Résolution approchée par décomposition de processus décisionnels de Markov appliquée à l'exploration en robotique mobile

Guillaume Lozenguez[1], Lounis Adouane[2], Aurélie Beynier[3], Philippe Martinet[4] et Abdel-Illah Mouaddib[1]

[1] GREYC, Université de Caen Basse-Normandie - prenom.nom@unicaen.fr

[2] Institut Pascal, Université Blaise Pascal, Clermont-Ferrand
lounis.adouane@univ-bpclermont.fr

[3] LIP6, Université Pierre & Marie Curie, Paris - aurelie.beynier@lip6.fr

[4] IRCCYN, Ecole Centrale de Nantes - Philippe.Martinet@irccyn.ec-nantes.fr

Résumé : Dans cet article, nous nous intéressons à l'exploration d'une zone par un robot mobile autonome. Notre approche procède à des re-calculs réguliers des politiques en utilisant des processus décisionnels de Markov, au fur et à mesure que la connaissance du robot est actualisée. Dans la mesure où, l'exploration conduit, à chaque instant, à une explosion combinatoire des possibilités sur les politiques d'actions. Le problème de planification par résolution d'un MDP large décomposé est soulevé avec une contrainte forte sur la rapidité des calculs mis en œuvre. L'approche proposée cherche à partitionner l'espace des états du robot de façon à construire, ensuite, une hiérarchie de MDP : un MDP abstrait haut niveau et plusieurs MDPs locaux aux régions construites. L'approche est ensuite évaluée par plusieurs expérimentations statistiques. L'originalité consiste en la capacité de l'agent à hiérarchiser l'intérêt de chaque zone à explorer relativement à toutes les autres pour optimiser la politique construite à chaque modification de la carte.

Mots-clés : Exploration, Robotique Mobile, Processus Décisionnel de Markov et Décomposition

1 Introduction

Dans cet article, nous nous intéressons à l'exploration d'une zone par un robot mobile autonome [1]. Notre approche procède à des re-calculs réguliers des politiques, au fur et à mesure que la connaissance est actualisée [**????**]. En effet, nous montrerons qu'une modélisation, même simplifiée, d'un problème d'exploration implique des espaces de recherche qui sont intraitables de part leurs volumes. À chaque modification de sa connaissance, le robot a pour charge de recalculer sa politique d'action visant à visiter l'ensemble des zones inconnues.

Ainsi, en construisant un modèle décisionnel pour l'agent, le problème s'apparente à une succession de problèmes du "voyageur de commerce" résolus avec des ressources restreintes (temps de calcul de l'ordre de la seconde avec l'ordinateur embarqué sur le robot). Le problème décisionel individuel peut s'exprimer à l'aide d'un processus décisionnel de Markov orienté par plusieurs objectifs (Goal Oriented MDP, GO-MDP). Nous verrons que cette modélisation conduit à des MDP excessivement larges, c'est-à-dire que même une simple énumération des états serait fastidieuse. Le problème de planification par résolution de MDPs larges est envisagé avec une contrainte forte sur la rapidité des calculs mis en œuvre.

L'approche que nous proposons se base sur le modèle des processus décisionnels de Markov décomposés [**?**] et cherche à partitionner un MDP définissant la dynamique du système (sans intégrer les objectifs) de façon à construire, ensuite, une hiérarchie de GO-MDPs. Cette solution permet de restreindre la résolution d'un problème multi-objectif au calcul de seulement quelques politiques locales, région par région.

1. Ces travaux sur la résolution hiérarchique de la planification des déplacements, intégrés dans ma thèse [**?**], ont été présentés dans un cadre multi-robot lors de la conférence internationale PAAMS (Practical Application of Agent and Multi-Agent Sytems) [**?**] et dans une version approfondie dans le journal international MAGS (Multi-Agent and Grid Systems) [**?**]. Ici, nous laissons de côté l'aspect multi-agent pour discuter de la décomposition d'un MDP dans notre cadre applicatif.

Nous décrirrons des expérimentations qui permettent d'évaluer statistiquement la qualité et la rapidité d'une résolution hiérarchique approchée.

2 Exploration en robotique mobile

L'exploration consiste à permettre à un robot de se déplacer dans son environnement de façon à acquérir une connaissance (sous forme d'une carte) sur les zones qui doivent être visitées. Pour valider la bonne couverture des zones à visiter, le robot doit maintenir une connaissance sur sa localisation lors de ses déplacements. Pour chacun de ces états possibles, le robot doit choisir une action de déplacement pour lui permettre d'étendre sa carte connue. Quelle que soit la représentation de son environnement, grille d'occupation ou carte topologique [**??**], ce choix est matérialisé par une position à atteindre parmi l'ensemble des positions frontières (à la limite des zones connues et inconnues).

La problématique de planifier une exploration peut être considérée comme double avec une phase de planification hiérarchisant les zones à visiter et une phase de planification du chemin permettant de naviguer entre les zones à visiter. La fin d'une mission d'exploration est définie lorsqu'il n'existe plus de zones atteignables à visiter. Le principe des algorithmes de planification repose généralement sur l'évaluation d'un nombre important de politiques pour converger vers la meilleure. Cependant, une énumération des politiques en considérant tous les chemins possibles reliant toutes les zones à visiter et en considérant toutes les possibilités dans la construction de la carte, va s'avérer fastidieuse et incalculable.

2.1 Modélisation générale

Concrètement l'état s d'un robot à l'instant t est défini par sa carte courante M_s (Map) et sa position p_s. La position du robot appartient à l'ensemble des positions P_s définies par la carte M_s. Une carte M_s permet de lier un ensemble de positions P_s entre elles via un ensemble de connexions C_s.

$$s = (M_s, p_s) \quad | \quad M_s = \{P_s, C_s\}, \quad p_s \in P_s$$

L'espace des états correspond à l'ensemble de tous les états atteignables à partir de l'état courant. La taille de l'espace des états dépend alors de la taille et du nombre de cartes potentielles qui peuvent être construites. La carte et la position du robot sont définies de façon différente en fonction de la représentation de l'environnement choisi (grille d'occupation ou carte topologique).

De façon à borner le nombre de cartes potentielles, on définit l'ensemble P comme l'ensemble maximal de toutes les positions potentielles modélisables dans la carte du robot. C'est-à-dire que, quelque soit un état s à un instant donné, P_s (défini par la carte courante M_s) est un sous-ensemble de P. En considérant P fini et dénombrable, (et sans tenir compte des configurations possibles des connexions C), l'ensemble des cartes potentielles $\mathbb{M}$ est défini sur $2^{|P|}$ possibilités (chaque position de P est atteignable ou non). L'ensemble des états possibles S du robot est défini comme le produit cartésien des cartes potentielles et des positions possibles du robot pour chaque carte :

$$|S| \simeq 2^{|P|}|P|$$

Ce cadre de modélisation relativement simpliste permet tout de même de conclure que la taille de l'espace des états liés au nombre de cartes potentielles sera exponentiel par rapport au nombre maximal de positions qui peuvent être découvertes $|P|$.

2.2 Vers des re-planifications régulières

Il est raisonnablement impossible de planifier les actions à réaliser, de la configuration initiale du robot, jusqu'à l'exploration totale d'une zone en tenant compte de toutes les évolutions possibles de la carte. La plupart des stratégies d'exploration se basent sur une résolution à horizon limité sur l'évolution de la carte (incluant un nombre limité de modifications possibles dans la carte) [**????**].

La politique des robots (ensemble des actions de déplacement à réaliser) consiste alors à viser des états objectifs permettant d'étendre la connaissance courante soit, les positions frontières entre les zones connues et inconnues. Cette politique est ensuite actualisée au fur et à mesure que la connaissance s'accroît. La difficulté de la mise en œuvre d'une telle stratégie d'exploration réside dans les mécanismes permettant au

robot de hiérarchiser, en cours de mission, l'importance de chacune des positions frontières courantes par rapport aux autres.

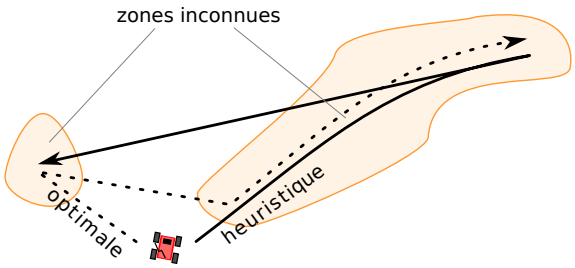

FIGURE 1 – Exemple où l'heuristique visant la frontière la plus proche ne permet pas une stratégie d'exploration optimale.

La Figure 1 met en évidence que la taille et la configuration des zones frontières doivent être prises en compte dans le calcul de la politique si l'on se compare à une solution heuristique privilégiant uniquement la frontière la plus proche. Une évaluation fine de l'intérêt de viser une frontière plutôt qu'une autre passe alors par une planification de chemin englobant plusieurs frontières à explorer [**?**]. Ce constat appelle alors une planification multi-objectif pour pouvoir éviter au robot des retours coûteux sur des zones laissées loin derrière.

