Itération de la Valeur basée sur des Avantages pour des MDPs avec récompenses Inconnues – *Résumé étendu*

Pegah Alizadeh, Yann Chevaleyre, François Lévy*

*LIPN, UMR CNRS 7030
Institut Galilée - Université Paris 13
99, avenue Jean-Baptiste Clément
93430 Villetaneuse, France
prénom.nom@lipn.univ-paris13.fr

Résumé. Cet papier considère un problème de *Processus de Décision de Markov* (MDP) avec des récompenses inconnues. Pour résoudre ce problème et trouver la politique optimale, il transforme le MDP en une *forme vectorielle (VVMDP)*. Il introduit un nouvel algorithme ABVI basé sur l'algorithme d'itération de la valeur sur les VVMDPs, et qui intégre à cette méthode des requêtes à l'utilisateur pour réduire l'incertitude en cas de besoin. ABVI classifie les vecteurs *avantage*s, à savoir les $\beta(s).(\bar{Q}(s,a) - \bar{V}(s))$, pour réduire le nombre de requêtes. L'algorithme est décrit et justifié, et des résultats expérimenraux montrent son intérêt.

1 Introduction

Les *Processus de Décision de Markov(MDP)* sont des modèles pour résoudre les problèmes de décision séquentielle où un agent interagit avec l'environnement grâce à des actions qui reçoivent des récompenses numériques. Dans la plupart des cas réels, il est difficile de déterminer toutes les récompenses numériquement, parce que les données sont complexes et parce qu'elles peuvent comporter des préférences qualitatives. Dans ce cas, la *politique optimale* doit être calculée selon les préférences de l'agent, en utilisant un MDP avec des récompenses imprécises.

De nombreux chercheurs ont étudié le calcul direct des politiques optimales robustes pour les *MDP avec des récompenses inconnues (IRMDPs)* (Delage et Mannor, 2007; Macmahan et al., 2003; Regan et Boutilier, 2009; Xu et Mannor, 2009). (Weng, 2011; Weng et Zanuttini, 2013) ont reformulé le problème à l'aide de *MDP avec des valeurs vectorielles (VVMDP)*. Les récompenses inconnues sont exprimées comme un produit scalaire entre un vecteur des types de récompense et un vecteur de poids de chaque type noté $\bar{\lambda}$. La solution optimale est calculée par Itération de la Valeur Interactive : des requêtes à l'utilisateur suppléent à l'ignorance de $\bar{\lambda}$.

Nous proposons une méthode d'itération de la valeur modifiée pour minimiser les interactions avec l'utilisateur. La principale contribution de cet article utilise les vecteurs *avantages*, calculés à partir des vecteurs $\bar{Q}(s,a) - \bar{V}(s)$, dans chaque itération. Pour réduire les interactions, on classifie les avantages selon leur direction et on produit de nouveaux vecteurs

équivalents à la somme des clusters. Cette idée de base nous permet de comparer des vecteurs moins nombreux et plus grands. Cela réduit le nombre de comparaisons et génère des requêtes plus robustes. Notre contribution finale rapporte quelques expériences qui indiquent comment la méthode présentée est effective après avoir posé moins de questions à l'agent. Toutefois, le nombre total de requêtes proposées dans notre approche est supérieur à celui de la méthode de Weng et Zanuttini (2013) ; la majorité des requêtes sont proposées après l'obtention d'une solution raisonnablement précise.

2 Point de départ

Un MDP est défini par un n-uplet $< S, A, p, r, \gamma, \beta >$ où S est l'ensemble des états, A est l'ensemble des actions, $p(s'|s,a)$ la probabilité de passer dans l'état s' sachant que l'on est dans l'état s et qu'on a choisi l'action a. $r : S \times A \longrightarrow \mathbb{R}$ est la fonction de récompense, $\gamma \in [0,1[$ est le facteur d'ammortissement et β la distribution sur les états initiaux. On se concentre sur les MDP d'horizon infini avec un nombre fini d'états et d'actions.

