Planification robuste avec (L)RTDP

Olivier Buffet et Douglas Aberdeen

National ICT Australia &
The Australian National University
{olivier.buffet,douglas.aberdeen}@nicta.com.au
http://rsise.anu.edu.au/∼{buffet,daa}

Résumé : Les problèmes de chemin le plus court stochastique (SSP : Stochastic Shortest Path problem), un sous-ensemble des problèmes de décision markoviens (MDPs), peuvent être efficacement traîtés en utilisant l'algorithme *Real-Time Dynamic Programming* (RTDP). Toutefois, les modèles des MDPs sont souvent incertains (obtenus à l'aide de statistiques ou par intuition). Une approche usuelle est alors la planification robuste : chercher la meilleure politique sous le pire modèle. Cet article montre comment RTDP peut être rendu robuste dans le cas commun où l'on sait que les probabilités de transition se trouvent dans un intervalle donné. Cela permet d'effectuer une planification en tenant compte de l'incertitude d'un modèle appris alors que les approches classiques font l'hypothèse d'un modèle "moyen".

1 Introduction

Pour la planification dans le cadre de la théorie de la décision, les problèmes de décision markoviens (Bertsekas & Tsitsiklis, 1996) sont d'un intérêt majeur quand un modèle probabilitiste du domaine est disponible. Divers algorithmes permettent de trouver un plan (une politique) optimisant l'espérance de l'utilité à long terme. Toutefois, les résultats de convergence vers la politique optimale dépendent tous de l'hypothèse que le modèle probabiliste du domaine est précis.

Malheureusement, un grand nombre de modèles de MDPs sont basés sur des probabilités (et récompenses) incertaines. Nombre d'entre elles dépendent de modèles statistiques de systèmes physiques ou naturels, tels que pour le contrôle d'usines ou l'analyse de comportements d'animaux. Ces modèles statistiques sont parfois basés sur des simulations (elles-mêmes étant des modèles mathématiques), des observations d'un système réel ou une expertise humaine.

Travailler avec des modèles incertains requiert d'abord de répondre à deux questions étroitement liées : 1– comment modéliser l'incertitude, et 2– comment utiliser le modèle résultant. Les travaux existants montrent que l'incertitude est parfois représentée à travers un ensemble de modèles possibles, à chacun étant assigné une probabilité (Munos, 2001). L'exemple le plus simple est celui d'un ensemble de modèles possibles que l'on assume d'égales probabilités (Bagnell *et al.*, 2001; Nilim & Ghaoui, 2004). Mais plutôt

que de construire un ensemble éventuellement infini de modèles, nous choisissons de représenter l'incertitude sur le modèle en définissant chaque probabilité à l'intérieur du modèle comme se trouvant dans un intervalle donné (Givan *et al.*, 2000; Hosaka *et al.*, 2001).

Les probabilités incertaines ont été étudiées dans des problèmes d'allocation de ressources pour trouver le modèle le plus adapté (Munos, 2001) :

– ressource temporelle : comment explorer efficacement (Strehl & Littman, 2004), et
– ressource spatiale : comment aggréger des états (Givan *et al.*, 2000) ;

et dans le but de trouver des politiques robustes (Bagnell *et al.*, 2001; Hosaka *et al.*, 2001; Nilim & Ghaoui, 2004). Nous nous concentrons sur ces derniers, considérant un jeu à deux joueurs où l'adversaire choisi parmi les modèles possibles celui qui dégrade le plus l'utilité à long-terme.

Notre principal objectif est de développer un planificateur efficace pour un sous-ensemble commun de MDPs pour lesquels toutes les politiques optimales ont la garantie de s'arrêter dans un état terminal : les problèmes de chemin le plus court stochastique (SSP : Stochastic Shortest Path). L'algorithme glouton *Real-Time Dynamic Programming* (RTDP) (Barto *et al.*, 1995) est particulièrement adapté aux SSPs, trouvant de bonnes politiques rapidement et ne nécessitant pas une exploration complète de l'espace d'états.

Cet article montre que RTDP peut être rendu robuste, permettant ainsi une planification plus adaptée à un modèle incertain parce qu'appris par expérimentations, voire par intuition. En section 2, nous présentons les SSPs, RTDP et la robustesse. Puis la section 3 explique comment RTDP peut être transformé en un algorithme robuste. Finalement, des expérimentations sont présentées pour analyser le comportement de l'algorithme obtenu, avant une discussion et conclusion.[1]

2 Contexte

2.1 Chemin le plus court stochastique

Un problème de chemin le plus court stochastique (Bertsekas & Tsitsiklis, 1996) est défini ici par un uplet $\langle S, s_0, G, A, T, c\rangle$. Il décrit un problème de contrôle où S est l'ensemble fini des **états** du système, $s_0 \in S$ est un état de départ, et $G \subseteq S$ est un ensemble d'états buts. A est l'ensemble fini des **actions** possibles. Les actions contrôlent les transitions d'un état s à un autre s' selon la dynamique probabiliste du système, décrite par la **fonction de transition** T définie par $T(s, a, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a)$. L'objectif est d'optimiser une mesure de performance basée sur la **fonction de coût** $c : S \times A \times S \to \mathbb{R}^+$.[2]

Les SSP requièrent l'hypothèse qu'il existe une politique propre, c'est-à-dire pour laquelle un état but est accessible depuis tout état dans S, de sorte qu'il n'est pas possible de rester bloqué dans un sous-ensemble d'états. On fait de plus l'hypothèse qu'une

[1] Ce travail est présenté plus en détails dans (Buffet & Aberdeen, 2004).

[2] Le modèle n'étant pas certain, nous ne faisons pas l'hypothèse usuelle $c(s, a) = \mathbb{E}_{s'}[c(s, a, s')]$.

politique impropre conduit à un coût à long terme infini pour au moins un état. Un algorithme de résolution d'un SSP doit trouver une **politique** associant à chaque état une distribution de probabilité sur les actions $\pi : S \rightarrow \Pi(A)$ qui optimise le **coût à long terme** J défini comme l'espérance de la somme des *coûts* pour atteindre un état but.