2.3 Processus décisionnel de Markov (MDP)

La problématique de la planification pour l'exploration relève d'un problème d'optimisation de chemin permettant de visiter plusieurs zones frontières. de plus, en considérant des erreurs possibles lors des déplacements, la planification se fait dans un cadre incertain. Les Processus Décisionnels de Markov (MDP) sont considérés comme un outil mathématique de référence pour modéliser des problèmes d'optimisation de décision sous incertitude. Le modèle est utilisé dans de nombreux travaux pour la planification de chemin en robotique mobile dont voici un échantillon : [**????**]. Il permet de modéliser l'incertitude par des probabilités de transition et d'exprimer les performances par une fonction de coût/récompense liée à la réalisation des actions.

Un MDP est défini par un tuple $\langle S, A, t, r\rangle$ [**??**] représentant respectivement les ensembles des états et des actions, les fonctions de transition et de récompense. Ce tuple définit le système et ses possibilités d'évolution. En robotique mobile, les états vont être construits sur la base de la carte du robot à un instant donné et des actions unitaires permettant au robot de passer d'une position à une autre dans sa carte. La fonction de transition t est définie par $t : S \times A \times S \to [0, 1]$ et donne la probabilité $t(s, a, s')$ d'atteindre s' depuis s en exécutant l'action a. La fonction de coût/récompense r est définie par $r : S \times A \to \mathbb{R}$, où $r(s,\ a)$ retourne la récompense obtenue en exécutant l'action a depuis s si $r(s,\ a)$ est positif ou le coût d'exécuter a en s si $r(s,\ a)$ est négatif.

Trouver une solution optimale à un MDP consiste à chercher la politique optimale $\pi^* : S \to A$ qui maximise les gains espérés sur les récompenses. Bellman [**?**] a proposé une fonction permettant une évaluation récursive des états par rapport à π la politique employée $V^\pi : S \to \mathbb{R}$. La valeur d'un état $V^\pi(s)$ est alors définie comme une somme pondérée de sa récompense immédiate après réalisation de l'action $\pi(s)$ et des valeurs des états atteignables proportionnellement aux probabilités de les atteindre :

$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} t(s, \pi(s), s') V^\pi(s')$$

Le paramètre $\gamma \in [0, 1]$ pondère l'importance entre les récompenses immédiates et futures.

2.4 Exploration : un MDP orienté par des objectifs

Dans le cadre de l'exploration d'une zone par des robots mobiles, l'utilisation des Processus Décisionnels de Markov (MDP) à horizon limité sur les évolutions des connaissances impose de matérialiser les frontières à atteindre comme des objectifs intermédiaires. Un coût est associé à chacune des actions de déplacement

relativement à l'énergie ou au temps consommé. Des récompenses sont également associées à chaque action permettant de visiter une zone inconnue [**??**].

Dans l'idée de choisir la frontière à repousser en considérant les frontières laissées derrière le robot, il est alors nécessaire d'ajouter une mémoire des objectifs atteints ou non. En considérant un MDP $\langle S_D, A_D, t_D, r_D \rangle$ modélisant la dynamique des déplacements du robot à un instant donné, un MDP $\langle S, A, t, r \rangle$ orienté par des objectifs (GO-MDP) est construit en enrichissant chaque état $s_D \in S_D$ du robot dans sa carte d'une mémoire des objectifs qui ont été atteints G_s (avec $S_D = P$ et $A_D = A = C$, les ensembles de positions et de déplacements). L'ensemble G_s des objectifs atteints est un sous-ensemble de tous les objectifs G. La mission s'arrête lorsque tous les objectifs sont atteints (soit : $G_s = G$).

$$S = \{s \quad | \quad s = (s_D,\, G_s), \quad s_D \in S_D, \quad G_s \subset G\}$$

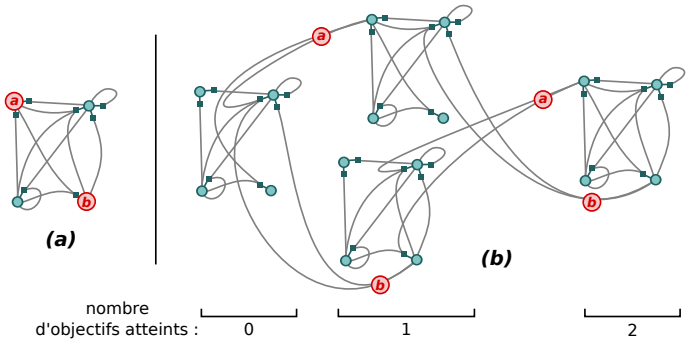

FIGURE 2 – Exemple de GO-MDP (b) construit sur la base du modèle illustré en (a) incluant un ensemble de 2 objectifs $\{a,\, b\}$.

Le système est donc défini sur $2^{|G|}|S_D|$ états (Figure 2) en considérant toutes les successions possibles dans l'ordre de la réalisation des objectifs ($2^{|G|}$ possibilités pour G).

Sur la base d'une connaissance exprimée sur les déplacements sous forme d'une fonction de transition $t_D : S_D \times A \times S_D \to [0,\, 1]$ et une fonction de récompense $r_D : S_D \times A \to \mathbb{R}^-$ exprimant les coûts des déplacements, il est possible de construire les fonctions de transition t et de récompense r définies sur l'espace d'états du GO-MDP.

La fonction de transition t retourne la probabilité d'atteindre l'état incluant la position suivante et l'ensemble des objectifs actualisés en accord avec les transitions données par les déplacements. L'ensemble des objectifs atteints est alors augmenté si la position atteinte est un objectif.

$$t(s = (s_{Ds},\, G_s),\, a,\, s' = (s_{Ds'},\, G_{s'})) = \begin{cases} t_D(s_{Ds},\, a,\, s_{Ds'}) & \text{si} \quad G_{s'} = G_s \cup (G \cup s') \\ 0 & \text{sinon} \end{cases}$$

D'une façon similaire, la fonction de récompense est construite relativement aux coûts de déplacement et à un gain perçu pour visiter une nouvelle frontière. Le gain est défini pour chacun des objectifs. Pour l'exploration, le gain peut être défini de façon homogène pour toutes les positions frontières ($gain$ = constante). Pour garantir que toutes les frontières soient visitées, la valeur de gain doit être supérieure au coût cumulé pour traverser l'environnement de part et d'autre. Le gain espéré est alors proportionnel aux chances qu'une frontière soit effectivement visitée après réalisation de l'action de déplacement.

$$r(s = ((s_{Ds},\, G_s),\, a)) = r_D(s_{Ds},\, a) + gain \sum_{s_D \in G,\; s_D \notin G_s} t_D(s_{Ds},\, a,\, s_D)$$

A chaque pas de temps, la politique optimale construite dicte au robot le chemin le plus court à suivre, lui permettant de visiter l'ensemble des frontières. Cette stratégie permet d'évaluer l'intérêt immédiat d'une frontière vis à vis de toutes les autres. Cependant, la politique doit toujours être actualisée pour toutes modifications de la carte.

La mise en œuvre de cette stratégie nécessite de nombreux calculs de politiques en-ligne or un GO-MDP est construit sur $2^{|G|}|S_D|$ états. En considérant la borne de quelques millions d'états [**???**] comme borne

maximale pour la résolution optimale d'un MDP, l'utilisation d'un processus décisionnel de Markov orienté par plusieurs objectifs est limité à 18 objectifs. Cependant, cette solution serait parmi les solutions applicables les plus complètes en terme d'expressivité du modèle pour un problème d'exploration en robotique.

3 Processus de Markov décomposés

L'idée maîtresse des MDPs décomposés consiste à agréger les états fortement connectés en sous MDPs de façon à décomposer le calcul des politiques [**??**]. La décomposition d'un MDP permet alors d'accélérer le calcul de politiques optimales en construisant une hiérarchie entre des problèmes locaux et une solution globale.

L'algorithme de référence "Value iteration" [**?**] consiste en des ré-évaluations approximatives successives de la valeur des actions en chaque état jusqu'à stabilisation de la fonction de valeur $V^\pi : S \to \mathbb{R}$. La résolution optimale d'un MDP demande un nombre d'opérations polynômial par rapport à la taille du problème ($|S|$ et $|A|$) [**?**]. En théorie, avec une décomposition homogène en k parties, on devrait pouvoir passer de la résolution d'un MDP large à k résolutions polynômiales de sous-MDPs de tailles divisées par k.

