, \tilde{o}^{t+1}) \cdot T(s^t, \langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}),
\end{aligned}
$$

où $\tilde{O}^{t+1}$ est une variable temporaire, pas une variable d'état ou d'observation. Ici, comme illustré par la figure 3 (haut droite), le problème est que la variable d'observation O_i^{t+1} n'est pas indépendante de E^t étant donné E^{t+1} et A^t.

$e^t = \langle s^t, n_{\neq i}^t, \tilde{o}_i^t \rangle$ **? –** Nous corrigeons la première tentative en ajoutant une variable d'état $\tilde{O}_i^t$, définissant ainsi $e^t = \langle s^t, n_{\neq i}^t, \tilde{o}_i^t \rangle$, d'où :

$$
\begin{aligned}
&T_e(e^t, a_i^t, e^{t+1}) \stackrel{\text{def}}{=} Pr(e^{t+1}|e^t, a_i^t) \\
&= Pr(s^{t+1}, n_{\neq i}^{t+1}, \tilde{o}_i^{t+1}|s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t)
\end{aligned}
$$

5. Une telle variable n'apparaît usuellement pas dans le formalisme POMDP, mais est requise ici pour calculer les fonctions de transition et d'observation à l'aide du modèle Dec-POMDP et des FSC.

$$= \sum_{o_{\neq i}^{t+1}} Pr(s^{t+1}, n_{\neq i}^{t+1}, o_{i-1}^{t+1}, \tilde{o}_i^t | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t)$$
$$= \sum_{o_{\neq i}^{t+1}} Pr(n_{\neq i}^{t+1} | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t, s^{t+1}, o^{t+1})$$
$$\cdot Pr(s^{t+1}, o_{\neq i}^{t+1}, \tilde{o}_i^{t+1} | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t)$$
$$= \sum_{o_{\neq i}^{t+1}} Pr(n_{\neq i}^{t+1} | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t, s^{t+1}, o^{t+1})$$
$$\cdot Pr(o_{\neq i}^{t+1}, \tilde{o}_i^{t+1} | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t, s^{t+1}) \cdot Pr(s^{t+1} | s^t, n_{\neq i}^t, \tilde{o}_i^t, a_i^t)$$
$$= \sum_{o_{\neq i}^{t+1}} \Big(\prod_{j \neq i} \eta(n_j^t, o_j^{t+1}, n_j^{t+1}) \Big)$$
$$\cdot O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, o^{t+1}) \cdot T(s^t, \langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}).$$

La formule ci-dessus ne soulève pas le même problème que précédemment puisque $\tilde{O}_i^t$ est une variable d'état, et la variable d'observation O_i^t ne dépend plus maintenant de l'état précédent étant donné le nouveau et l'action (*cf.* figure 3 (bas gauche)). Nous avons aussi la fonction d'observation suivante :

$$O_e(a_i^t, e_i^{t+1}, o_i^{t+1}) \stackrel{\text{def}}{=} Pr(o_i^{t+1} | a_i^t, e_i^{t+1})$$
$$= Pr(o_i^{t+1} | a_i^t, \langle s^{t+1}, n_{\neq i}^{t+1}, \tilde{o}_i^{t+1} \rangle) = \mathbf{1}_{o_i^{t+1} = \tilde{o}_i^{t+1}},$$

et la fonction de récompense triviale :

$$r_e(e^t, a_i^t) = r(s^t, a_i^t, \psi_{\neq i}(n_{\neq i}^t)).$$

$\underline{e^t = \langle s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t \rangle}$ **?** – Dans ce travail, nous avons aussi considéré un troisième choix pour l'état étendu, défini comme $e^t = \langle s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t \rangle$, où $\tilde{O}_{\neq i}^t$ est une variable d'état (pas d'observation) correspondant aux observations des autres agents au temps t :