Dans cet article, nous considérons des SSPs à des fins de planification, avec connaissance complète de l'uplet définissant le problème : $\langle S, s_0, G, A, T, c\rangle$. Dans ce cadre, des algorithmes de programmation dynamique stochastique biens connus tels que *Value Iteration* (VI) permettent de trouver une politique déterministe optimale. Value Iteration fonctionne en calculant la fonction $J^*(s)$ qui donne l'espérance de coût à long terme (finie avec l'hypothèse faite d'existence d'une politique propre) des politiques optimales. C'est le point fixe solution (unique) de l'équation de Bellman :

$$J(s) \quad = \quad \min_{a \in A} \sum_{s' \in S} T(s,a,s') \left[c(s,a,s') + J(s')\right]. \tag{1}$$

Mettre à jour J par cette formule entraîne la convergence asymptotique vers J^*. Pour des raisons pratiques, nous introduisons aussi la Q-valeur :

$$Q(s,a) = \sum_{s' \in S} T(s,a,s')[c(s,a,s') + J(s')].$$

Les SSPs peuvent facilement être vus comme des problèmes de chemin le plus court dans lesquels choisir un chemin ne mène que de manière probabiliste vers la destination espérée. Ils peuvent représenter un sous-ensemble très utile des MDPs, puisqu'il s'agit essentiellement de MDPs à horizon finis.

2.2 RTDP

L'algorithme *Trial based*[3] *Real-Time Dynamic Programming* (RTDP), introduit dans (Barto *et al.*, 1995), utilise le fait que les coûts du SSP sont positifs et l'hypothèse supplémentaire que chaque essai (parcours depuis l'état de départ) atteindra un état but avec une probabilité 1. Ainsi, avec une initialisation nulle de la fonction de coût à long terme J, J comme les Q-valeurs croissent de manière monotone durant leur calcul itératif.

L'idée derrière RTDP (algorithme 1) est de suivre des chemins depuis l'état de départ s_0 en choisissant toujours de manière gloutonne des actions associées au coût à long terme le plus bas, et en mettant à jour $Q(s,a)$ au fur et à mesure que les états s sont rencontrés. En d'autres termes, l'action choisie est celle dont on espère qu'elle mènera aux coûts futurs les plus bas, jusqu'à ce que les calculs itératifs montrent qu'une autre action semble pouvoir faire mieux.

RTDP a l'avantage de vite éviter les plans qui conduiraient à des coûts élevés. Ainsi, l'exploration regarde principalement un sous-ensemble prometteur de l'espace d'états. Toutefois, parce que l'algorithme suit les chemins en suivant la dynamique du système, les transitions rares ne sont prises en compte que rarement. L'utilisation de la simulation permet d'obtenir de bonnes politiques tôt, mais au prix d'une convergence finale lente, du fait de la mauvaise fréquence de mise à jour des transitions rares.

[3] On assumera toujours la version *trial based* de RTDP.

Algorithme 1 Algorithme RTDP pour SSPs

RTDP(s : état) // $s = s_0$
répéter
  ESSAIRTDP(s)
jusqu'à // *pas de condition d'arrêt*

ESSAIRTDP(s : état)
tant que ¬BUT(s) **faire**
  a =ACTIONGLOUTONNE(s)
  J(s)= $Q(s, a)$
  s =CHOISIRETATSUIVANT(s, a)
fin tant que

2.3 Robust Value Iteration

Pessimisme et optimisme

Nous passons maintenant au problème de tenir compte de l'incertitude du modèle lors de la recherche d'une "meilleure" politique. L'ensemble (potentiellement infini) des modèles possibles est noté $\mathcal{M}$.

Une approche simpliste est de calculer le modèle moyen sur $\mathcal{M}$, ou le modèle le plus probable, puis d'utiliser des méthodes d'optimisation standard pour SSPs. De telles approches ne garantissent rien sur le coût à long terme de la politique si le vrai modèle diffère de celui choisi pour l'optimisation.

Nous suivons l'approche décrite dans (Bagnell *et al.*, 2001), laquelle consiste à trouver une politique se comportant bien face au pire modèle possible. Cela revient à considérer un jeu à deux joueurs et à somme nulle, i.e. où le gain d'un joueur et la perte de l'autre. Le joueur choisit une politique sur les actions (dans l'espace de politiques stochastiques Π_A) alors que son adversaire "perturbateur" choisit simultanément une politique sur les modèles (dans l'espace $\Pi_{\mathcal{M}}$). Comme c'est un jeu simultané, les politiques optimales peuvent être stochastiques. Cela mène à un algorithme de type max-min :[4]

$$\max_{\pi_{\mathcal{M}} \in \Pi_{\mathcal{M}}} \min_{\pi_A \in \Pi_A} J_{\pi_{\mathcal{M}}, \pi_A}(s_0).$$

Dans ce jeu SSP, Value Iteration converge vers une solution fixe (Patek & Bertsekas, 1999).

Il est aussi possible d'être optimiste, considérant que les deux joueurs collaborent (du fait qu'ils endurent les mêmes coûts), ce qui transforme le $\max$ en un $\min$ dans la formule précédente. Ce second cas est équivalent à un SSP classique où une décision consiste en le choix d'une action **et** d'un modèle local.

[4] Le jeu étant simultané, l'ordre entre $\max$ et $\min$ est sans importance.

Localité

Un tel algorithme max-min serait particulièrement coûteux à implémenter. Même en restreignant la recherche à une politique déterministe sur les modèles, il faudrait calculer la fonction de coût à long terme optimale pour chaque modèle avant de choisir le pire modèle et la politique optimale associée. Toutefois, un processus plus simple peut être utilisé pour calculer J en cherchant en même temps le pire modèle. Il faut pour cela faire l'hypothèse que les distributions $T(s, a, \cdot)$ sont indépendantes d'une paire état-action (s, a) à l'autre. Cette hypothèse n'est pas toujours valide, mais rend les choses plus faciles pour l'adversaire puisqu'il peut ainsi choisir à travers un ensemble de modèles élargi. On ne risque alors que d'avoir des politiques "trop robustes" (parce que trop pessimistes).