3.1 Formalisme

Un Processus Décisionnel de Markov Décomposé (DMDP) est défini par un tuple $\langle S, A, t, r, \mathbb{P}\rangle$ avec $\langle S, A, t, r\rangle$ définissant un MDP standard et $\mathbb{P}$ son partionnement. $\mathbb{P}$ est défini par l'ensemble des k régions qui divisent l'espace d'états S de façon à former une partition (Figure 3). C'est-à-dire que chaque état s de S appartient à une et une seule région. Dans les faits, il est nécessaire que chaque état appartienne à une unique région pour que la résolution couvre l'ensemble des états sans intersection entre les politiques locales.

$$\mathbb{P} = \{R_i \mid R_i \subset S\}, \quad \bigcup_{R_i \in \mathbb{P}} R_i = S \quad \text{et} \quad \forall R_i, R_j \in \mathbb{P}, \quad (R_i \cap R_j) = \emptyset$$

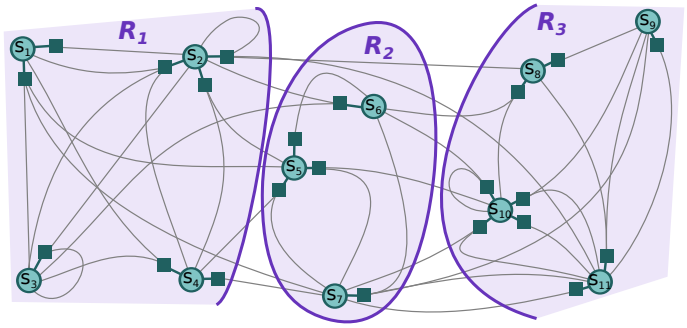

FIGURE 3 – Exemple de Processus Décisionnel de Markov à 11 états décomposé en 3 régions.

A partir d'un MDP décomposé en k régions, il est possible de construire un sous-MDP $\langle R_i, S, A, t, r\rangle$ pour chaque région R_i de la partition $\mathbb{P}$. La résolution optimale d'un sous-MDP consiste alors à itérer, avec "Value iteration" par exemple, uniquement sur les états s contenus par la région R_i.

Cependant, certains états atteignables à partir de R_i et utiles au calcul de la valeur des états de R_i n'appartiennent pas à la région. Ces états sont définis comme les états périphériques (Perifery) à R_i. Inversement, les états entrées de R_i (Entrance) forment l'ensemble des états de R_i directement atteignables depuis les états des autres régions. Nous noterons ainsi $S_{Perifery}$, l'ensemble de tous les états périphériques quelle que soit la région considérée :

$$Perifery(R_i) = \{s \mid s \in S, \quad s \notin R_i, \quad \exists s' \in R_i, \quad \exists a \in A, \quad t(s',\ a,\ s) > 0\}$$

$$Entrance(R_i) = \{s \mid s \in R_i, \quad \exists s' \in S, \quad s' \notin R_i, \quad \exists a \in A, \quad t(s',\ a,\ s) > 0\}$$

$$S_{Perifery} = \bigcup_{R_i \in \mathbb{P}} Perifery(R_i) \quad \text{et} \quad Entrance(R_i) = S_{Perifery} \cap R_i$$

3.2 Résolution itérative et hiérarchique

Construire la politique globale avec un Processus Décisionnel de Markov Décomposé (DMDP) nécessite de calculer au moins une politique par sous-MDP. Plusieurs calculs de politiques par sous-MDP sont alors nécessaires à l'ajustement des politiques locales entre elles. Il existe deux approches pour coordonner les politiques locales : une approche itérative et une approche hiérarchique.

L'approche itérative consiste à re-calculer la politique d'une région au fur et à mesure que de nouvelles valeurs sont connues pour une des régions voisines. L'actualisation des valeurs périphériques se fait en même temps que les politiques locales sont affinées. De façon similaire à "Value iteration", un algorithme itératif "Region Iteration" va parcourir l'ensemble des régions jusqu'à ce que les améliorations sur la fonction de valeur soient inférieures à un seuil donné ϵ [**?**]. Dans le cadre de problèmes avec une certaine structure topologique sur les transitions entre états, un partitionnement cohérent permet à l'algorithme itératif de converger rapidement vers la politique globale optimale. L'algorithme n'aura alors besoin que de quelques itérations (à l'échelle du nombre total d'états $|S|$) pour converger.

L'approche hiérarchique consiste à calculer un certain nombre de politiques locales pour chaque région puis, à sélectionner les politiques locales permettant une politique globale cohérente. On a besoin alors d'un MDP abstrait construit sur l'ensemble des régions pour coordonner les politiques locales entre elles. Un MDP abstrait est un MDP classique $\langle S_A, A_A, r_A, t_A \rangle$ avec S_A et A_A respectivement, l'ensemble des macro-états et l'ensemble des macro-actions. Un macro-état est défini pour chaque région $R_i \in P$ du MDP décomposé $\langle S, A, t, r, \mathbb{P} \rangle$. Les macro-actions correspondent aux politiques localement applicables connues. Les fonctions de transition t_A et de récompenses r_A définissent la dynamique du système abstrait en fonction des politiques locales exécutées [**?**].

La définition d'un MDP abstrait est basée sur la connaissance de politiques locales. Cette connaissance est construite par le pré-calcul d'un certain nombre de politiques locales optimales pour chaque région R_i. Ce pré-calcul est effectué via plusieurs vecteurs de valeurs hypothétiques définis pour les états périphériques à la région R_i ($Perifery(R_i)$). Ainsi, une résolution hiérarchique d'un DMDP consiste à : (1) définir un ensemble de vecteurs de valeurs possibles pour chaque région ; (2) calculer les politiques locales optimales pour chaque région et pour chaque vecteur de valeurs ; (3) construire et résoudre un MDP abstrait.

Les deux approches reposent sur des philosophies différentes avec des caractéristiques différentes en terme d'efficacité et d'optimalité. L'efficacité se mesure ici en terme de temps de calcul nécessaire à la construction de la politique globale. Elle dépend directement de la taille des régions et du nombre de politiques locales calculées par région. La résolution itérative garantira l'optimalité mais, pas l'efficacité et inversement, la solution hiérarchique garantie un bon contrôle de l'efficacité, mais pas de l'optimalilté.

3.3 Partitionnement

Les algorithmes de résolution d'un MDP décomposé se basent sur le partitionnement en régions pour accélérer la résolution. La difficulté consiste alors à partitionner l'espace d'état d'une façon cohérente avec la future résolution du DMDP. C'est-à-dire que la partition optimale sera la partition permettant la résolution la plus rapide et/ou offrant les meilleures garanties sur l'optimalité de la solution (pour une résolution itérative ou hiérarchique). Il est alors incohérent (et impossible) de calculer systématiquement la politique globale pour chaque partition possible afin de valider l'optimalité ou non d'une partition. De ce fait, les critères de partitionnement reposent sur des hypothèses *a priori* [**??**] : équilibre sur la taille des régions et du nombre de régions, minimisation de la somme des transitions coupées ou minimisation de l'ensemble des états périphériques ($S_{Perifery}$).

Une fois le critère (ou fonction objectif) défini, le problème de la détermination de la partition optimale d'un graphe est connu pour être, lui-même, NP-difficile [**??**]. Le partitionnement d'un MDP large, en vue de sa résolution, n'est donc intéressant que s'il existe des heuristiques rapides, permettant un partitionnement efficace. Malgré cela, cette méthode combinée à la résolution du DMDP ne permet pas de traiter, en cours de mission, un GO-MDP avec $2^{|G|}$ états pour G. Il convient alors de proposer une heuristique de partitionnement qui s'appuie sur la structure du problème pour éviter l'énumération de tous les états lors de cette phase.

4 Partitionnement Glouton

L'approche que nous adoptons pour l'algorithme de partitionnement consiste à diviser directement le MDP $\langle S_D,\ A,\ t_D,\ r_D \rangle$ définissant la dynamique du système sans considérer les objectifs G puis, à générer le modèle de planification hiérarchique multi-objectif.

Le partitionnement cherche à minimiser la coupe sur l'ensemble des transitions et à équilibrer le nombre d'objectifs par région. En effet, la taille des sous-MDPs construits par la décomposition du GO-MDP est exponentielle par rapport au nombre d'objectifs locaux. Ainsi, un nombre équilibré d'objectifs par région permet d'équilibrer la taille des futurs GO-MDPs locaux.

La partition en soi n'est pas la solution finale mais un biais mis en place afin de conduire à un processus décisionnel rapide. Aussi, un algorithme glouton est proposé pour privilégier l'économie de ressources de calcul au détriment de la qualité de la partition. Le principe de l'algorithme glouton repose sur une construction successive des régions. Une région est elle-même construite par additions successives d'états. A chaque itération, l'état optimisant un critère local est alors sélectionné pour ajout.

