Problèmes Décisionnels de Markov Temporels

—

Formalisation et Résolution

—

THESE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITE DE TOULOUSE

délivré par l'**Institut Supérieur de l'Aéronautique et de l'Espace**

Ecole doctorale : Systèmes
Spécialité : Systèmes Embarqués
Unité de recherche : Equipe d'accueil ISAE-ONERA CSDV
présentée et soutenue par

Emmanuel Rachelson

le 23 mars 2009

—

JURY

Patrick Fabiani	Ingénieur de recherche ONERA, co-directeur de thèse.
Frédérick Garcia	Directeur de recherche INRA, co-directeur de thèse.
Michail G. Lagoudakis	Professeur Technical University of Crete, examinateur.
Michael Littman	Professeur Rutgers University, examinateur.
Rémi Munos	Directeur de recherche INRIA, rapporteur.
Olivier Sigaud	Professeur Université Paris 6, rapporteur.
Olivier Teytaud	Chargé de recherche INRIA, examinateur.

Résumé

Ce travail de recherche s'intéresse au problème de planification dans l'incertain en environnement instationnaire. Notre motivation première provient du problème consistant à construire un agent logiciel autonome, capable de se coordonner avec l'évolution incertaine de son environnement. Cet environnement peut être constitué d'autres agents communiquant leurs intentions, ou de processus concurrents non-contrôlables pour lesquels un modèle est disponible sous une forme ou une autre. Nous explorons différentes approches de modélisation d'une dépendance temporelle continue dans le cadre des Processus Décisionnels de Markov (MDP), ce qui nous mène à définir le cadre des Problèmes Décisionnels de Markov Temporels. Puis, nous nous intéressons à deux paradigmes en particulier. En premier lieu, nous considérons les problèmes stochastiques temporels de décision via les modèles à *événements implicites* et les décrivons au travers du formalisme des MDP dépendants du temps (TMDP). Nous étendons la portée de résultats précédents concernant l'existence d'une équation d'optimalité et présentons un nouvel algorithme d'Itération de la Valeur fondé sur les propriétés des représentations polynômiales par morceaux. Cet algorithme permet d'étendre la résolution des TMDP à un plus large classe de problèmes. Ces conclusions ouvrent alors la voie à une discussion plus générale au sujet des actions paramétriques dans le cadre des MDP à espaces d'états et d'actions hybrides et à temps continu observable. Dans un second temps, nous considérons la possibilité de modéliser séparément les contributions concurrentes d'événements exogènes au système. Cette approche de modélisation à *événements explicites* nous mène au cadre des Processus Décisionnels Semi-Markoviens Généralisés (GSMDP). Nous établissons un lien entre le cadre général de la théorie de Spécification des Systèmes à Evénements Discrets (DEVS) et le formalisme des GSMDP, permettant ainsi la spécification de simulateurs compatibles avec tout système à événements discrets. Puis nous définissons une approche d'Itération de la Politique fondée sur la simulation pour construire des politiques de contrôle des Problèmes Décisionnels de Markov Temporels à événements explicites. Cette contribution algorithmique rassemble des résultats de simulation, de recherche en avant pour les MDP et d'apprentissage statistique. L'approche à événements implicites est évaluée sur un problème de planification d'un rover martien et sur un problème de patrouille d'un drone, tandis que l'approche à événements explicite est testée sur un problème de contrôle d'un réseau de métro.

Mots clés

Décision dans l'incertain, Processus Décisionnels de Markov, problèmes de planification dépendant du temps, espaces d'états hybrides, modélisation de processus décisionnels stochastiques dépendant du temps, contrôle de processus à événements discrets implicites ou explicites.

Remerciements

Quand vient le temps de rédiger les remerciements d'une thèse de doctorat, c'est que la rédaction est achevée ou dans les dernières phases de correction. Du moins est-ce mon cas et c'est avec des sentiments partagés entre le soulagement (cette longue rédaction est finie), le plaisir (ces dernières années ont tout simplement été extraordinaires) et la gratitude que j'ouvre ces dernières pages.

Avant tout, je souhaite remercier Rémi Munos et Olivier Sigaud, les deux rapporteurs de ma thèse, pour avoir accepté de relire ce (long) document.

Mes remerciements vont également au jury de soutenance dans son ensemble pour leur lecture du manuscrit et leur participation à la présentation.