$$T_e(e^t, a_i^t, e^{t+1}) \stackrel{\text{def}}{=} Pr(e^{t+1} | e^t, a_i^t)$$
$$= Pr(s^{t+1}, n_{\neq i}^t, \tilde{o}_{\neq i}^{t+1} | s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$= Pr(\tilde{o}_{\neq i}^{t+1} | s^{t+1}, n_{\neq i}^t, s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$\cdot Pr(s^{t+1}, n_{\neq i}^t | s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$= Pr(\tilde{o}_{\neq i}^{t+1} | s^{t+1}, n_{\neq i}^t, s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$\cdot Pr(s^{t+1} | n_{\neq i}^t, s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$\cdot Pr(n_{\neq i}^t | s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t, a_i^t)$$
$$= Pr(s^{t+1} | s^t, n_{\neq i}^t, a_i^t) \cdot Pr(n_{\neq i}^t | n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t)$$
$$\cdot Pr(\tilde{o}_{\neq i}^{t+1} | s^{t+1}, n_{\neq i}^t, a_i^t)$$
$$= Pr(s^{t+1} | s^t, n_{\neq i}^t, a_i^t) \cdot Pr(n_{\neq i}^t | n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t)$$
$$\cdot \sum_{\tilde{o}_i^{t+1}} Pr(\tilde{o}_{\neq i}^{t+1}, \tilde{o}_i^{t+1} | s^{t+1}, n_{\neq i}^t, a_i^t)$$
$$= T(s^t, \langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}) \cdot \prod_{j \neq i} \eta(n_j^{t-1}, o_j^t, n_j^t)$$
$$\cdot \sum_{\tilde{o}_i^{t+1}} O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, \langle \tilde{o}_{\neq i}^{t+1}, \tilde{o}_i^{t+1} \rangle).$$

La formule ci-dessus ne soulève pas de problèmes. Comme illustré par la figure 3 (bas droite), la variable d'observation O_i^{t+1} dépend de E^t étant donnés E^{t+1} et A^t. Nous avons aussi la fonction d'observation suivante :

$$O_e(a_i^t, e_i^{t+1}, o_i^{t+1}) \stackrel{\text{def}}{=} Pr(o_i^{t+1} | a_i^t, e_i^{t+1})$$
$$= Pr(o_i^{t+1} | a_i^t, \langle s^{t+1}, n_{\neq i}^t, \tilde{o}_{\neq i}^{t+1} \rangle)$$
$$= \frac{Pr(\tilde{o}_{\neq i}^{t+1}, o_i^{t+1}, s^{t+1}, n_{\neq i}^t, a_i^t)}{Pr(\tilde{o}_{\neq i}^{t+1}, s^{t+1}, n_{\neq i}^t, a_i^t)}$$
$$= \frac{Pr(\tilde{o}_{\neq i}^{t+1}, o_i^{t+1} | s^{t+1}, n_{\neq i}^t, a_i^t)}{\sum_{o_i^{t+1}} Pr(\tilde{o}_{\neq i}^{t+1}, o_i^{t+1} | s^{t+1} n_{\neq i}^t, a_i^t)}$$
$$= \frac{O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, \langle \tilde{o}_{\neq i}^{t+1}, o_i^{t+1} \rangle)}{\sum_{\tilde{o}_i^{t+1}} O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, \langle \tilde{o}_{\neq i}^{t+1}, \tilde{o}_i^{t+1} \rangle)}.$$

On peut voir que le dénominateur est identique à la dernière partie de la fonction de transition. En pratique, lors du calcul des probabilités de transition, nous stockerons les valeurs pour $\sum_{\tilde{o}_i^{t+1}} O(\langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle, s^{t+1}, \langle \tilde{o}_{\neq i}^{t+1}, \tilde{o}_i^{t+1} \rangle)$ de manière à les réutiliser los du calcul de la fonction d'observation. Finalement, la fonction de récompense est obtenue avec :

$$r_e(e^t, a_i^t) = r(\langle s^t, n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t \rangle, a_i^t)$$
$$= \sum_{n_{\neq i}^t} Pr(n_{\neq i}^t | n_{\neq i}^{t-1}, \tilde{o}_{\neq i}^t) \cdot r(s^t, n_{\neq i}^{t-1}, a_i^t)$$
$$= \sum_{n_{\neq i}^t} \Big(\prod_{j \neq i} \eta(n_j^{t-1}, \tilde{o}_j^t, n_j^t) \Big) \cdot r(s^t, \langle \psi_{\neq i}(n_{\neq i}^t), a_i^t \rangle).$$