4.1 Critère sur le voisinage local

Lors de la construction d'une région $R_i \in \mathbb{P}$, l'état le plus intéressant à ajouter est l'état qui maximise la somme des transitions le connectant à un des états de la région et qui minimise la somme des transitions le connectant à un des états non encore sélectionnés dans aucune région. $\mathbb{P}$, pendant le partitionnement, ne couvre pas la totalité de S_D mais uniquement, les états précédemment sélectionnés. En d'autres termes, l'état $s_D^* \in S_D - \mathbb{P}$ à ajouter, parmi les états non encore sélectionnés, maximise le ratio entre l'ensemble des transitions intérieures à la région $t_{in}(R_i,\ s_D)$ et l'ensemble des transitions extérieures aux régions du partitionnement courant $t_{out}(\mathbb{P},\ s_D)$.

$$s_D^* = \underset{s_D \in S_D - \mathbb{P}}{argmax}\ (\ ratio(\mathbb{P},\ R_i,\ s_D)\ ), \quad ratio(\mathbb{P},\ R_i,\ s_D) = \frac{t_{in}(R_i,\ s_D)}{t_{out}(P,\ s_D)}$$

$$t_{in}(R_i,\ s_D) = \sum_{s'_D \in R_i} (t_D(s_D,\ a,\ s'_D) + t_D(s'_D,\ a,\ s_D))$$

$$t_{out}(\mathbb{P},\ s_D) = \sum_{s'_D \in S_D - \mathbb{P}} (t_D(s_D,\ a,\ s'_D) + t_D(s'_D,\ a,\ s_D))$$

$$\text{avec} \quad S_D - \mathbb{P} = \{s_D \mid s_D \in S_D, \quad \forall R_j \in \mathbb{P}, \quad s_D \notin R_i\}$$

Ainsi, tout état présentant un ratio supérieur à 1 connecte davantage d'états dans la région qu'il ne connecte d'états non encore sélectionnés. Le ratio s'intéresse uniquement aux transitions non encore sélectionnées pour favoriser la construction des régions le long des frontières établies par les régions précédemment construites. L'état ayant le ratio maximal optimise la différence entre les transitions qui seront sauvées et les transitions qui seront potentiellement coupées. Ce critère local n'est valable que pour l'état courant du partitionnement, ainsi, la partition finale n'offre pas de garantie d'optimalité. Ce constat est vrai notamment pour les premiers états sélectionnés d'une région. Leur sélection se fait à l'aveugle et plusieurs des états voisins comptabilisés comme extérieurs seront sélectionnés lors d'itérations suivantes.

4.2 Contrôle de la taille des régions

Une partition idéale est aussi définie par le nombre d'objectifs contenus dans chaque région ainsi que par le nombre total de régions. Cependant, cette contrainte n'est pas dure. C'est-à-dire que malgré un nombre idéal de n^* objectifs par région, la minimisation des coupes sur les transitions peut amener à des partitions hétérogènes.Dans le cadre du partitionnement glouton, cela consiste à définir la condition permettant de continuer la construction d'une région, ou de passer à la suivante. En posant comme stratégie que, "tant qu'il existe un état dont le ratio est supérieur à 1, alors continuer la région", il faut adapter le ratio pour intégrer la taille de la région comme paramètre. Dans le cadre d'une région encore trop petite, il est nécessaire de simuler artificiellement que, parmi toutes les transitions localement coupées, certaines ne le seront plus. C'est à dire que, parmi les transitions qui sont coupées à un instant donné de la construction, certaine de ces transitions connecte probablement des états qui seront inclus dans la région lors des itérations suivantes.

En moyenne, il y a $t_{o.total}/|S_D|$ transitions sortantes par état modélisant la dynamique du système avec $t_{o.total}$ la somme des probabilités de toutes les transitions sortantes. Il est possible de considérer initialement que $t_{o.total}/|S_D|$ transitions ne seront finalement pas coupées. Cela correspond à retrancher ce volume de transitions au poids des transitions localement extérieures $t_{out}(\mathbb{P},\ s_D)$ pour générer une somme prédictive des probabilités de transitions extérieures $t_{o.pre}$. Ensuite, le facteur d'ajustement $t_{o.total}/|S_D|$ va être minoré au fur et à mesure que des objectifs sont ajoutés à la région et en fonction du nombre idéal d'objectifs par région n^*.

$$t_{o.pre}(\mathbb{P},\ R_i,\ s_D) = t_{out}(\mathbb{P},\ s_D) - \frac{n^* - |G \cap \mathbb{P}|}{n^*} . \frac{t_{o.total}}{|S_D|}$$

$$\text{avec} \quad t_{o.total} = \sum_{(s,\ s',\ a) \in S^2 \times A,\ s \neq s'} t_D(s, a, s')$$

Inversement, lorsqu'une région possède plus d'objectifs que le nombre idéal, le poids des transitions localement extérieures $t_{o.pre}$ est artificiellement augmenté. Plus la région contient des objectifs, plus il sera difficile pour un état, d'intégrer la région.

$$ratio(\mathbb{P},\ R_i,\ s_D) = \frac{t_{in}(R_i,\ s_D)}{\max(\ t_{out}(\mathbb{P},\ s_D) - \frac{n^* - |G \cap R_i|}{n^*} . \frac{t_{o.total}}{|S_D|},\ \epsilon\ )}$$

$$\text{avec} \quad \epsilon \ll \min(\ t_D(s,\ a,\ s')\ )$$

Le dénominateur du ratio est alors borné de façon strictement supérieure à zéro. Cela permet de contrôler la situation où le facteur d'ajustement est supérieur ou égal à la somme des transitions localement extérieures. Le ratio local dépend uniquement du voisinage de chaque état et du nombre d'objectifs dans la région courante. Ce ratio peut être calculé avec un nombre insignifiant d'opérations par rapport au nombre total d'états $|S_D|$ dans un MDP non fortement connexe ($t_{o.total} \ll |S_D|^2$).

4.3 Algorithme glouton de partionnement

L'idée générale du partitionnement glouton repose donc sur l'optimisation d'un critère local. L'algorithme glouton (algorithme 1) consiste alors à parcourir à chaque itération l'ensemble des états pour sélectionner l'état localement optimal.

Algorithme 1 : Partitionnement glouton

Données : un MDP $\langle S_D, A, t_D, r_D \rangle$, $s_{d0} \in S_D$, $G \subset S_D$, $n^* \in \mathbb{R}^+$
Résultat : P, $\bigcup_{R_i \in \mathbb{P}} R_i = S_D$ et $\forall R_i, R_j \in P$, $(R_i \cap R_j) = \emptyset$

initialiser $\mathbb{P} : \mathbb{P} \leftarrow \emptyset$;
tant que $\exists s_D \in S_D$, $\forall R_j \in P$, $s_D \notin R_j$ **faire**
    initialiser une nouvelle région : $R_i \leftarrow \emptyset$;
    choisir $s'_D : s'_D = \underset{s_D \in S_D}{argmin}(\ distance(s_{d0},\ s_D)\ )$;
    répéter
        ajouter s'_D à R_i;
        choisir $s'_D : s'_D = \underset{s_D \in S_D}{argmax}(\ ratio(\mathbb{P},\ R_i,\ s_D)\ )$;
    jusqu'à $ratio(\mathbb{P},\ R_i,\ s_D) > 1$;
    ajouter R_i à $\mathbb{P}$;

L'algorithme 1 construit la partion sur $|S_D|$ itérations et chaque itération demande au pire $|S_D|$ calculs de ratio. Ainsi, la complexité s'évalue à $o(n^2)$. Pareillement, la distance entre un état et tous les autres états peut se calculer en $o(n^2)$. Cette distance peut être calculée par exemple, comme le nombre minimal de transitions nécessaires pour atteindre un état. Cette notion sert à choisir un état pour initialiser une nouvelle région qui soit voisine d'une région existante sans laisser d'états isolés.

Dans les faits, avec des structures de données adaptées, il est possible de réduire encore la complexité. L'idée consiste alors à ne parcourir, à chaque itération, que les états atteignables. De même, la fonction

de distance peut être construite au fur et à mesure que de nouveaux états sont visités. Il en résulte une forme itérative de partitionnement acceptant un espace potentiellement infini. A partir d'un état initial s_{d0}, l'algorithme construira un nombre de régions avec n^* objectifs attendus par région en fonction du nombre d'itérations autorisées.

5 Résolution hiérarchique approchée

La partition $\mathbb{P}$ du MDP $\langle S_D,\ A,\ t_D,\ r_D\rangle$ modélisant la dynamique d'un système permet de déduire directement un partitionnement du GO-MDP $\langle S,\ A,\ t,\ r\rangle$ construit sur l'ensemble des objectifs G. Cependant, même décomposé, l'ensemble des sous-MDPs recouvre la totalité des états du GO-MDP, soit au minimum $2^{|G|}$ états. La solution que nous avons retenue vise une planification rapide des actions à réaliser, en considérant que même l'énumération de tous les états est impossible. Cette planification se fait alors, au détriment de l'optimalité de la solution.

La rapidité du calcul de la politique que nous avons mis en place repose sur l'heuristique que : une région peut être soit traversée, soit explorée totalement. Ainsi, il est possible de découper l'ensemble des objectifs en paquets d'objectifs. Un paquet d'objectifs correspond aux objectifs contenus dans une région. Cette méthode permet, dans une résolution hiérarchique d'un GO-MDP décomposé, de réduire considérablement le nombre d'états abstraits construits pour le GO-MDP de niveau supérieur $\langle S_a,\ A_a,\ t_a,\ r_a\rangle$ (Figure 4) et le nombre de GO-MDPs locaux potentiels.