Une politique stationnaire est une fonction $\pi : S \longrightarrow A$. La *fonction de valeur* $V^\pi(s)$ est l'espérance de la somme actualisée des récompenses en applicant π. Sa *valeur espérée* est $\mathbb{E}_{s\sim\beta}[V^\pi(s)] = \beta \cdot V^\pi$. Pour calculer la politique optimale et sa valeur espérée, on utilise les approximations successives produites par la boucle d'itération ci-après :

$$\begin{aligned} Q^\pi(s,a) &= r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^\pi(s') \\ \pi(s) &= \text{argmax}_a \; Q^\pi(s,a) \\ V^\pi(s) &= Q^\pi(s, \pi(s)) \end{aligned} \tag{1}$$

Dans ce calcul, chaque $Q^\pi(s,a)$ a une amélioration par rapport à $V^\pi(s)$. Cette différence pondérée par la distribution initiale sur l'état est appelée *avantage* (Baird, 1993)

$$A(s,a) = \beta(s)\{Q^\pi(s,a) - V^\pi(s)\}$$

Le choix de politique dans l'equation 1 peut alors être exprimé par $\pi(s) = \text{argmax}_a \; A(s,a)$

Lorsque des cas réels sont modélisés par des MDPs, spécifier la fonction de récompense est généralement un problème difficile. Pour remédier à cette difficulté, Weng (2011) modélise d'abord par un IRMDP. Nous utilisons la même structure et la même transformation en VVMDP que Weng et Zanuttini (2013) avec une différence : les récompenses inconnues ne sont pas ordonnées et on peut les combiner avec d'autres récompenses connues.

Un IRMDP est défini à l'aide d'un ensemble de variables $E = \{\lambda_1, .., \lambda_{d-1}\}$. E est l'ensemble des valeurs possibles pour les récompenses inconnues pour le MDP donné. Initialement, on sait seulement que $\forall i,\ 0 \leq \lambda_i \leq 1$ et que $\forall s, a,\ r(s,a) \in E \cup \mathbb{R}$.

Pour la transformation en VVMDP, nous définissons le vecteur des poids $\bar{\lambda} = (\lambda_1, .., \lambda_{d-1}, 1)$ et la fonction vectorielle *sélecteur de récompense* $\bar{r} : S \times A \longrightarrow \mathbb{R}^d$ comme suit [1] :

$$\bar{r}(s,a) = \begin{array}{l} (1)_j \text{ si } r(s,a) \text{ a la valeur inconnue } \lambda_j \ (j < d) \\ x.(1)_d \text{ si } r(s,a) \text{ est connu pour être exactement } x. \end{array}$$

1. $(1)_j$ note des vecteurs de dimensions d de form $(0, .., 0, 1, 0, .., 0)$ ayant un seu1 in dans la j-ième élément

La récompense scalaire esst obtenue par un produit scalaire :

$$r(s,a) = \bar{\lambda} \cdot \bar{r}(s,a) \tag{2}$$

Les techniques de base des MDP peuvent être appliquées composante par composante au VVMDP. Afin de trouver le vecteur maximal dans l'équation 1, l'étape d'itération a besoin de comparer les vecteurs de valeur entre eux. Ce point est abordé dans la Section 3.

Maintenant, supposons que $\bar{V}^{\pi}$ est la fonction de valeur vectorielle calculée par composantes. Prennant en compte la distribution initiale, la valeur vectorielle espérée est

$$\mathbb{E}_{s\sim\beta}[\bar{V}^{\pi}(s)] = \sum_{s\in S} \beta(s)\bar{V}^{\pi}(s)$$

En choisissant une valeur numérique pour le vecteur de poids $\bar{\lambda}$, on peut calculer la valeur espérée scalaire $v^{\pi} = \bar{\lambda}\cdot\mathbb{E}_{s\sim\beta}[\bar{V}^{\pi}](s)$. On obtient ainsi une comparaison entre deux politiques (à $\bar{\lambda}$ fixé). Quand $\bar{\lambda}$ varie, restreindre ses valeurs émonde certaines valeurs vectorielles parce qu'elles ne sont plus maximales pour aucun $\bar{\lambda}$. Inversement, sachant laquelle de deux politiques est préférée émonde les valeurs de $\bar{\lambda}$ qui entrent en conflit avec cette connaissance.

Dans la suite, l'ensemble de tous les vecteurs de poids possibles $\bar{\lambda}$ pour le VVMDP est noté Λ, l'ensemble de toutes les politiques possibles Π, et l'ensemble de leurs valeurs vectorielles $\bar{\mathcal{V}}$ ($\bar{\mathcal{V}} = \{\bar{V}^{\pi} : \pi \in \Pi\}$). Sans perte de généralité, on suppose que le polytope Λ est un cube unité de dimension d.