Puis j'aimerais remercier l'ONERA et tout le personnel du département de Commande des Systèmes et Dynamique du Vol pour avoir rendu ces travaux possibles, financièrement et matériellement. Ma gratitude se porte en particulier vers l'équipe Conduite et Décision au sein de laquelle règne une ambiance motivante et accueillante qui m'a fourni de nombreuses raisons de me remettre en question au fil de ces trois années. Il y a là trop de noms pour tous les mentionner : ma gratitude va à tous pour avoir fait de ces trois ans une expérience humaine autant que scientifique.

Je souhaite également remercier les personnes avec lesquelles j'ai eu l'occasion d'interagir, au LAAS-CNRS et dans l'équipe de Biométrie et Intelligence Artificielle de l'INRA.

De même, alors que je me retourne vers ces trois années, je prends conscience de ce que je dois aux nombreuses et passionnantes discussions que j'ai pu avoir avec les membres des communautés MDP et Apprentissage par Renforcement. J'y ai trouvé de nombreux acteurs passionnés, travaillant dans un domaine exigeant et avides d'échanges sur leurs travaux.

J'aimerais également remercier en particulier Sylvie Thiébaux pour ses conseils et son soutien pendant cette dernière année. Je te souhaite (pour le moins) un bon retour en Australie et espère avoir de tes nouvelles bientôt.

Alors que je me rapproche de plus en plus des gens présents au quotidien lors de ma thèse, j'ai une pensée particulière pour Jean-Loup Farges. A la fin de mon stage de M2R je te remerciais déjà pour ton soutien "discret et efficace". Je ne prenais pas conscience à

l'époque d'à quel point c'était vrai : Jean-Loup est un pilier, présent et disponible à tout moment, exprimant des avis critiques sur des sujets insoupçonnés qui ont grandement contribué à la rigueur et à l'amélioration de ce travail.

Alors que nous partagions le même bureau lors de mon stage de M2R, Florent Teichteil a prit un fort ascendant sur mes idées de recherche. Merci pour ces *quatre* années, pour ton soutien et tes conseils.

J'aimerais à présent remercier Patrick Fabiani, mon premier directeur de thèse, qui a lancé ce sujet avec — je suppose — une idée de recherche assez différente de ce que j'ai développé au final et un nouveau poste de directeur de département accaparant beaucoup de son temps, mais qui a trouvé un intérêt à la problématique que je lui proposais et m'a encouragé à continuer. L'incroyable énergie que tu déploies dans des tentatives plus ou moins fructueuses pour te rendre disponible n'a d'égal que la justesse de tes conseils.

Frédérick Garcia a été, je pense, le directeur de thèse parfait pour moi. Je ne peux dresser une liste de toutes les raisons pour lesquelles te remercier : c'est un tout, qui commence avec la très grande gentillesse et s'étend loin au delà de la passion que tu mets dans les questions scientifiques. Dans les coups durs comme dans les moments faciles nous avons au final échangé bien plus que des idées de recherche.

Je suis convaincu que ces trois années n'auraient pas été les mêmes sans le groupe assez surprenant de doctorants que nous formions. Ainsi, avec une délicatesse toute contestable puisque je m'inclus dans les remerciements, j'aimerais nous remercier, tous, pour les bons et les mauvais moments que nous avons passés, tant au labo qu'en dehors. Ma thèse n'aurait clairement pas été la même sans *nous*.

J'aimerais également exprimer ma très profonde gratitude à l'inventeur anonyme de la coinche (ou belote contrée). Je vous dois un nombre incalculable d'heures de frustrations, d'argumentations et de plaisir stratégique[1].

Il y a enfin deux amis et collègues que j'aimerais remercier remercier en particulier. Commençons avec Grégory Bonnet. Merci infiniment pour avoir survécu à trois années de cohabitation dans le même bureau : ta patience est légendaire mon ami ! Un grand merci également pour toutes les profondes discussions que nous avons eues : sur les systèmes multiagents, sur les bons et les mauvais films, sur Noam Chomsky, sur la topologie des patatoïdes et surtout, surtout, merci pour l'inoubliable "Subotaï le magnifique".

Enfin, je ne peux finir sans remercier du fond du cœur Julien Guitton. Depuis tes points de vue très clairs sur la planification à la plus improbable et solide des amitiés, j'ai des milliers de raisons de te remercier. Laisse-moi au moins le faire pour les heures sans fin que nous avons passées à discuter de tout et de rien. Récupère, s'il te plaît, pour nous deux, le prix du tournoi de coinche de l'ONERA que nous venons de gagner.