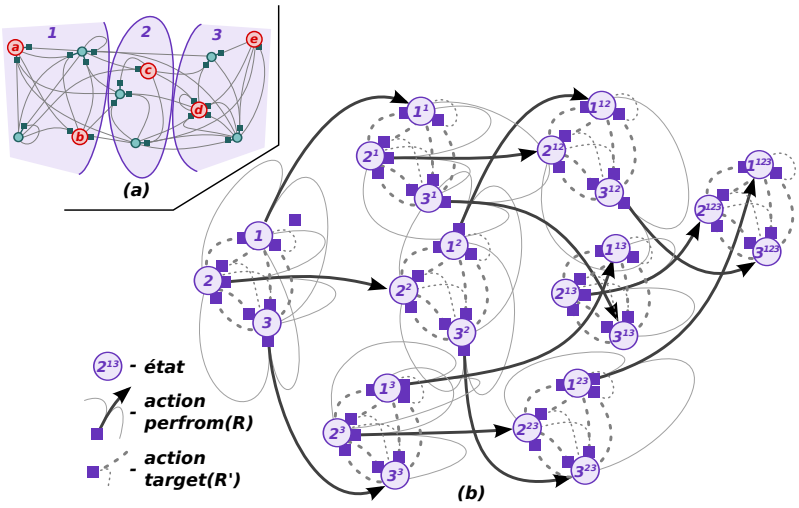

FIGURE 4 – Exemple de GO-MDP abstrait. Le GO-MDP abstrait (b) est construit à partir d'un MDP décomposé (a) et d'un ensemble de 5 objectifs $G = \{a,\ b,\ c,\ d,\ e\}$. Ici l'état (2^{13}) correspond à l'état $s_a = (2,\ \{1,\ 3\})$: le système est dans la région 2 et les objectifs des régions $1,\ 3$ sont atteints.

5.1 MDP de niveau supérieur

A partir de la partition $\mathbb{P}$ de taille k sur le MDP $\langle S_D,\ A,\ t_D,\ r_D\rangle$ et un ensemble d'objectifs G, il est possible de définir un ensemble de régions objectifs $RG \subseteq \mathbb{P}$ comme l'ensemble des régions qui possède au moins un objectif (toutes les régions avec un partitionnement glouton).

$$RG = \{R_i \mid \exists s \in R_i, \quad s \in G\}$$

Un état $s_a = (R_i,\ H) \in S_a$ du GO-MDP abstrait (Figure 4) modélise : R_i la région incluant l'état courant de la dynamique du système et H l'état courant d'exécution par rapport à l'ensemble des régions contenant des objectifs à atteindre RG. Dans le GO-MDP abstrait approché, depuis un état $s_a = (R_i,\ H) \in S_a$ les seules actions possibles sont : chercher à atteindre une région voisine ($target(R_j)$) ou réaliser entièrement l'ensemble des objectifs locaux ($perform(R_i)$).

$$S_a = \{(R_i,\ H) \mid R_i \in \mathbb{P}, \quad H \subseteq RG\}$$
$$A_a = \{target(R_j) \mid R_i \in \mathbb{P}, \quad R_j \neq R_i\} \cup \{perform(R_i) \mid R_i \in RG\}$$

Ainsi, la dynamique du système dans la région 1 avec les objectifs de la région 2 atteints correspond à l'état $(1,\ \{2\})$ du GO-MDP abstrait (soit l'état (1^2) dans la figure 4). Les 3 actions possibles sont : réaliser les objectifs de la région 1 ; chercher à atteindre la région 2 et chercher à atteindre la région 3. Les états atteignables sont : $(1,\ \{2\})$ pas de changement ; $(2,\ \{2\})$ ou $(3, \{2\})$ l'agent a changé de région ; $(1,\ \{1,\ 2\})$ les objectifs de la région 1 ont aussi été atteints (soit les états (1^2), (2^2), (3^2), (1^{12}) dans la figure 4).

5.2 Approximation des transitions et récompenses abstraite

La fonction de transition $t_a : S_A \times A_A \times S_A \to [0,\ 1]$ retourne les probabilités $t_a(s_a, a_a, s'_a)$ d'atteindre l'état abstrait s'_a à partir de s_a en exécutant l'action abstraite a_a. La fonction de récompenses abstraites $r_a : S_A \times A_A \to \mathbb{R}$ permet, quant à elle, d'évaluer l'intérêt contextuel des actions abstraites. Les transitions et les récompenses pour passer d'une région à une autre dans le MDP abstrait dépendent de l'état de départ, de la dynamique du système dans la région courante R_i et de la politique locale appliquée. Celle-ci dépend elle même des valeurs des états définissant une entrée dans la région suivante. Une évaluation optimale des transitions et des récompenses impliquerait de calculer toutes les politiques locales optimales pour toutes les valeurs potentielles des états périphériques. Ces valeurs dépendent de l'ensemble des objectifs qu'il reste à réaliser dans toutes les régions.

De nouveaux, dans un cadre de ressources de calcul contraintes, les transitions et les récompenses abstraites sont simplement approximées sans aucun calcul de politiques locales. L'idée de l'approximation repose sur le fait que, le GO-MDP abstrait ne sert plus à sélectionner les meilleures politiques locales mais à approximer les valeurs des états périphériques à un GO-MDP local de façon à orienter la politique locale. Pour une région R_i, l'approximation est réalisée à partir de moyennes calculées sur l'ensemble des transitions et des récompenses au niveau du sous-MDP système $\langle R_i,\ S_D,\ A,\ t_D,\ r_D \rangle$, de l'ensemble des objectifs inclus dans la région $G_i = G \cap R_i$ et de leurs gains associés.

Dans le cadre d'une transition d'une région R_i à une autre R_j par une action $target(R_j)$, la politique locale à R_i conduira l'agent sur un état frontière de R_i permettant de sortir en direction d'une autre région. En supposant une équi-probabilité d'atteindre un des états frontières de la région R_i, la transition abstraite peut être calculée de façon heuristique, vis à vis des actions maximisant les chances d'atteindre R_j.

$$t_A(\ (R_i, H),\ target(R_t),\ (R_j, H')\ ) = \begin{cases} \frac{1}{|Frontier(R_i)|} \sum\limits_{s_D \in Frontier(R_i),\ s'_D \in R_j} t_D(s_D,\ a^*(s_D, R_t),\ s'_D) & \text{si} \quad H \neq H' \\ 0 & \text{sinon} \end{cases}$$

$$\text{avec} \quad a^*(s_D, R_t) = \underset{a \in A_A}{argmax} \left(\sum_{s^t_D \in R_t} t_D(s_D,\ a,\ s^t_D) \right)$$
$$\text{et} \quad Frontier(R_i) = \{s_D \mid s_D \in R_i, \quad \exists t_D(s_D, a, s'_D) > 0, \quad s'_D \notin R_i\}$$

Inversement, lors de la réalisation d'une action $perform(R_i)$ à partir de la région R_i, la transition abstraite est évaluée pour les actions maximisant les chances de rester dans la région R_i et de finir les objectifs en cours. Nous considérons alors une équi-probabilité d'atteindre n'importe quel état frontière de la région R_i à un moment donné. Dans ces états frontières, nous supposons que la politique locale cherchera à atteindre des états dans la même région R_i.

$$t_A(\ (R_i, H),\ perform(R_t),\ (R_j, H')\ ) = \begin{cases} 0 & \text{si} \quad R_i \neq R_j \text{ et } H \neq H' \\ 0 & \text{si} \quad R_i = R_j \text{ et } H' \neq H \cup \{R_i\} \\ \frac{1}{|Frontier(R_i)|} \sum\limits_{s_D \in Frontier(R_i),\ s'_D \in R_j} t_D(s_D,\ a^*(s_D, R_i),\ s'_D) & \text{sinon} \end{cases}$$

De façon similaire aux transitions, il est possible d'évaluer approximativement la récompense obtenue dans une région R_i sur la base d'une moyenne sur les récompenses $r_m(R_i)$.

$$r_m(R_i) = \frac{1}{|R_i|} \sum_{s \in R_i} \left(\frac{\sum\limits_{a \in A_{ds}} r_D(s, a)}{|A_{ds}|} \right) \quad \text{avec} \quad A_{ds} = \left\{ a \ \middle| \ \begin{array}{c} \exists s' \in S_D, \quad s' \neq s, \\ t_D(s,\ a,\ s') > 0 \end{array} \right\}$$

A_{ds} représente ici l'ensemble des actions disponibles depuis l'état $s \in S_D$. Ainsi, la récompense approchée pour atteindre une région voisine peut être grossièrement évaluée avec un nombre théorique β_t d'actions à réaliser pour sortir d'une région. De la même façon, nous proposons de définir le coût approché pour réaliser les objectifs d'une région en fonction d'un nombre théorique β_p d'actions nécessaires pour passer d'un objectif à un autre. La récompense d'atteindre les objectifs d'une région se calcule alors comme la somme des gains associés aux objectifs moins les coûts évalués.

$$r(\ (R_i, H),\ target(R_j)\ ) = \beta_t.r_m(R_i)$$

$$r(\ (R_i, H),\ perform(R_i)\ ) = \sum_{s \in G_i - H} gain(s) - \beta_p.r_m(R_i)$$

5.3 Résolution locale région par région

Sur la base des valeurs approximatives calculées via le GO-MDP abstrait, la solution que nous proposons afin de planifier l'exploration consiste à ne calculer que la politique locale pour la région courante R_i. Ainsi, en cas d'actualisation de la connaissance, l'actualisation de la politique, avant un contrôle effectif, nécessite uniquement : un nouveau partitionnement, la résolution du GO-MDP abstrait approché et la résolution du GO-MDP local à la région R_i. La solution proposée approche une politique sur la base de la résolution séquentielle de 2 GO-MDPs (un abstrait et un local), construits avec k et $|G_i|$ objectifs (avec $k \simeq |G|/n^*$ régions et $|G_i| \simeq n^*$ objectifs par région).