Parce que nous faisons l'hypothèse d'une indépendance au niveau "état-action" (pas seulement au niveau "état"), c'est équivalent à une situation où le second joueur prend une décision dépendant de l'état courant et de l'action du premier joueur. Cette situation revient à un jeu *séquentiel* où le mouvement du joueur précédent est connu du joueur suivant : les deux joueurs peuvent agir de manière déterministe sans perte d'efficacité.

Le résultat de cette hypothèse est que le pire modèle peut être choisi localement quand Q est mis à jour pour une paire état-action donnée. Comme on peut le voir sur l'algorithme 2, le pire modèle local m_s^a peut changer pendant que les Q-valeurs évoluent. De précédentes mises à jour des coûts à long terme d'états atteignables peuvent changer leur ordre relatif, de sorte que les résultats considérés comme les plus mauvais ne sont pas les mêmes.

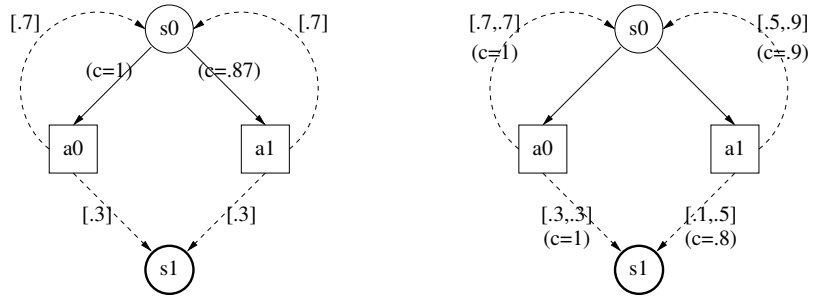
Algorithme 2 Robust Value Iteration (pour un SSP)

Initialiser J to 0.
répéter
  pour tout s : état **faire**
    pour tout a : action **faire**
      $Q_{\max}(s, a) \leftarrow \max_{m_s^a \in \mathcal{M}_s^a} \sum_{s' \in S} T_{m_s^a}(s, a, s') \left[J(s') + c_{m_s^a}(s, a, s') \right]$
    fin pour
    $J(i) \leftarrow \min_{a \in A} Q_{\max}(s, a)$
  fin pour
jusqu'à J converge

L'apport principale de cet article est de montrer que RTDP peut être rendu *robuste*, permettant la planification dans des domaines très grands et incertains, en assurant le comportement dans le pire cas.

3 Robust RTDP

Nous considérons désormais des SSPs incertains **basés sur des intervalles**, où $T(s, a, s')$ se trouve dans un intervalle $[Pr^{\min}(s'|s, a), Pr^{\max}(s'|s, a)]$. La figure 1 montre un exemple d'un tel SSP. Nous discutons l'approche pessimiste, l'optimiste amenant à des résultats similaires.



a) SSP certain b) SSP incertain

FIG. 1 – Deux vues d'un même SSP, selon que l'incertitude sur le modèle est prise en compte (coûts entre parenthèses). Dans le SSP incertain, l'action a_0 sera préférée du fait qu'elle atteint rapidement le but s_1.

Pour une paire état-action donnée (s, a), il existe une liste $R = (s'_1, \cdots, s'_k)$ d'états atteignables (R est choisi pour "reachable"). Pour chaque état atteignable, $T(s, a, s'_i) \in I_i = [p_i^{\min}, p_i^{\max}]$. Ainsi, les modèles possibles sont ceux qui respectent les contraintes représentées par ces intervalles tout en assurant $\sum_i T(s, a, s'_i) = 1$. La figure 2 illustre ceci avec trois états atteignables.

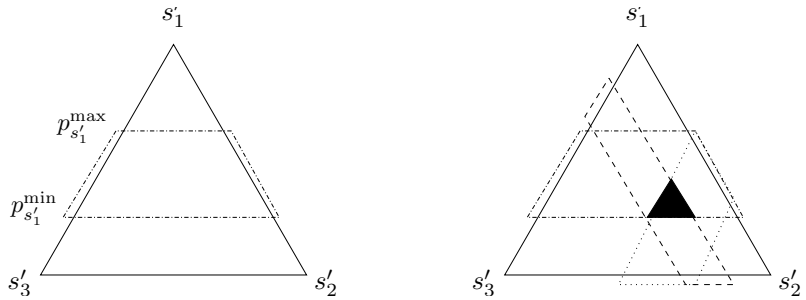


FIG. 2 – Un triangle est un simplexe représentant toutes les distributions de probabilité possibles pour trois résultats différents ($Pr(s'_i) = 1$ au sommet s'_i). Sur le triangle de gauche, le trapèze montre l'intervalle-contrainte pour s'_1. Le triangle de droite montre les modèles possibles à l'intersection des trois intervalles-contraintes.

Pires modèles locaux —

L'étape de maximisation pour calculer $Q(s, a)$ dans l'algorithme 2 est effectuée en donnant la plus grande probabilité au pire résultat. Ceci requiert d'abord d'ordonner les états atteignables de manière décroissante selon les valeurs : $c(s, a, s'_1) + J(s'_1) \geq c(s, a, s'_2) + J(s'_2) \geq \cdots c(s, a, s'_k) + J(s'_k)$. Après, la pire distribution est celle associant la plus grand probabilité au premier état s'_1, puis à s'_2, et ainsi de suite jusqu'à s'_k. Comme indiqué dans (Givan *et al.*, 2000), il est équivalent de trouver l'index $r \in [1..k]$

tel que

$$\sum_{i=1}^{r-1} p_i^{\max} + \sum_{i=r}^{k} p_i^{\min} \leq 1.$$

Les transitions de probabilités résultantes sont alors :

$$Pr(s'_i) = \begin{cases} p_i^{\max} \text{ if } i < r \\ p_i^{\min} \text{ if } i > r \end{cases} \quad (2)$$

$$Pr(s'_r) = 1 - \sum_{i=1, i\neq r}^{k} Pr(s'_i). \quad (3)$$

En utilisant la borne pré-calculée $B_{\min} = \sum_{i=1}^{k} p_i^{\min}$, l'algorithme 3 donne une implémentation complète. L'algorithme de *tri par insertion*[5] est choisi pour profiter de ce que la liste sera souvent déjà ordonnée.