Différentes formalisations conduisent à différents espaces d'états (étendus) avec différentes tailles, de sorte que les complexités en temps et en espace de la génération de ce POMDP meilleure-réponse ou de sa résolution dépendront de ce choix de formalisation. Le meilleur choix dépendra du Dec-POMDP considéré, et éventuellement des FSC courants. Nous avons opté pour la plus simple des deux formalisations présentées ci-dessus (que nous appelons "formalisation MOMDP" parce qu'une des variables d'état est complètement observable), mais avons observé peu de différences en pratique dans nos expérimentations.

B Figures supplémentaires

Concernant l'**élimination d'états** (comme expliquée en section 4.2), il se trouve que, pour le POMDP meilleure-réponse "MOMDP", le ratio du nombre initial d'états (étendus) sur le nombre d'états après élimination dépend du problème considéré mais pas de l'exécution de inf-JESP (voir figure 4). Ce ratio est de 1 pour DecTiger, 9 pour Grid, et 2 pour Recycling. Ces différences sont dues à la stochasticité du processus d'observation, laquelle limite voire prévient l'élimination d'états. Actuellement, l'élimination d'états repose sur la probabilité de générer une observation donnée depuis un état en considérant les actions possibles. Toutefois, pour le problème DecTiger, chaque observation

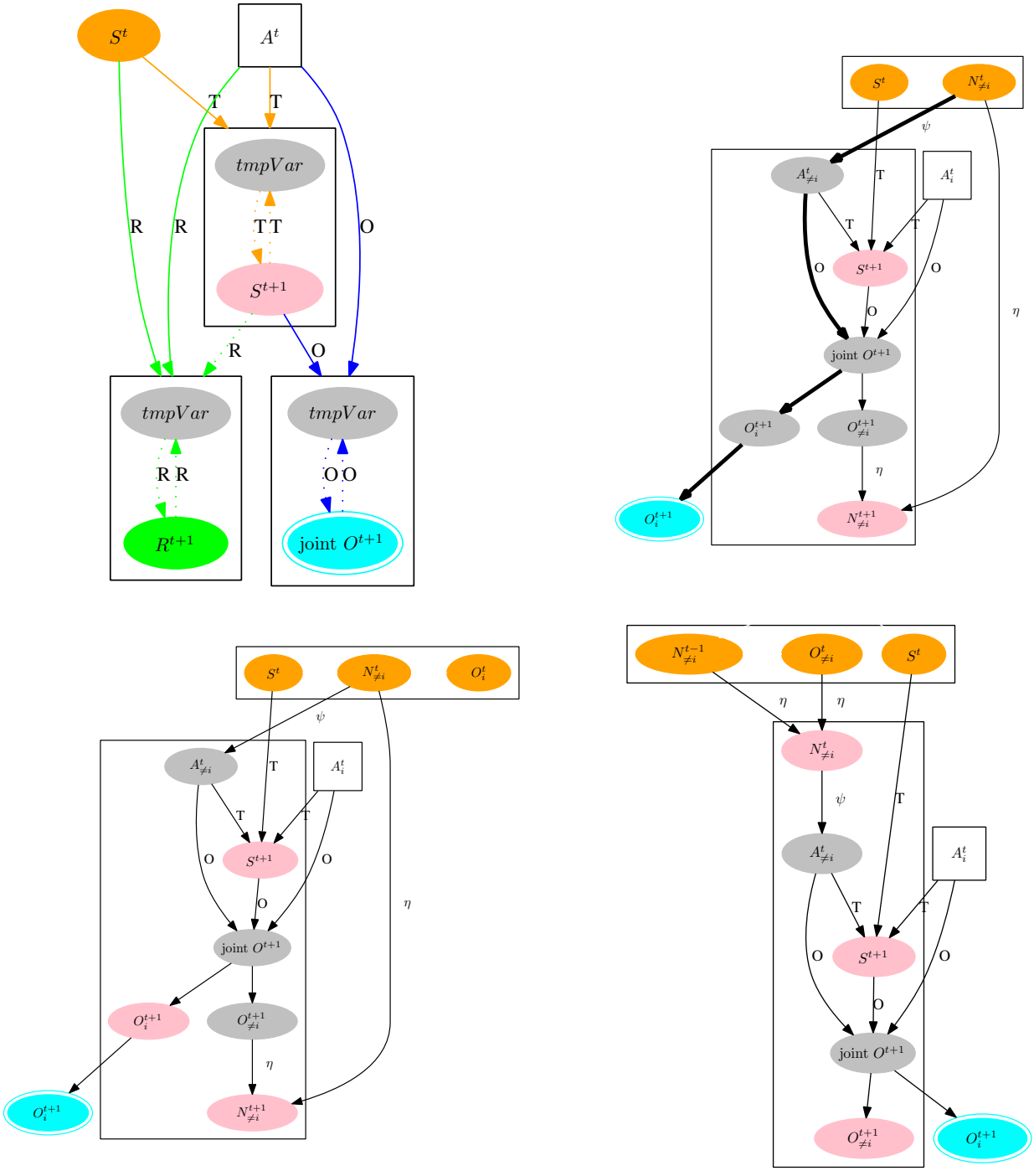

FIGURE 3 – (haut gauche) Dépendances standard d'un POMDP, incluant des variables intermédiaires employées pour calculer la fonction de transition (seulement) ; et 2 formalisations candidates pour un POMDP meilleure-réponse avec : (haut droite) $e^t_i \stackrel{\text{def}}{=} \langle s^t, n^t_{\neq i} \rangle$, (bas gauche) $e^t_i \stackrel{\text{def}}{=} \langle s^t, n^t_{\neq i}, o^t_i \rangle$, et (bas droite) $e^t_i \stackrel{\text{def}}{=} \langle s^t, n^{t-1}_{\neq i}, o^t_{\neq i} \rangle$.

peut être générée quelle que soit l'action considérée, ce qui empêche toute élimination d'état (contrairement aux problèmes Grid et Recycling).

Concernant la **taille des FSC finaux** obtenus après convergence d'une exécution d'inf-JESP avec initialisation aléatoire (voir figure 5), nous avons aussi observé différents comportements. Appliqué au problème DecTiger, inf-JESP conduit à une distribution très dispersée des tailles des FSC finaux. Appliqué au problème Recycling, inf-JESP conduit aussi à une distribution dispersée des tailles des FSC finaux, mais les tailles des FSC des deux agents sont symétriques. Appliqué au problème Grid, inf-JESP génère un grand nombre de paires de FSC de taille 10 avec parfois d'énormes variations et avec une tendance à l'assymétrie,