La résolution d'un GO-MDP local est conditionnée par la valeur des états atteignables et notamment ceux dans les régions voisines en fonction de l'étape de la mission (objectifs atteints et non-atteints) (Figure 5). Concrètement, la réalisation des objectifs d'une région va être orientée de façon à terminer dans un état facilitant le passage à la région suivante.

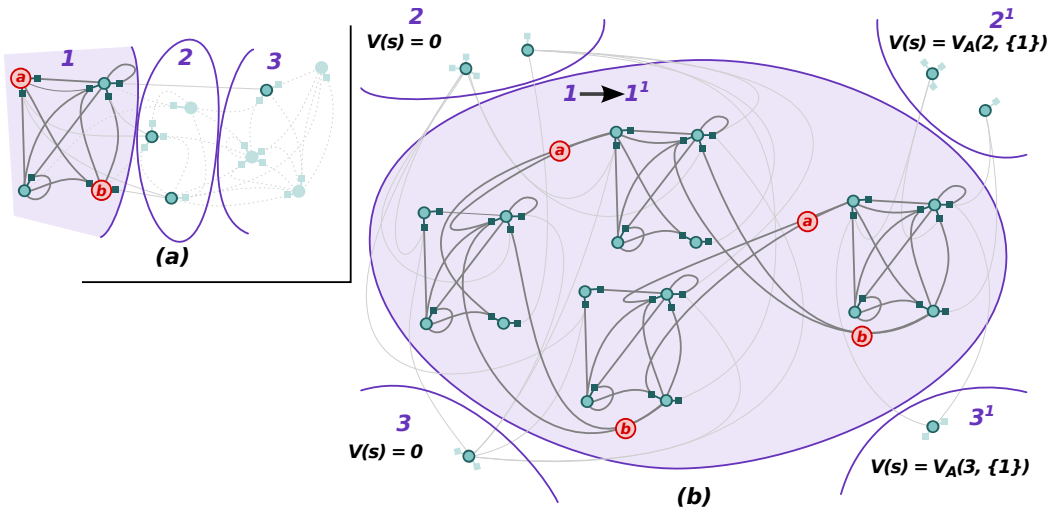

FIGURE 5 – Exemple de GO-MDP local (b) construit à partir de la région 1 et de l'ensemble des 2 objectifs $\{a,\ b\}$ (a). Dans la mesure où tous les objectifs locaux sont réalisés, il sera possible de changer de région en considérant les objectifs de la région 1 comme réalisés.

5.4 Initialiser une résolution locale

La résolution locale utilise alors une évaluation initiale V_{init} des états extérieurs et atteignable à partir de la région courante R_i (des valeurs qui ne seront pas modifiées). Dans l'approche proposée ici, l'évaluation initiale découle des valeurs abstraites $V_A^{\pi_A^*}$ attachées à la politique π_A^* calculée pour le GO-MDP abstrait approché.

Les valeurs des états dans les régions voisines sont initialisées à 0 (neutre) tant que tous les objectifs de la région courante ne sont pas atteints. Ainsi, les actions locales de l'agent seront orientées vers la réalisation

de tous les objectifs (en sous-entendant des gains positifs forts pour la réalisation des objectifs). Ensuite, la valeur d'un état voisin est associée à la valeur approximative de sa région pour l'étape de réalisation courante H (ensemble des régions dont les objectifs sont atteints) et en considérant que les objectifs de la région actuelle R_i seront aussi validés.

$$\forall s \in Perifery(R_i) \quad \begin{array}{ll} V_{init}(s) = V_A^{\pi^*_A}(R_j,\ H \cup \{R_i\}) & \text{si tous les objectifs de } R_i \text{ sont atteints} \\ V_{init}(s) = 0 & \text{sinon} \end{array}$$

Ainsi, l'action abstraite, définissant une préférence sur l'ordre des régions à parcourir, peut être localement remise en cause. Par exemple, une résolution locale à la région 1 orientée par les deux régions 2 et 3 peut être initialisée avec une préférence sur la région 3 (Figure 5). La valeur abstraite pour $(3,\ \{1\})$ sera supérieure à la valeur de $(2,\ \{1\})$. La politique locale peut, elle, chercher à exécuter l'objectif a puis b et viser ensuite la région 2 (si le coût d'aller en 3 est supérieur à la différence des valeurs abstraites entre les régions 2 et 3.)

Malgré des valeurs initiales désavantageuses, le système peut changer de région involontairement avant que tous les objectifs locaux ne soient validés. En effet, l'agent peut être amené à choisir des actions avec une probabilité, même faible, de sortir de la région. Dans une situation de sortie prématurée, il sera alors utile de ré-initialiser la politique en actualisant le partitionnement, le GO-MDP abstrait et le GO-MDP local courant en tenant compte des objectifs atteints ou non.

6 Validation statistique

L'approche hiérarchique proposée permet de résoudre de façon approchée un GO-MDP composé d'un nombre conséquent d'objectifs. Le GO-MDP permet, dans le cadre de nos travaux, d'exprimer la problématique de planification d'un robot explorateur. Dans le scénario final visé, les objectifs représentent, à chaque instant, les frontières à repousser. La carte et les objectifs évoluent au fur et à mesure que l'exploration est réalisée.

La validation expérimentale permet de tester les heuristiques utilisées. Une évaluation qualitative cherche à évaluer les politiques construites notamment vis à vis de politiques optimales. D'autres séries d'expérimentations sont ensuite présentées pour évaluer la rapidité de l'approche pour construire la politique et, par conséquent, la possible utilisation de l'approche en cours de mission d'exploration en robotique mobile. En dehors de l'évaluation rapide du partitionnement, les expériences sont réalisées avec une carte composée de zones hétérogènes (différentes densités d'obstacles), tel que présenté sur la Figure 6.

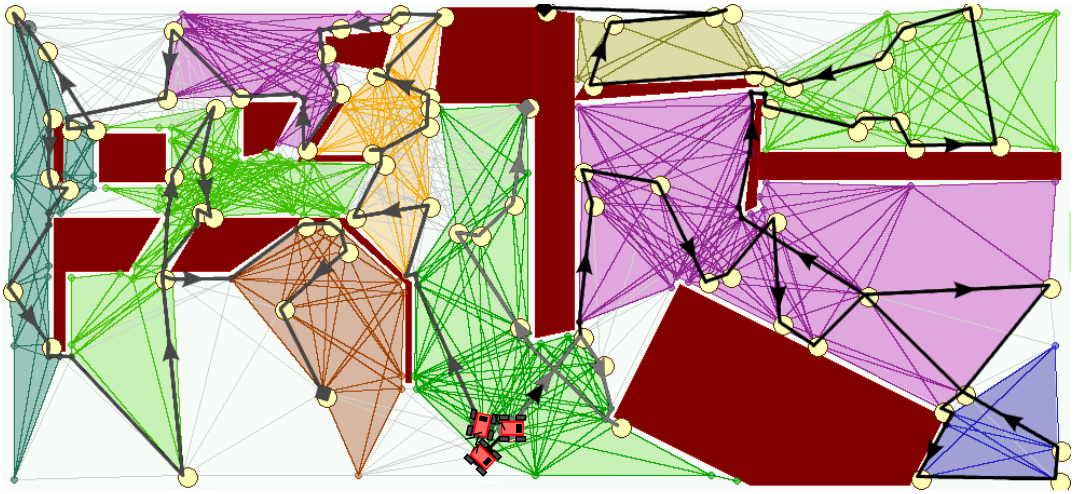

FIGURE 6 – Résolution hiérarchique, exemple des parcours probables empruntés par les robots après allocation des régions et calcul individuel de leurs politiques locales.