Algorithme 3 Pire modèle pour la paire état-action (s, a)

PIREMODÈLE(s : état, a : action)
$R = (s'_1, \cdots, s'_k) =$ ETATSATTEIGNABLES(s,a)
TRI(R)
$i = 1$, $borne = B_{\min}$
tant que $(borne - p_i^{\min} + p_i^{\max} < 1)$ **faire**
  $borne \leftarrow borne - p_i^{\min} + p_i^{\max}$
  $Pr(s'_i) \leftarrow p_i^{\max}$
  $i \leftarrow i + 1$
fin tant que
$r = i$
$Pr(s'_r) \leftarrow 1 - (borne - p_r^{\min})$
pour tout $i \in \{r + 1, \ldots, k\}$ **faire**
  $Pr(s'_i) \leftarrow p_i^{\min}$
fin pour
return $(R, Pr(\cdot))$

En résumé, Robust VI sur un SSP basé sur des intervalles consiste à appliquer Value Iteration tout en mettant à jour les probabilités de transition à travers l'algorithme 3.

Nous n'avons besoin que d'un seul pire modèle pour calculer les pires Q-valeurs. Toutefois, parce que plusieurs états atteignables s'_i peuvent avoir la même valeur $c(s, a, s'_i) + J(s'_i)$ que s'_r (on note cet ensemble d'états S'_r), il peut y avoir une infinité de pire modèles locaux équivalents. Tout modèle ne différant que par la distribution de la masse de probabilité parmi les états également mauvais de S'_r est aussi un pire modèle local.

Pires modèles globaux —

Contrairement à VI, RTDP ne visite pas nécessairement tout l'espace d'états. C'est pourquoi (Barto *et al.*, 1995) introduit la notion d'état *pertinent* ("relevant state"), que

[5] `http ://en.wikipedia.org/wiki/Insertion_sort`

nous étendons au cas incertain : un état s est dit être *pertinent* pour $\mathcal{M}$ s'il existe un état de départ s_0, un modèle $m \in \mathcal{M}$ et une politique optimale π sous ce modèle tels que s peut être atteint de l'état s_0 quand le contrôleur utilise cette politique sous ce modèle.

Cette notion est importante parce que deux modèles locaux également mauvais sur une paire état-action peuvent interdire l'accès à différents états, de sorte que pour deux modèles m_1 et m_2, un état peut être pertinent (dans le sens de (Barto *et al.*, 1995)) dans m_1 mais pas dans m_2. Mais RTDP ne devrait pas trouver une politique optimale juste pour les états pertinents d'un seul pire modèle global. Et la politique ne doit pas s'appliquer à tous les états possibles. Elle devrait s'appliquer à tous les états *atteignables* sous *tout* modèle (pour des politiques optimales), i.e. à tous les états pertinents. Mais couvrir les états pertinents du pire modèle utilisé pour ré-évaluer les Q-valeurs ne couvre pas nécessairement tous les états pertinents pour $\mathcal{M}$: cela dépend du modèle utilisé pour choisir l'état suivant, c'est-à-dire pour simuler la dynamique du système.

Pour éviter de manquer des états pertinents, chaque modèle local utilisé pour la simulation doit assurer que tout état atteignable (d'après $\mathcal{M}$) peut être visité. Comme on peut le voir sur la figure 2, l'ensemble des modèles locaux possibles pour une paire état-action est un polytope convexe à n dimensions. Tout modèle *à l'intérieur* de ce polytope, excluant la frontière, est ainsi approprié puisque, pour tout s'_i, il garantie que $P(s'_i|s,a) > 0$.

Ainsi il existe un modèle global m_d qui peut être employé pour simuler la dynamique du système sans manquer quelque état potentiellement atteignable qu'il soit.

3.1 Robust (Trial-Based) RTDP

Robust RTDP diffère du RTDP original en ce que :

- A chaque fois que l'algorithme met à jour l'évaluation d'un état, l'adversaire cherche le pire modèle local, utilisé pour calculer les Q-valeurs.
- Pour l'exploration, l'algorithme suit une dynamique possible du système qui tient compte de toutes les transitions possibles (utilisant le modèle m_d).
- Les états "pertinents" sont maintenant les états atteignables en suivant une politique optimale sous n'importe quel modèle possible.

De là, nous pouvons adapter à notre contexte le théorème de convergence 2 de (Barto *et al.*, 1995), ainsi que la preuve correspondante, en discutant principalement les modifications qu'elle requiert.

Théorème 1
Dans des problèmes de chemin le plus court stochastique **incertain** *et avec atténuation,* **robust** *Trial-Based RTDP, avec l'état initial de chaque essai restreint à un ensemble d'états de départ, converge (avec probabilité un) vers J^* sur l'ensemble des états pertinents, et la politique optimale du contrôleur converge vers une politique optimale (éventuellement non-stationnaire) sur l'ensemble des états pertinents, sous les mêmes conditions que le théorème 3 dans (*Barto et al.*, 1995).*

Preuve :

La preuve dans (Barto et al.*, 1995) montre que les états indéfiniment mis à jour par RTDP sont les états pertinents, de sorte qu'une preuve de convergence classique sur les SSP peut être invoquée.*

Une première remarque est qu'introduire $\max_{m \in \mathcal{M}}$ *dans la formule de mise à jour ne change pas le fait que* J_t *est croissante et non-surestimante.*

Dans notre cas, l'utilisation du modèle m_d *assure de manière similaire que les états indéfiniment mis à jour par robust RTDP sont tous les états pertinents (du SSP incertain).*

Nous avons établi que nous sommes dans un jeu séquentiel de type chemin le plus court ("Stochastic Shortest Path Games"=SSPG). (Bertsekas & Tsitsiklis, 1996) montre qu'il s'agit de cas particuliers de SSPG général (simultané). La convergence pour les SSPGs généraux est prouvée dans la proposition 4.6 de (Patek & Bertsekas, 1999), laquelle établie que les coûts à long terme convergent avec une probabilité 1 *sur l'ensemble des états pertinents.* □

Quel que soit le modèle réel, l'algorithme apprend toutes les décisions optimales pour tout état pertinent sous l'hypothèse la plus pessimiste. Un état pertinent s peut d'ailleurs ne pas être atteignable à travers un pire modèle global, mais l'environnement réel peut y mener. Ainsi la politique doit couvrir tous les états pertinents mais assume que le pire modèle s'applique depuis ces états.