—

Je dédie ce travail à tous mes amis, passés et actuels.
Vous savez à quel point vous comptez à mes yeux.

[1] Pour être tout à fait exact, il me faut également mentionner que je vous dois l'inspiration de certaines idées de recherche, veuillez me contacter pour tout ce qui concerne les droits d'auteur et les citations.

Table des matières

Notations

$*$	Opérateur de convolution
$\|\cdot\|_{I,\infty}$	$\|g\|_{I,\infty} = \sup_{x \in I} \|g(x)\|$
δ	Numéro d'étape d'un processus
ρ	Trajectoire d'exécution
σ	Etat augmenté d'un SMDP+, correspond à (s, t)
μ	Réalisation d'un TMDP
A	Espace d'actions
a	Action
a^δ	L'action choisie à l'étape δ
a_n	Dans un espace d'actions fini, ceci est le $n^{\text{ième}}$ élément de A
$F(\tau\|s, a)$	Fonction de durée d'un SMDP (fonction de répartition)
$f(\tau\|s, a)$	Fonction de durée d'un SMDP (densité de probabilité)
L	Opérateur de Bellman
L^π	Opérateur d'évaluation des politiques
$Pr(X = x)$	Probabilité que la variable aléatoire X vaille x
$P(s'\|s, a)$	Modèle de transition d'un MDP
R	Modèle de récompense par transition d'un MDP : $R(s, a, s')$
r	Modèle de récompense par paire état-action d'un MDP : $r(s, a)$
S	Espace d'états
s	Etat
s^δ	La variable aléatoire "état" à l'étape δ
s_μ	L'état associé à la réalisation μ dans un modèle TMDP
s_n	Dans un espace d'états fini, désigne le $n^{\text{ième}}$ élément de S
V^*	Fonction de valeur optimale
V^{π^*}, V^π	Fonction de valeur de la politique optimale / de la politique π

Préambule

Ce manuscrit en français est un résumé des travaux de thèse. Conformément au directives de rédaction des mémoires de thèse, ce document auto-suffisant fait référence au document compagnon, plus détaillé, rédigé en anglais et suivant le même plan. Ainsi, il est recommandé au lecteur anglophone de se tourner vers le manuscrit en anglais afin de disposer de la version la plus détaillée des travaux.

Le contenu de ce mémoire en français doit cependant permettre au lecteur désirant rapidement parcourir le contenu de la thèse de se faire une idée générale des contributions et des résultats obtenus. Il nécessite toutefois quelques prérequis supplémentaires, rendus nécessaires par la réduction des détails et des discussions.

Par ailleurs, ayant été rédigé après le manuscrit en anglais, ce document présente la qualité d'avoir plus de recul sur les travaux que lors de la rédaction en détail de la thèse.

Première partie

Introduction

1

Bien décider : des exemples aux Problèmes Décisionnels de Markov Temporels

Ce chapitre d'introduction s'ouvre sur une vue générale du contexte de travail de cette thèse, puis se concentre de façon plus précise sur notre domaine d'étude. Nous commençons au niveau "humain", considérant la question générale de la *Décision* et ses différents aspects, motivant ainsi les diverses approches de formalisation de la question et soulignant les caractéristiques des approches formelles. Puis nous passons en revue un certain nombre de caractéristiques définissant différents domaines des *Sciences de la Décision*. Ceci nous permet de cerner plus précisément notre sujet d'étude. Nous nous concentrons alors sur les domaines de la *Planification dans l'Incertain* et de l'*Apprentissage par Renforcement*, au sein desquels nous définissons enfin la classe des *Problèmes Décisionnels de Markov Temporels*.

1.1 La question de la décision

La démarche de décider est au coeur de notre vie de tous les jours. L'acte de décider est un concept qui a fasciné de nombreux penseurs de tous bords : philosophes, mathématiciens, biologistes, psychologues, informaticiens. On peut ramener les enjeux de la décision à la question de choisir une "action" parmi plusieurs disponibles et ce de la manière qui nous semble la plus appropriée.

Du point de vue de l'intelligence artificielle, construire un agent autonome du point de vue de la décision est un vieux rêve. Or définir un tel agent passe par une formulation rigoureuse du problème afin d'en analyser les propriétés mathématiques et de concevoir les outils informatiques nécessaires à sa résolution. Une telle analyse passe par la définition de concepts tels que

- les variables de l'environnement de décision,
- les variables de décision elles-mêmes,
- les relations entre ces variables indiquant quelles sont les configurations et combinaisons possibles de l'environnement,
- la dynamique temporelle de ces relations au fur et à mesure de l'interaction de l'agent avec son environnement, ce qui amène la notion
- d'agent ou d'agents décideur(s).