6.1 Évaluation rapide du partitionnement

Dans un objectif d'illustrer nos propos, une évaluation visuelle effectuée sur une première série d'expérimentations permet de tirer quelques conclusions sur le partitionnement de trois cartes des déplacements (Figure 7). Ainsi, dans des environnements plus ou moins structurés, il a été remarqué que l'algorithme de partitionnement glouton construit une partition proche d'une partition qui serait attendue par un opérateur humain. Dans des environnements structurés (Figure 7(c)) avec un nombre cohérent d'objectifs par région, le partitionnement glouton construit approximativement une région par pièce. C'est un bon résultat compte tenu du fait qu'il n'y a pas de calculs globaux sur les coupes dans la succession de choix réalisés par l'algorithme.

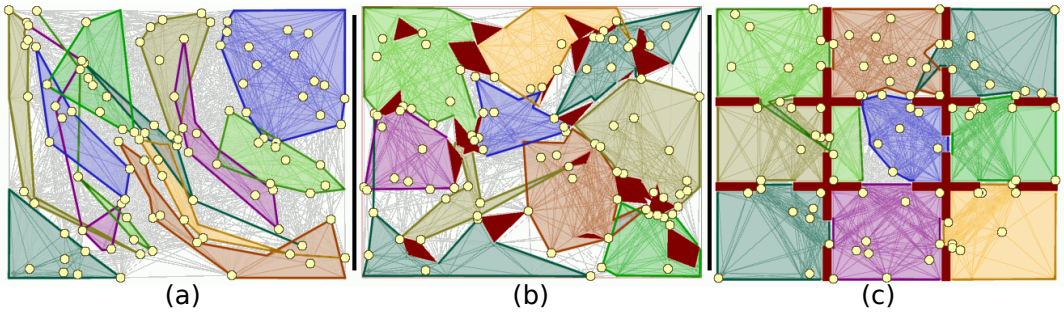

FIGURE 7 – Exemples de partitionnements gloutons construits pour des environnements présentant des structures différentes : (a) grand espace libre, (b) obstacles aléatoires et (c) pièces de tailles homogènes. L'algorithme glouton est paramétré avec, comme position initiale, le point supérieur gauche.

Inversement, dans un environnement sans obstacle (MDP fortement connexe), le partitionnement rencontre des difficultés pour construire des régions significatives (Figure 7(a)). Les régions sont "allongées" et se chevauchent les unes par rapport aux autres. Cela est notamment dû à la non-prise en compte des récompenses et coûts sur les transitions. La notion de distance est seulement basée sur les fortes ou faibles probabilités d'atteindre un état. Dans les exemples de la Figure 7, les transitions liées aux actions de déplacements sont définies de façon homogène quelle que soit la distance entre les positions. Ce phénomène (régions incohérentes), se réduit dans des environnements encombrés même avec des obstacles aléatoires (Figure 7(b)).

6.2 Évaluation qualitative

Cette série d'expériences compare la valeur des politiques après une allocation des régions entre une petite flotte d'agents. En effet, les scénarios visés consistent en l'application de ces méthodes dans un cadre multi-robot. Aussi, l'évaluation qualitative proposée ici consiste en une comparaison des valeurs des politiques optimales après une allocation consécutive au partitionnement et des politiques optimales associées à la meilleure et la pire des allocations possibles.

La valeur d'une allocation équivaut à la somme des gains des objectifs moins la somme des déplacements probables de chaque robot en fonction de son attribution. Cette valeur est calculée en sommant les utilités espérées données par l'équation de Bellman associée aux politiques optimales globales référentes. Les politiques optimales globales sont calculées en considérant tous les objectifs attribués dans un GO-MDP unique (sans hiérarchisation : GO-MDPs abstrait et locaux). Ces politiques ne servent que pour l'évaluation des allocations dans cette série d'expériences.

$$value(G_0, \ldots, G_n) = \sum_{Ag \in [O,\, n]} V^{\pi^*}(s_{Ag}, G_{Ag}) \quad \text{avec } s_{Ag} \text{ l'état initial du robot } Ag$$

L'utilité espérée $V^{\pi^*}(s)$ dépend des coûts et récompenses qui seront statistiquement perçus par le robot. Dans le cadre des déplacements, les coûts sont ici calculés de façon proportionnelle à la distance entre les 2 positions ciblées par l'action. Le coût maximal est défini ici à 51.2 pour un déplacement virtuel sur la diagonale de l'environnement. Les gains pour visiter une position objectif sont posés à 1000 pour garantir

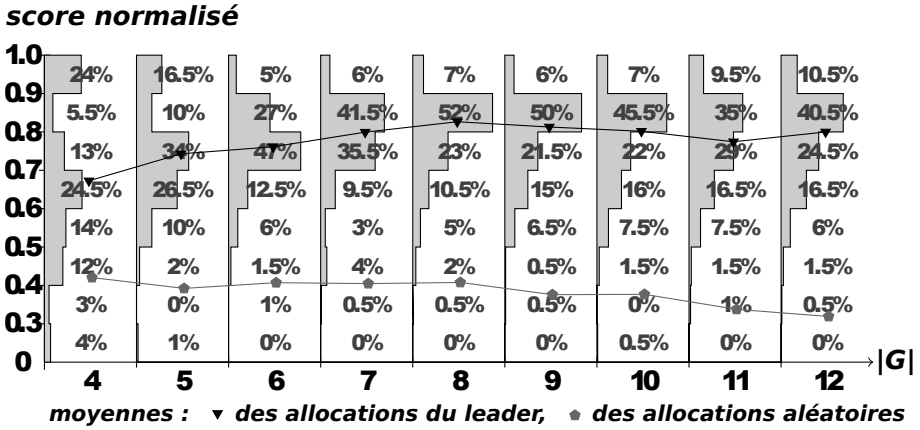

FIGURE 8 – Évaluation des allocations résultant d'un partionnement glouton. La population des scores obtenus par une allocation basée sur un GO-MDP abstrait approché est présentée en fonction du nombre d'objectifs considérés $|G|$. Par exemple pour $|G| = 8$: 7% des allocations construites ont eu un score entre 0.9 et 1.0 (avec 1.0 : allocation optimale) ; 52% entre 0.8 et 0.9, etc.

l'intérêt de toujours chercher à atteindre un objectif. Enfin, le partitionnement glouton est paramétré pour un idéal de 3 objectifs par région. Les expériences sont classifiées en fonction du nombre total d'objectifs $|G|$ considérés. Ce nombre est limité à 12 à cause de la complexité de calculer l'allocation optimale via une énumération exhaustive de toutes les allocations possibles.

Pour chaque expérience, un score normalisé est calculé pour l'allocation approchée et une allocation aléatoire. Le score correspond à la valeur résultant de la somme des politiques individuelles proportionnellement aux valeurs de la meilleure et de la pire allocation possible. Par exemple, une des expériences, dans la classe $|G| = 11$, qui a permis d'établir les moyennes (Figure 8), a donné les valeurs : 10913.8 (meilleur) ; 10822.9 (pire) ; 10872.9 (aléatoire) ; et 10884.5 (par décomposition) soit un score normalisé de 0.550 pour l'aléatoire et de 0.678 pour l'approche développée dans ce papier.

Cette série d'expériences est composée de 200 générations aléatoires d'objectifs pour chaque classe considérée (nombre paramétré d'objectifs). Ce volume permet une évaluation statistique de l'allocation réalisée sur la base d'un GO-MDP abstrait approximatif (Figure 8).

Les scores obtenus (Figure 8) montrent la difficulté de trouver l'allocation optimale sur la base du GO-MDP abstrait. La moyenne des scores est proche de 77% pour des problèmes comprenant entre 6 et 12 objectifs ce qui induit de 2 à 4 régions. Cela reste une moyenne acceptable en considérant que le calcul de cette allocation est quasiment instantané. Cette moyenne est affectée négativement par un petit nombre d'expériences présentant de mauvais scores (< 0.5). Ces mauvais scores découlent d'un partitionnement inapproprié avec l'allocation des régions entre les robots. Le nombre de ces expériences diminue avec l'augmentation du nombre d'objectifs (et le nombre de régions construites) pour se stabiliser à un score autour des 80% (entre la pire et la meilleure allocation possible). Ces dernières expériences correspondent à un ratio avec autant et un peu plus de régions que d'objectifs par région.

6.3 Temps de calcul de la politique

L'efficacité de la solution s'évalue, par la capacité du partitionnement à contrôler la taille (en terme d'objectifs) et le nombre des régions construites. Ces paramètres vont permettre en retour, de contrôler la taille du GO-MDP abstrait et des GO-MDPs locaux.

Notre approche a été conçue afin pouvoir considérer, en cours de mission, un nombre important d'objectifs. Cette série d'expériences évalue le temps nécessaire au calcul de la politique abstraite (partitionnement et résolution du GO-MDP abstrait). Les expériences sont réalisées sur la base de : jusqu'à 120 objectifs et un nombre idéal de 10 (tailles et nombre de régions similaires) objectifs par région ($|G| \leq 120$ et $n_i^* = 10$). La série d'expérimentations est générée par positionnement aléatoire des objectifs : 500 générations aléatoires pour les différents nombres d'objectifs $|G|$ considérés (Tableau 1).