4 Expérimentations

Labelled RTDP (Bonet & Geffner, 2003) est une version modifiée de RTDP qui peut être rendue robuste de manière similaire. Les expériences effectuées illustrent le comportement de *robust LRTDP*. Dans ce but, il est comparé au *Robust Value Iteration* de Bagnell, ainsi qu'à *LRTDP*. Dans tous les cas, le critère de convergence est $\epsilon = 10^{-3}$:

- pour LRTDP, un état s a convergé si ses enfants aussi et si son résidu $|J_{t+1}(s) - J_t(s)|$ est plus petit que ϵ, et
- pour VI, nous nous arrêtons quand le plus grand changement dans le coût à long terme d'un état au court d'une itération est plus petit que ϵ.

4.1 Cœur

Dans cette première expérimentation, nous comparons une politique non-robuste optimale avec une robuste sur le petit exemple de la figure 1-b. Le tableau 1 montre les coûts à long terme espérés théoriques de chaque politique sur le modèle normal (plus probable), ainsi que sur les modèles pessimistes et optimistes. La politique robuste est largement meilleure dans le cas pessimiste. On a ici un exemple caricatural du fait qu'une politique robuste fait usage de transitions moins incertaines qu'une politique qui est optimale pour le modèle le plus probable, d'où une moins grande variabilité de son efficacité quand elle est évaluée sur divers modèles.

TAB. 1 – Evaluation théorique de politiques robustes et non-robustes sur divers modèles, concordant exactement à l'évaluation empirique.

	Normal	Pessimiste	Optimiste
Non-robuste	2.90	8.90	1.70
Robuste	3.33	3.33	3.33

4.2 La voiture sur la montagne

Nous employons ici le problème de la voiture sur la montagne tel que défini dans (Sutton & Barto, 1998) : partant du fond d'une vallée, une voiture doit acquérir assez d'élan pour atteindre le haut d'une montagne (voir figure 3). La dynamique utilisée est la même que décrite dans le logiciel "mountain car".[6] L'objectif est de minimiser le nombre de pas de temps pour atteindre le but.

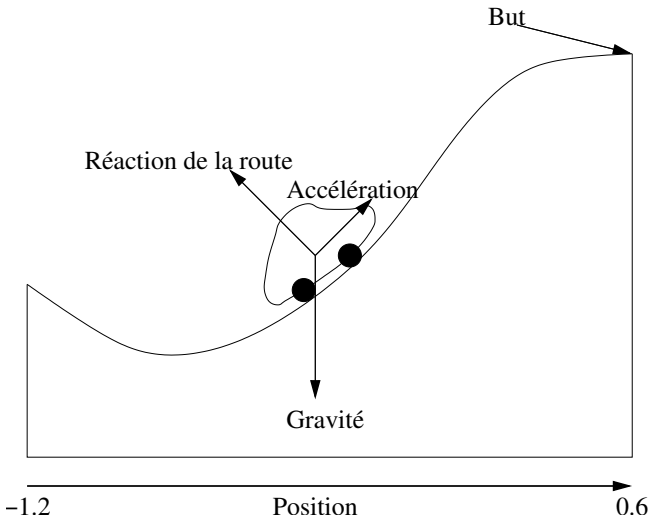


FIG. 3 – Le problème de la voiture sur la montagne.

L'espace d'état continu est discrétisé (grille 32×32) et le modèle de transitions incertaines correspondant est obtenu par échantillonage de 1000 transitions depuis chaque paire état-action (s, a). Pour chaque transition, on calcule les intervalles dans lesquels se trouve le vrai modèle avec une confiance de 95%. Ce calcul décrit en annexe B dans (Buffet & Aberdeen, 2004) utilise la variance empirique et assure que le modèle le plus probable satisfait les contraintes obtenues (on a donc toujours au moins un modèle possible).

[6] `http ://www.cs.ualberta.ca/~sutton/MountainCar/MountainCar.html`

Résultats

Remarque préliminaire : simuler un chemin montre généralement une voiture oscillant plusieurs fois avant de quitter la vallée. Ceci a deux explications principales : 1- la vitesse acquise est juste suffisante pour atteindre le sommet, et 2- le modèle discrétisé n'est pas assez précis : appliquer la politique obtenue sur le vrai modèle mathématique (au lieu du discrétisé) devrait donner de bien meilleurs résultats.

La figure 4 montre la fonction de coût à long terme obtenue en utilisant *value iteration*, LRTDP et leur contreparties robustes sur le problème de la voiture sur la montagne. Les axes x et y donnent la position et la vitesse de la voiture. L'axe z est l'espérance du coût jusqu'au but. Sur la surface est représenté un exemple de chemin depuis l'état de départ jusqu'à l'état but : il suit la politique gloutonne face au modèle le plus probable.

La forme générale de la surface obtenue est toujours la même, avec des parties de l'espace d'états inexplorées par LRTDP et Robust LRTDP (comme attendu). Les échelles verticales sont bien plus grandes dans les cas robustes. Cela reflète le fait qu'atteindre le but consomme bien plus de temps sous un modèle pessimiste. Parce que J peut ici être interprêté comme le temps moyen avant d'atteindre le but, ces graphes montrent comment l'accumulation de petites incertitudes peut amener à des politiques plus longues. Ici les temps sont multipliés par plus de 2.5.