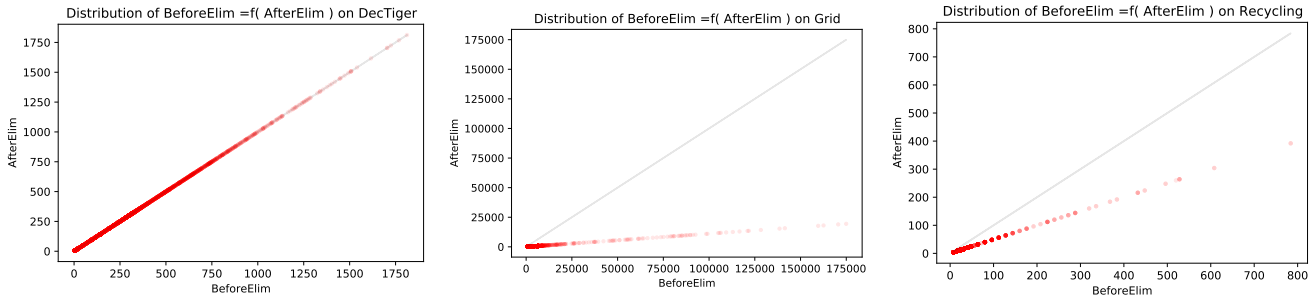

FIGURE 4 – Effet de l'élimination d'états pour les problèmes DecTiger, Grid et Recycling (de gauche à droite). Chaque figure trace, pour chaque POMDP meilleure-réponse obtenu en exécutant inf-JESP(R-1_1), le nombre d'états de ce POMDP après élimination en fonction de ce nombre avant élimination.

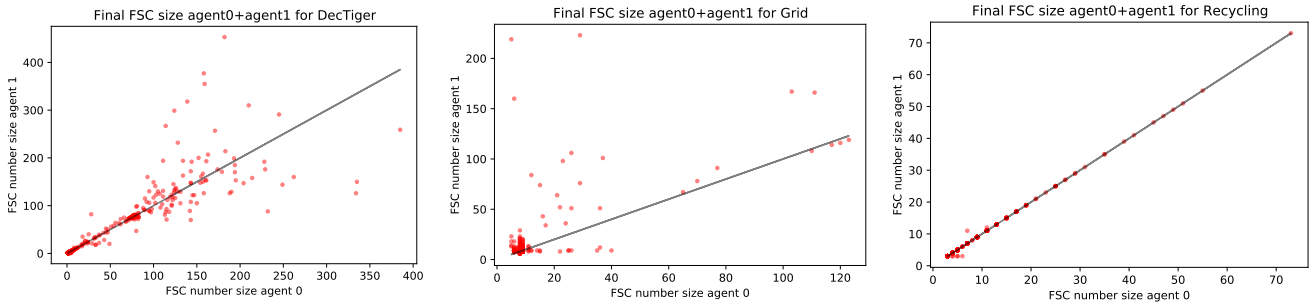

FIGURE 5 – Tailles de FSC finaux obtenus avec inf-JESP(R-1_1) après convergence pour les problèmes DecTiger, Grid et Recycling (de gauche à droite). chaque point correspond à une paire contenant les tailles des FSC des deux agents.

le premier agent optimisé ayant une tendance à avoir plus de nœuds dans son FSC. Ces résultats nécessitent des investigations supplémentaires pour les comprendre clairement et voir s'il est possible de les lier à la nature du domaine abordé.

Il faut aussi noté que les **tailles des FSC** ne croissent pas de manière monotone pendant une itération d'inf-JESP (données non présentées ici). Parfois la taille du FSC calculé pendant une amélioration d'inf-JESP décroît, ce qui signifie que la solution au POMDP meilleure-réponse est un FSC de plus petite taille mais de plus grande valeur (comme observé couramment dans le problème DecTiger). En regardant les FSC obtenus à la fin d'une exécution d'inf-JESP, un autre phénomène observé est que plus le nombre d'itérations requis pour atteindre l'équilibre est grand, plus petits sont les FSC finaux (voir figure 6). Mais cela ne signifie pas que la valeur associée est plus grande (voir figure 7).

Finalement, la figure 8 présente les valeurs obtenues en fonction de la somme des tailles des FSC obtenus par inf-JESP(R-1_1) après convergence. On peut observer que de petits FSC semblent suffisants pour générer une valeur élevée, ce qui ouvre de nouvelles directions combinant inf-JESP avec la compression de FSC.