3 Fonctions de valeur vectorielles

L'idée principale est de trouver le vecteur optimal $\bar{V}^{*}$ dans $\bar{\mathcal{V}}$ en utilisant la méthode d'itération de la valeur sur le VVMDP, et de rechercher $\bar{\lambda}^{*} \in \Lambda$ en utilisant l'explicitation des préférences pour réduire le polytope Λ. L'algorithme d'itération de la valeur sur un VVMDP (Weng et Zanuttini (2013)) a besoin de comparer des vecteurs. On utilise trois méthodes de comparaison en cascade. Elles sont testées dans l'ordre donné jusqu'à ce que l'une donne une réponse.

1. *la Dominance de Pareto* est la dominance scalaire dans chacune des coordonnées.
2. La *DominanceK* définit $\bar{V}^{\pi_i} \succcurlyeq_K \bar{V}^{\pi_j}$ par $\min_{\bar{\lambda}\in\Lambda} \bar{\lambda}\cdot(\bar{V}^{\pi_i} - \bar{V}^{\pi_j}) > 0$, ce qui peut être décidé à l'aide d'un programme linéaire (Weng et Zanuttini, 2013).
3. soumettre à l'utilisateur une requète qui permet de décider si $\bar{\lambda}\cdot\bar{V}^{\pi_i} \succ \bar{\lambda}\cdot\bar{V}^{\pi_j}$. Cette solution est l'option la plus coûteuse et la moins désirée. Notre but est de trouver la solution optimale par rapport au nombre d'interruptions pour interagir avec l'utilisateur.

Soit π'' une politique qui diffère de π en remplaçant les actions des états $\hat{s}_1 \ldots \hat{s}_k$ par $\hat{a}_1 \ldots \hat{a}_k$, En supposant que chaque $\hat{a}_i$ est choisi de sorte que $\bar{Q}^{\pi}(\hat{s}_i, \hat{a}_i) \geq \bar{V}^{\pi}(\hat{s}_i)$, on a :

$$\mathbb{E}_{s\sim\beta}[\bar{V}^{\pi''}] - \mathbb{E}_{s\sim\beta}[\bar{V}^{\pi}] = \sum_{i=1}^{k} \bar{A}_{\hat{s}_i,\pi''(\hat{s}_i)}$$

Autrement dit, plusieurs avantages vectoriels sont additifs. Pour faire moins de comparaisons entre les vecteurs, nous essayons de choisir des avantages qui améliorent la réponse avec

Algorithm 1 Value Iteration Algorithm with Advantages

Input : MDP($S, A, p, \bar{r}, \gamma, \beta$), ϵ
Output : near-optimal $\bar{V}^*(s)$ for each s

$t \longleftarrow 0$
$\pi \longleftarrow$ choose random policy
$\forall s\ \bar{V}_0(s) \longleftarrow (0, .., 0)$: zero vector of d dimension
$\mathcal{K} \longleftarrow$ set of constraints on the unit cube
repeat
  $\mathcal{A} \longleftarrow \emptyset$
  for each s, a **do**
    $\bar{Q}_t(s,a) \longleftarrow \bar{r}(s,a) + \gamma \sum_{s'} p(s'|s,a)\bar{V}_t(s')$
    $\bar{A}_{s,a} \longleftarrow \beta(s)\{\bar{Q}_t(s,a) - \bar{V}_t(s)\}$
    $\mathcal{A} \longleftarrow$ Add $\bar{A}_{s,a}$ to $\mathcal{A}$
  $\mathcal{C} \longleftarrow$ **Cluster**$(\mathcal{A}, \epsilon)$
  Cleanup-Clusters$(\mathcal{C})$: if for some clusters $c_i \in \mathcal{C}$, for some state s, there exist two pairs (s,a) , (s,a') in c_i, remove one of them arbitrarily.
  best $\longleftarrow 1$
  $\bar{V}_{\text{best}} \longleftarrow \sum_s \beta(s)\bar{V}_t(s) + \sum_{\bar{A}\in\mathcal{C}_1} \bar{A}$
  for $i \in \{2, \ldots, |\mathcal{C}|\}$ **do**
    $\bar{V} \longleftarrow \sum_s \beta(s)\bar{V}_t(s) + \sum_{\bar{A}\in\mathcal{C}_i} \bar{A}$
    if IsPrefered$(\bar{V}, \bar{V}_{\text{best}})[0]$ **then**
      best $\longleftarrow i$
    $\mathcal{K} \longleftarrow$ **IsPrefered**$(\bar{V}, \bar{V}_{\text{best}})[1]$
  $\pi(s) \longleftarrow \begin{cases} a & \text{if } (s,a) \in c_{\text{best}} \\ \pi(s) & \text{o.w.} \end{cases}$
  $\bar{V}_t \longleftarrow \bar{V}_{\text{best}}$
  $t \longleftarrow t + 1$
until $||\bar{V}_{t-1} - \bar{V}_t|| < \epsilon$
return $\bar{V}_t$