Lors de l'exécution des quatre différents algorithmes, une évaluation de la politique gloutonne courante était effectuée tous les $10 * nStates = 10\,240$ mises à jour d'une Q-valeur. Les résultats apparaissent sur les figures 5 a) et b), l'axe des ordonnées donnant l'espérance de coût à long terme depuis l'état de départ. Sur les deux sous-figures, les algorithmes basés sur LRTDP obtiennent de bonnes politiques rapidement, mais ont de longs temps de convergence de : VI=2.46×10^6 mises à jour, LRTDP=9.00×10^6, rVI=8.09×10^6, rLRTDP=11.5×10^6.

Une dernière mesure intéressante est la "Value-at-Risk" (VaR). La VaR donne, pour un seuil de "risque" $r \in [0, 1]$, le coût à long terme J' tel que $Pr(J > J') \leq r$. La figure 6 a) montre les courbes de Value-at-Risk pour trois modèles possibles (moyen, bon et mauvais) et pour une politique optimale normale et une politique optimale robuste (d'où un total de 6 courbes). Dans ce cas précis, les courbes de l'une et l'autre politiques se superposent, leurs comportements étant identiques sur chacun des trois modèles. Il semble donc que la politique optimale normale soit déjà robuste.

4.3 Navigation maritime

Le problème de navigation utilisé ici partage des similarités avec la voiture sur la montagne. Sa description complète peut être trouvée dans (Vanderbei, 1996), et une autre utilisation se trouve dans (Peret & Garcia, 2004). Ici, l'espace est discrétisé en une grille de 10×10, $\times 8$ angles de vent et $\times 8$ directions possibles. L'incertitude de ce système est due à l'évolution stochastique de la direction du vent. Le modèle incertain est aussi appris en tirant 1000 échantillons au hasard pour chaque paire état-action, en utilisant la même confiance de 95%.

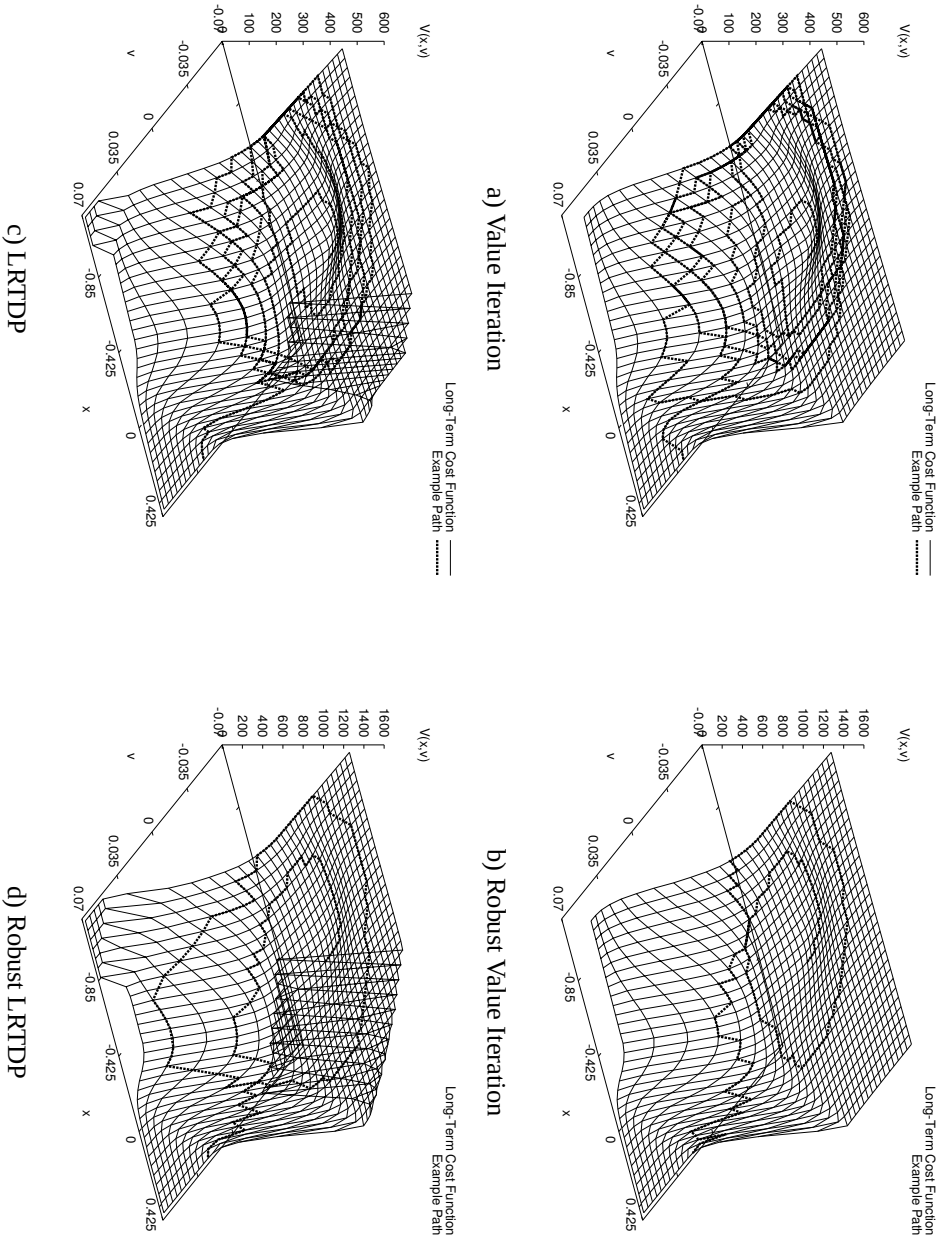


FIG. 4 – Fonction de coût à long terme pour le problème de la voiture sur la montagne Dans tous les cas, le modèle le plus probable est utilisé pour générer un exemple de chemin.