Nous nous intéresserons dans ce manuscrit à des problèmes :

- de décision séquentielle, impliquant

- un unique agent, interagissant avec
- un environnement présenté comme un système dynamique à événements discrets,
- décrit dans un cadre probabiliste ,
- dont le modèle est fourni comme un système à événements implicites ou comme un simulateur et
- impliquant un unique critère de décision fondé sur l'observation de l'interaction entre l'agent et son environnement.

1.2 Planifier et Apprendre à agir

Le problème de la décision séquentielle sous incertitudes probabilistes a été étudié sous plusieurs aspects qui portent les noms de Planification dans l'incertain, Commande optimale stochastique ou Apprentissage par renforcement. Ces trois approches utilisent la même base de modélisation : les Processus Décisionnels de Markov, mais diffèrent dans les hypothèses qu'elles effectuent sur les problème et sur l'approche de résolution qu'elles envisagent.

1.3 Temps, Incertitudes et Problèmes de décision séquentiels

Des exemples comme le problème de coordination en cours de mission de deux drones patrouillant un territoire inconnu permettent d'illustrer la place de la variable temporelle dans les problèmes de planification dans l'incertain. Supposons pour l'exemple que le concepteur du système souhaite doter les deux drones d'autonomie décisionnelle afin que l'un ne soit pas asservi à l'autre pour planifier ses actions. Il met alors en place un procédé de coordination par déclaration d'intentions et déplace alors le problème de coordination vers un problème qui correspond, pour chaque agent à planifier dans un environnement instationnaire.

Cette instationnarité est ici induite par les intentions de l'autre agent, cependant, de nombreux autres exemples, comme la gestion de mission d'un rover Martien, l'optimisation des décisions de mise en service de lignes de métro dans un réseau urbain, la gestion du roulage au sol des avions dans les aéroports, etc. sont autant d'exemples de décision dans l'incertain en présence de phénomènes instationnaires. On peut alors extraire leur problématique commune comme celle de décider d'une politique d'action optimale, vis-à-vis des contraintes d'instationnarité et d'incertitude du problème ainsi que d'un critère d'optimalité prédéfini.

Sur la base de ces constatations, nous définissons les Problèmes Décisionnels de Markov Temporels comme les problèmes de décision séquentielle présentant les caractéristiques suivantes :

- Systèmes à événements discrets.
- Incertitudes sur les résultats d'action
- Incertitude sur les durées (éventuellement continues) de transition
- Dépendance explicite au temps (variable temporelle observable)

On peut alors rapidement se rendre compte qu'une des principales sources de la complexité d'analyse de tels processus tient à la :

- Concurrence de processus incontrôlables.

Cette caractéristique, bien que peu quantifiable, est cependant au coeur de nos préoccupations dans ce manuscrit.

2

Problèmes Décisionnels de Markov Temporels — Modélisation

La planification est la branche de la théorie de la décision qui traite de la sélection et de l'ordonnancement d'actions de haut niveau afin de parvenir à un certain but ou comportement. Ce chapitre introduit les notions de bases et les formalismes qui seront utilisés tout au long de la thèse. Nous commençons avec le cadre général des Processus Décisionnels de Markov (MDP) afin de modéliser l'incertitude sur les résultats d'action. Nous soulignons alors où se trouvent les difficultés lorsque l'on souhaite introduire un temps observable dans le cadre MDP. Cette discussion nous mènera à l'introduction progressive de modèles spécifiques issus de la littérature afin de modéliser la dépendance temporelle continue dans le cadre de la planification dans l'incertain.

2.1 Processus Décisionnels de Markov

Les Processus Décisionnels de Markov constituent un cadre devenu standard pour représenter les problèmes de décision séquentielle dans l'incertain. Un MDP est défini par la donnée de :

- S, un espace d'états, généralement discret et fini (dénombrable)
- A, un espace d'actions, généralement fini également
- P, une fonction de transition Markovienne, indiquant la probabilité $P(s'|s,a)$ d'atteindre l'état s' après avoir entrepris l'action a dans l'état s.
- r un modèle de récompense associant un gain ou une perte à chaque transition (s,a).