moins d'interaction, parce que plusieurs d'entre eux sont évalués par la même requête. C'est l'idée qui gouverne la classification des avantages.

4 Itération de la Valeur basée sur des Avantages

Cette partie décrit l'algorithme d'*Itération de la Valeur basée sur des Avantages(ABVI)* (voir Algorithme 1). ABVI est inspiré par l'*itération interactive de la valeur (IVI)* de Weng et Zanuttini (2013). Quelques notations sont utiles pour la suite : $\mathcal{K}$ désigne un ensemble de contraintes définissant le polytope Λ ; la fonction *isPrefered* compare ses arguments selon les trois méthodes énumérées dans la partie 3 et renvoie $\mathcal{K}$, modifié si la réponse a comporté l'ajout d'une nouvelle contrainte sur Λ.

Afin de réduire les interactions, on fait un classement sur l'ensembles des avantages $\mathcal{A}$ à chaque itération. En fait, au lieu de comparer $\bar{V}_t(s)$ avec $\bar{V}_t(s) + \frac{A_{s,a}}{\beta(s)}$ pour chaque s, a comme

dans IVI, on essaie d'utiliser des avantages de même direction, pour lesquels la préférence sera donc déjà connue. Pour cela, on utilise la classification hiérarchique avec la norme *CosineSimilarity*, qui est basée sur une mesure de colinéarité. ABVI prend également en compte la distribution de probabilité de l'état initial afin d'assigner un vecteur d'avantage agrégé à l'ensemble des états dans le MDP.

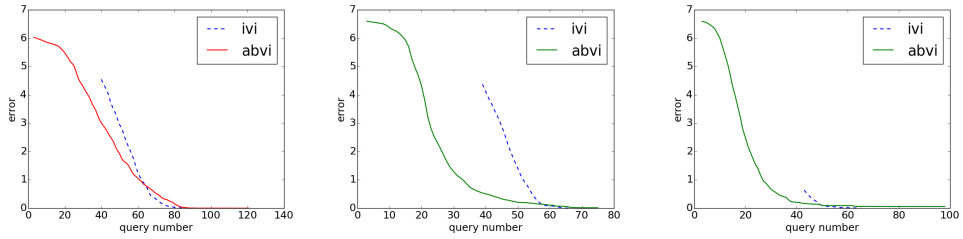
L'algorithme 1 implémente ces idées. Il est initialisé avec une politique aléatoire $\pi_{\text{best}} \in \Pi$ et une fonction de valeur nulle $\forall s \in S,\ \bar{V}_0(s) = \bar{\mathbf{0}}$. A chaque itération, l'algorithme rassemble les $|S||A|$ avantages dans l'ensemble $\mathcal{A}$. La fonction *Cluster* prend les avantages dans $\mathcal{A}$ et les classifie dans $\mathcal{C}$. Chaque cluster c de $\mathcal{C}$ est nettoyé des avantages redondants (si deux avantages s'appliquent au même état) et traduit en une paire (politique, fonction de valeur vectorielle) notée $(\pi_c, \textbf{value}(\pi_c))$ et définie par :

$$\pi_c(s) = \begin{cases} a & \text{si } A_{s,a} \in \textbf{Cluster-Cleanup}(c) \\ \pi_{\text{best}}(s) & \text{autrement} \end{cases}$$

$$\textbf{value}(\pi_c) = \sum_s \beta(s)\bar{V}_t(s) + \sum_{A \in \textbf{Cluster-Cleanup}(c)} A$$

L'algorithme sélectionne alors la meilleure paire de la liste. Si une comparaison portant sur $\textbf{value}(\pi_i) = \bar{V}_i$ et $\textbf{value}(\pi_j) = \bar{V}_j$ nécessite une interaction, une requête de la forme "Est ce que $\bar{V}_i$ est préféré à $\bar{V}_j$?" est proposée au tuteur. $\bar{V}_t$ reçoit la valeur de la meilleure politique π_{best} à chaque itération. La boucle s'arrête quand la variation de $\bar{V}_t$ tombe sous un seuil ϵ

5 Expériences



(a) Sans bruit, $d = 4$ (b) Bruit 0.1%, $d = 4$ (c) Bruit 1%, $d = 4$

FIG. 1: Evolution de l'erreur (axe vertical) v.s. le nombre de requêtes (axe horizontal) pour IVI et ABVI, pour des MDP à 128 états et 5 actions.