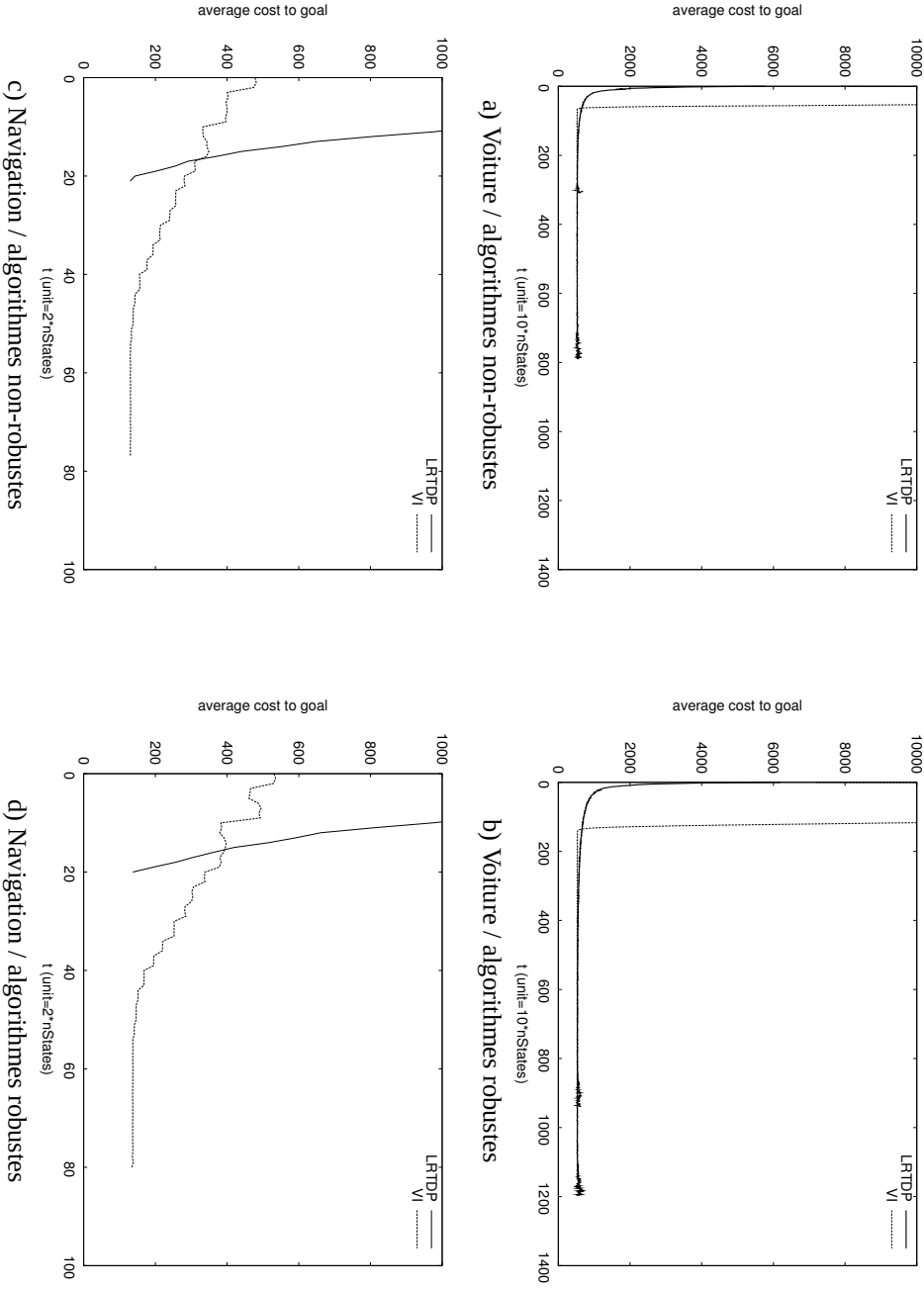
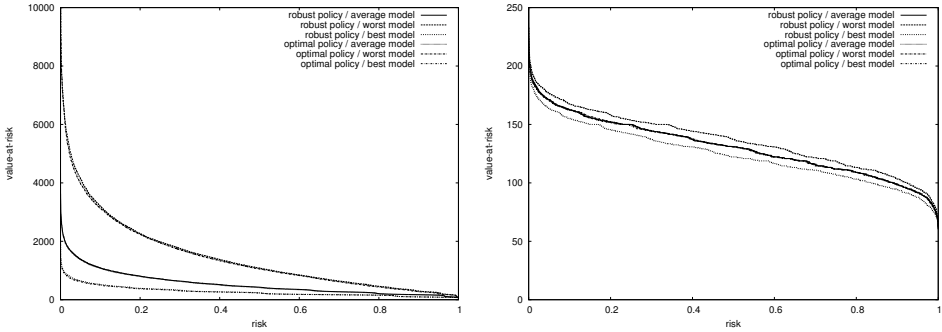


FIG. 5 – Coût moyen pour atteindre le but pour le problème de la voiture sur la montagne (et le problème de navigation), coût mesuré toutes les $10 * nStates$ (resp. $2 * nStates$) mises à jour de Q-valeurs.

Résultats

Les mêmes tests ont été effectués que sur le problème de la voiture sur la montagne. Les fonctions de coût à long terme obtenues montrent des phénomènes similaires tels que l'augmentation du temps pour atteindre le but. Seules les figures 5 c) et d) sont d'un intérêt particulier, puisqu'elles montrent combien les algorithmes de la famille LRTDP convergent vite. Dans ce problème au plus grand nombre de dimensions, trouver des solutions prend plus de temps au début, mais LRTDP s'avère très efficace pour éliminer les chemins inefficaces. En fait, la plupart des états pertinents se trouvent le long de la diagonale principale du lac (la plupart des états latéraux peuvent être évités par les politiques optimales). Pour les différents algorithmes, le temps de convergence est : VI=3.67×10^6, LRTDP=0.49×10^6, rVI=5.22×10^6, rLRTDP=0.60×10^6.