Le tableau 1 et la figure 9 présentent les résultats expérimentaux permettant de valider le bon contrôle

$\|G\|$	nombre moyen de régions (k)	bornes moyennes sur la taille des régions	temps (s)
5	1.13	$4.38 - 4.97$	0.059
20	3.41	$3.43 - 8.13$	0.047
40	5.6	$4.12 - 10.14$	0.053
60	7.71	$4.29 - 11.44$	0.116
80	9.43	$4.73 - 12.74$	0.339
100	10.63	$4.36 - 14.19$	0.897
120	12.42	$3.74 - 14.81$	4.95

TABLE 1 – Moyennes mesurées caractérisant la résolution approchée.

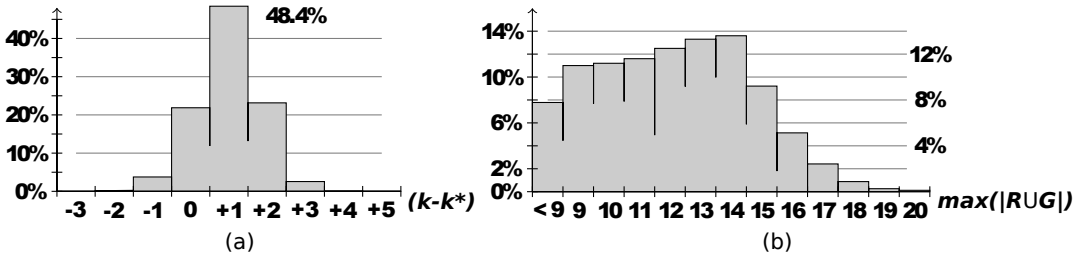

FIGURE 9 – Tailles des partitions gloutonnes créées. Les diagrammes (a) et (b) classifient les expériences en fonction : (a) de la différence entre le nombre construit k et le nombre idéal k^* de régions et (b) le nombre d'objectifs contenus dans la région la plus importante. Par exemple, (a) 48.4% des partitionnements ont construit une région de trop par rapport au nombre idéal k^* et (b) 13.5% des régions possèdent 14 objectifs.

du nombre de régions, du nombre d'objectifs par région et des temps de calcul associés. On peut noter un nombre déséquilibré d'objectifs par régions. La taille des GO-MDP et les temps de calcul augmentent exponentiellement avec le nombre de régions k construites (GO-MDP abstrait) et avec le nombre d'objectifs dans les régions (GO-MDP locaux). C'est un résultat tout à fait cohérent sachant que la taille des GO-MDPs augmente exponentiellement avec le nombre d'objectifs.

Les expériences comportant un faible nombre d'objectifs par rapport à la densité des obstacles (le nombre et leurs formes) conduisent à des partitions avec plus de régions qu'attendu ($k > k^*$) et certaines régions, voire toutes, avec un faible nombre d'objectifs ($|R_i \cup G| < n^*$) (Figure 9). Ce résultat montre la souplesse du partitionnement vis à vis du nombre idéal d'objectifs par région et sa capacité à profiter de la configuration des obstacles pour passer d'une région à une autre.

7 Conclusion

Nous avons traité ici du problème de planification par résolution d'un MDP large avec une contrainte forte sur la rapidité des calculs mis en œuvre. Les approches par décomposition d'un processus décisionnel de Markov (DMDP) permettent d'accélérer la résolution d'un MDP large. Ces approches nécessitent alors un partitionnement de l'ensemble des états couplés au calcul de plusieurs politiques par région. Une résolution optimale d'un MDP décomposé orienté par des objectifs (GO-DMDP) est difficilement envisageable dans la mesure où même une simple énumération des états est fastidieuse.

L'approche présentée ici propose une solution *ad hoc* à un problème d'exploration en robotique mobile, permettant de focaliser les calculs sur seulement quelques politiques locales, au fur est à mesure que les régions sont atteintes. Le partitionnement glouton d'un MDP définissant la dynamique du système (sans intégrer les objectifs) permet de construire une hiérarchie de GO-MDPs. Un GO-MDP abstrait approché, construit par rapport à l'ensemble des régions permet d'évaluer grossièrement les valeurs de gains espérés par région. Ces valeurs orientent ensuite la politique locale calculée pour la région dans laquelle le robot est positionné.

Les séries d'expériences que nous avons réalisées ont montré que l'approche permet de calculer des

politiques acceptables dans un temps cohérent. La vitesse de convergence vers la solution est garantie par un partitionnement glouton offrant un bon contrôle statistique sur la taille de la partition et sur le nombre d'objectifs contenus dans chaque région.

Dans la continuité directe de ce travail, nous souhaitons étudier, de façon plus approfondie, les conséquences du partitionnement sur la politique individuelle des robots. L'idée repose sur une évaluation comparative des techniques de partitionnement par rapport aux valeurs des politiques construites. Nous ne pensons, à priori, pas que le coût supplémentaire en ressources de calcul, d'une optimisation de la partition soit justifié par de réelles améliorations de la politique résultante. Cependant, cette supposition reste à vérifier.

Références

BELLMAN R. (1957). A markovian decision process. *Journal of Mathematics and Mechanics*, **6**, 679–684.

BERHAULT M., HUANG H., KESKINOCAK P., KOENIG S., ELMAGHRABY W., GRIFFIN P. & KLEYWEGT A. (2003). Robot exploration with combinatorial auctions. In *International Conference on Intelligent Robots and Systems*, volume 2, p. 1957–1962.

BICHOT C.-E. & SIARRY P. (2011). *Graph Partitioning*. Wiley-ISTE.

BOUTILIER C., DEAN T. & HANKS S. (1999). Decision-theoretic planning : Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, **11**, 1–94.

BURGARD W., MOORS M., STACHNISS C. & SCHNEIDER F. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, **21**.

DEAN T., HONG LIN S. & HONG LIN S. (1995). Decomposition techniques for planning in stochastic domains. In *14th International Joint Conference on Artificial Intelligence*.

FOKA A. F. & TRAHANIAS P. E. (2007). Real-time hierarchical pomdps for autonomous robot navigation. *Robotics and Autonomous Systems*, **55**(7), 561–571.

GAREY M. R., JOHNSON D. S. & STOCKMEYER L. (1974). Some simplified np-complete problems. In *6h Symposium on Theory of Computing*, p. 47–63, New York, NY, USA : ACM.

KUIPERS B. & BYUN Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, **8**, 47–63.

LITTMAN M., DEAN T. & KAELBLING L. (1995). On the complexity of solving markov decision problems. In *11th Conference on Uncertainty in Artificial Intelligence*, p. 394–402.

LOZENGUEZ G. (2012). *Stratégie coopérative pour la mise en œuvre d'une flotte de robots mobiles dans un milieu ouvert et encombré*. PhD thesis, UniversitÃ© de Caen Base-Normandie.

LOZENGUEZ G., ADOUANE L., BEYNIER A., MARTINET P. & MOUADDIB A.-I. (2011). Map partitioning to approximate an exploration strategy in mobile robotics. In *Advances on Practical Applications of Agents and Multiagent Systems*, volume 88, p. 63–72.

LOZENGUEZ G., ADOUANE L., BEYNIER A., MARTINET P. & MOUADDIB A.-I. (2012). Interleaving planning and control of mobiles robots in urban environments using road-map. In *International Conference on Intelligent Autonomous Systems*.

MATIGNON L., LAURENT J. & MOUADDIB A. (2012). Distributed value functions for multi-robot exploration. In *International Conference on Robotics and Automation*.

PAPADIMITRIOU C. & TSITSIKLIS J. (1987). The complexity of markov decision process. *Mathematics of Operations Research*, **12**, 441–450.

PARR R. (1998). Flexible decomposition algorithms for weakly coupled markov decision problems. In *14th Conference on Uncertainty in Artificial Intelligence*, p. 422–430.

PUTERMAN M. L. (1994). *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.

SABBADIN R. (2002). Graph partitioning techniques for markov decision processes decomposition. In *15th Eureopean Conference on Artificial Intelligence*, p. 670–674.

SIMMONS R., APFELBAUM D., BURGARD W., FOX D., MOORS M., THRUN S. & YOUNES H. (2000). Coordination for multi-robot exploration and mapping. In *National Conference on Artificial Intelligence*.

SUTTON P. & BARTO A. (1998). *Reinforcement learning : An introduction*. MIT Press, Cambridge, MA.

TEICHTEIL-KÖNIGSBUCH F. & FABIANI P. (2006). Autonomous search and rescue rotorcraft mission stochastic planning with generic dbns. In *IFIP AI*, p. 483–492.

THRUN S. (1998). Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, **99**(1), 21–71.