Nous avons testé l'algorithme ABVI avec des IRMDP générés de façon aléatoire, et l'avons comparé à IVI (Weng et Zanuttini, 2013). Pour chaque MDP avec n états, m actions et de dimension d, la fonction de transition joint effectivement chaque état à $\lceil \log_2(n) \rceil$ autres états ; la distribution initiale β est uniforme et l'ammortissement $\gamma = 0.95$. Chaque résultat est une moyenne sur 10 MDP aléatoires et le seuil d'arrêt ϵ de l'algorithme 1 est de 10^{-4}. Dans tous les tests, le seuil de classement (le diamètre maximum d'un cluster) pour la métrique cosinus

est 0.01. Enfin, nous avons expérimenté la résistance du système au bruit dû à un utilisateur qui exprime ses propres préférences, et donc la valeur pour lui d'une politique, avec une erreur aléatoire (écart-type = pourcentage de la référence).

La figure 1 illustre l'évolution de l'erreur avec le nombre de requêtes, mesurée à la fin de chaque itération. L'erreur est mesurée par rapport à la valeur espérée, calculée par un algorithme standard d'itération de la valeur avec la valeur λ de référence et une précision de 10^{-5} : $error = ||\bar{\lambda}.\bar{V}_{\text{iteration}} - \bar{\lambda}.\bar{V}_{\text{exact}}||_\infty$. Elle montre que, pour des MDP aléatoires de 128 états, 5 actions et $d = 4$ sans bruit (graphe a), ABVI converge plus vite jusqu'à une erreur réduite (de l'ordre de 10% de l'erreur initiale) ; l'inversion du comportement est due à la classification peu performante des petits avantages à la fin de l'algorithme. Nous cherchons actuellement à résoudre ce problême en variant le seuil de classement ou en supprimant les vecteurs à effet négatif dans les clusters choisis. Les résultats avec bruit ont été obtenus pour une imprécision de 0.1% et 1%. Avec un bruit standard (graphe b), l'avantage de ABVI est accentué. Il semble aussi résister mieux à un bruit plus élevé (graphe c). Ce résultat est cohérent avec le fait que IVI compare des vecteurs plus proches les uns des autres, et que donc le même bruit produit plus de réponses erronées que dans ABVI.

Références

Baird, L. C. (1993). Advantage updating. *Technical report. WL-TR-93-1146, Wright-Patterson Air Force Base*.

Delage, E. et S. Mannor (2007). Percentile optimization in uncertain markov decision processes with application to efficient exploration. *In Proceedings of the 24th International Conference on Machine Learning, ICML*. New York, NY, USA.

Macmahan, H., G. Gordon, et A. Blum (2003). Planning in the presence of cost functions controlled by an adversary. *ICML-03*.

Regan, K. et C. Boutilier (2009). Regret-based reward elicitation for markov decision processes. *UAI-09 The 25th Conference on Uncertainty in Artificial Intelligence*.

Weng, P. (2011). Markov decision processes with ordinal rewards : Reference point-based preferences. *International Conference on Automated Planning and Scheduling*.

Weng, P. et B. Zanuttini (2013). Interactive value iteration for markov decision processes with unknown rewards. *IJCAI*.

Xu, H. et S. Mannor (2009). Parametric regret in uncertain markov decision processes. *48th IEEE Conference on Decision and Control*.

Summary

This paper addresses the *Markov Decision Process* with unknown rewards problem, that is approximating the optimal policy, by transforming it to a *Vector-Valued MDP*. We introduce a new algorithm, ABVI, whose principle is to use value iteration algorithm on VVMDPs. This algorithm classifies *advantage* vectors $\beta(s).(\bar{Q}(s,a) - \bar{V}(s))$ to accelerate value iteration method. We integrate value iteration with querying the user to select appropriate backups. Our goal is to accelerate the value iteration algorithm and to reduce the number of queries.