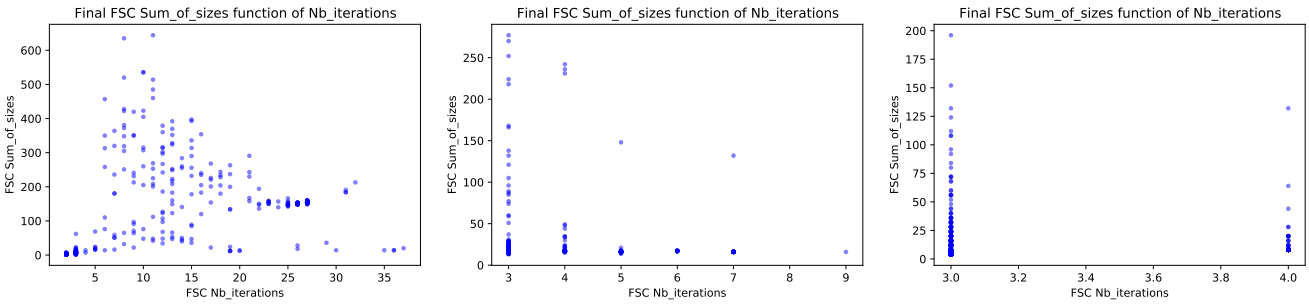

FIGURE 6 – Somme des tailles des FSC finaux obtenus avec inf-JESP(R-1_1) après convergence en fonction du nombre d'itérations requis pour les problèmes DecTiger, Grid et Recycling (de gauche à droite).

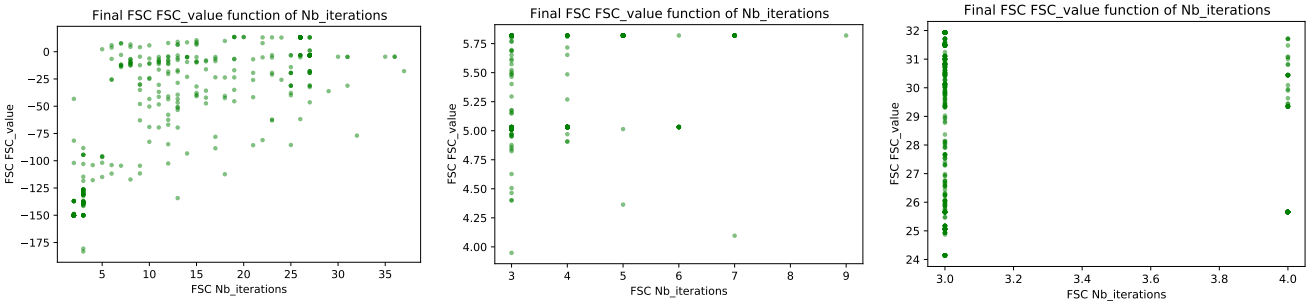

FIGURE 7 – Valeur des FSC finaux obtenus avec inf-JESP(R-1_1) après convergence en fonction du nombre requis d'itérations pour les problèmes DecTiger, Grid et Recycling (de gauche à droite).

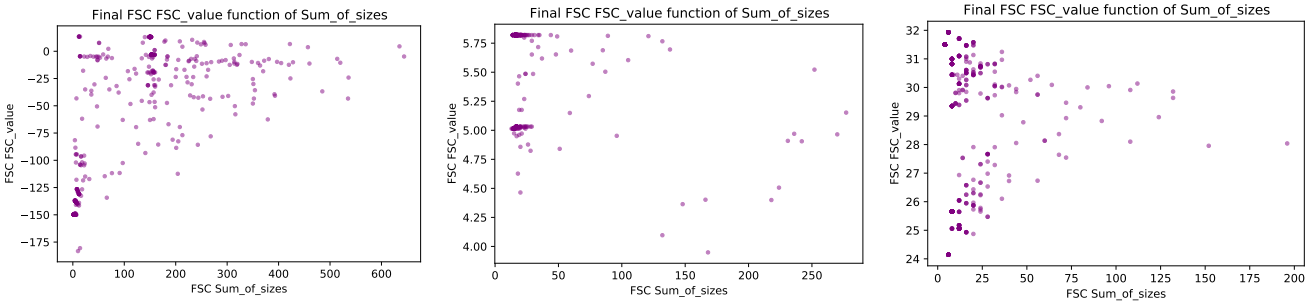

FIGURE 8 – Valeur des FSC finaux obtenus avec inf-JESP(R-1_1) après convergence en fonction de la somme des tailles des FSC finaux pour les problèmes DecTiger, Grid et Recycling (de gauche à droite).