Pour finir, la figure 6 b) donne les courbes de Value-at-Risk pour trois modèles possibles (moyen, bon et mauvais) et pour une politique optimale normale et une politique optimale robuste. Contrairement au problème de la voiture sur la montagne, on observe un comportement différent selon que la politique est robuste ou non. En fait, dans le cas de la politique robuste, les 3 courbes correspondants aux 3 différents modèles employés sont presque confondues (avec la courbe "politique optimale normale / modèle moyen"). On en déduit que la recherche d'une politique robuste amène à des prises de décisions différentes "uniformisant" la prise de risque : la probabilité de dépasser un certain coût à long terme est la même quel que soit le modèle réel du système.



a) Voiture sur la montagne b) Navigation

FIG. 6 – Courbes de Value-at-Risk pour les deux problèmes considérés et 3 modèles de référence.

Une autre expérimentation (Buffet & Aberdeen, 2004) confirme ces résultats sur un exemple illustrant cette approche sur un problème de planification temporelle. Dans ce cas, l'incertitude vient de ce que les probabilités d'échec des différentes tâches n'est connue que par consultation d'experts du domaine (Aberdeen *et al.*, 2004).

5 Discussion et conclusion

Une extension directe de ce travail, suggérée par (Hosaka *et al.*, 2001), est de trouver les meilleures décisions face aux pires modèles (comme nous le faisons dans cet article), puis de choisir parmi celles-ci les décisions optimales pour un *modèle optimiste*. Cette idée a été développée dans l'annexe C de (Buffet & Aberdeen, 2004). Si les calculs supplémentaires requis sont raisonnables, ils ne sont utiles que si diverses politiques robustes équivalentes existent, avec une variabilité de leurs résultats sur des modèles optimistes.

L'approche de la robustesse adoptée dans ce papier considère que l'on connait un ensemble de modèles possibles. Une question ouverte est de savoir s'il est possible d'utiliser plus d'information issue du modèle incertain en prenant en compte la distribution de probabilité sur les modèles possibles.

De manière similaire, l'incertitude sur le modèle a été considérée pour apprendre un modèle pendant la planification (Strehl & Littman, 2004). L'algorithme proposé est optimiste, mais ne semble pas s'adapter à notre cadre dans la mesure où l'évolution du modèle brise l'hypothèse de "non-surestimation" : $\forall s \in S,\ t \geq 0,\ J_t(s) \leq J^*(s)$. Il reste toutefois important de noter que *robust* RTDP ne souffrirait pas d'être utilisé en ligne, puisque la dynamique réelle peut être employée pour choisir l'état suivant (le pire modèle n'apparaît que dans la formule de mise à jour du coût à long terme).

Enfin, une hypothèse cruciale de RTDP est qu'un état but doit être atteignable depuis tout état. Nous présentons un algorithme répondant à ce problème dans (Buffet, 2004).

Conclusion

Des travaux récents montrent que l'incertitude du modèle est un problème important pour la planification dans le cadre de la théorie de la décision. Il peut être intéressant aussi bien d'analyser le modèle pour savoir où il pourrait être raffiné, que prendre des décisions en tenant compte de l'incertitude connue. Nous avons proposé une modification de l'algorithme RTDP lui permettant de calculer des politiques robustes efficacement dans des domaines de grande taille et incertains. L'incertitude sur le modèle est représentée à travers des intervals de confiance sur les probabilités de transition. La preuve de convergence de l'algorithme résultant est esquissée (détails dans (Buffet & Aberdeen, 2004)). Nous faisons la démonstration de robust LRTDP sur un domaine où les intervalles sont estimés de manière statistique.

Remerciements

Grand merci à Sylvie Thiébaux pour son aide et ses encouragements, et aux relecteurs pour leurs pertinentes remarques.

Le National ICT Australia est financé par le gouvernement australien. Ce travail a aussi bénéficié du soutien du DSTO (Australian Defence Science and Technology Organisation).

Références

ABERDEEN D., THIÉBAUX S. & ZHANG L. (2004). Decision-theoretic military operations planning. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS'04)*.

BAGNELL J., NG A. Y. & SCHNEIDER J. (2001). *Solving Uncertain Markov Decision Problems*. Rapport interne CMU-RI-TR-01-25, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

BARTO A., BRADTKE S. & SINGH S. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, **72**.

BERTSEKAS D. & TSITSIKLIS J. (1996). *Neurodynamic Programming*. Athena Scientific.

BONET B. & GEFFNER H. (2003). Labeled rtdp : Improving the convergence of real time dynamic programming. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS'03)*.

BUFFET O. (2004). *Robust (L)RTDP : Reachability Analysis*. Rapport interne, National ICT Australia.

BUFFET O. & ABERDEEN D. (2004). *Planning with Robust (L)RTDP*. Rapport interne, National ICT Australia.

GIVAN R., LEACH S. & DEAN T. (2000). Bounded parameter markov decision processes. *Artificial Intelligence*, **122**(1-2), 71–109.

HOSAKA M., HORIGUCHI M. & KURANO M. (2001). Controlled markov set-chains under average criteria. *Applied Mathematics and Computation*, **120**(1-3), 195–209.

MUNOS R. (2001). Efficient resources allocation for markov decision processes. In *Advances in Neural Information Processing Systems 13 (NIPS'01)*.

NILIM A. & GHAOUI L. E. (2004). Robustness in markov decision problems with uncertain transition matrices. In *Advances in Neural Information Processing Systems 16 (NIPS'03)*.

PATEK S. D. & BERTSEKAS D. P. (1999). Stochastic shortest path games. *SIAM J. on Control and Optimization*, **36**, 804–824.

PERET L. & GARCIA F. (2004). On-line search for solving markov decision processes via heuristic sampling. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'2004)*.

STREHL A. L. & LITTMAN M. L. (2004). An empirical evaluation of interval estimation for markov decision processes. In *Proceedings of the Sixteenth International Conference on Tools with Artificial Intelligence (ICTAI'04)*.

SUTTON R. & BARTO G. (1998). *Reinforcement Learning : an introduction*. Bradford Book, MIT Press, Cambridge, MA.

VANDERBEI R. J. (1996). Optimal sailing strategies, statistics and operations research program. University of Princeton, http ://www.sor.princeton.edu/˜rvdb/sail/sail.html.