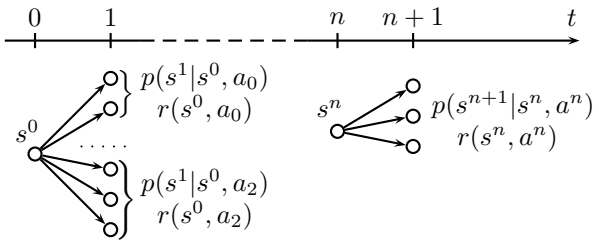


FIG. 2.1 – Evolution d'un MDP

La résolution d'un problème de contrôle d'un MDP se fait généralement au travers de la définition d'une politique π qui associe à chaque état une action. Une telle politique est dite Markovienne et l'on sait que pour les critères d'optimisation usuels, il existe au moins une

politique optimale qui soit Markovienne.

2.2 Temps et MDPs

La question de représenter l'évolution temporelle des processus stochastiques n'est pas neuve et de nombreux formalismes ont été introduits pour capturer différents aspects des problèmes de décision stochastiques et temporels. Nous résumons les relations entre ces différents formalismes sur la figure 2.2.

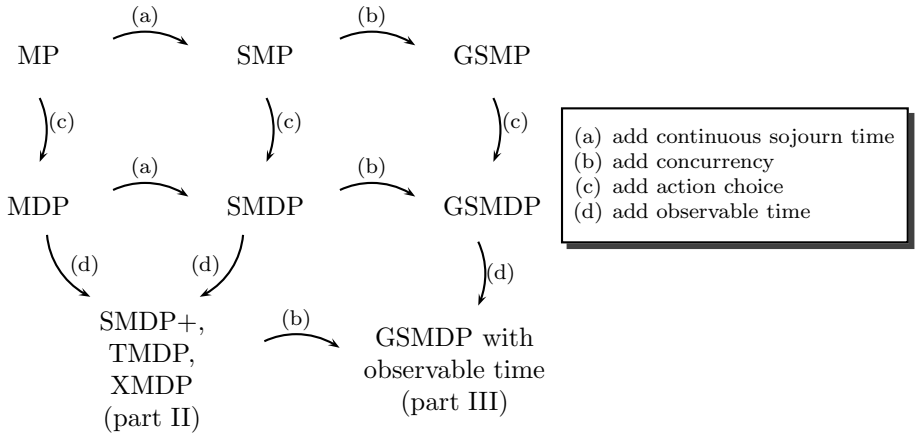


FIG. 2.2 – Carte des relations entre modèles

2.3 Similarités et différences avec les problèmes "classiques" MDP

L'introduction d'un temps continu observable dans le cadre d'un modèle MDP amène quelques remarques et *caveat*. D'une part il est crucial de distinguer trois notions de temps différents lorsqu'on parle de tels processus stochastiques.

- D'une part le temps de la chaîne de Markov sous-jacente au couple MDP-politique. Ce temps est un entier correspondant au nombre d'étapes parcourues par le processus jusque là.
- D'autre part les durées de transition, aussi appelées durées de séjour. Indiquant combien de temps le système passe dans chaque état avant d'entreprendre une nouvelle transition discrète.
- Enfin la variable temporelle usuelle elle-même, le temps de la montre, celui que l'on peut mesurer et considérer comme une variable d'environnement ou comme une ressource.

Par ailleurs, considérer un temps observable dans un problème MDP implique nécessairement de redéfinir la précédente notion d'horizon en regard de notre connaissance de l'instationnarité des phénomènes en jeu. Cela conduit à la distinction entre horizon de planification et horizon temporel et permet d'introduire la notion de pseudo-horizon.

Enfin, il est important de noter que l'introduction d'une variable temporelle observable dans l'espace d'état implique nécessairement une structuration de la dynamique du système.

Cette structuration peut-être vue comme la conséquence du principe de causalité et son exploitation permet d’envisager des approches dédiées aux problèmes temporels.

3 Organisation du manuscrit

Cette thèse est organisée en une partie d'introduction et deux parties de développement. L'introduction, à la fin de laquelle se trouve le présent chapitre, a pour vocation de présenter le domaine d'étude considéré en partant de considérations très générales accessibles à tous, pour arriver à un niveau de modélisation fin de ce que nous nommons "Problèmes Décisionnels de Markov Temporels". Cette introduction pose notamment un certain nombre de bases de modélisation qui sont à l'origine de la structure en deux parties du développement qui suit.

[Boutilier et al., 1999] effectue une distinction nette entre deux paradigmes de représentation de la dynamique des systèmes à événements discrets contrôlables. D'une part, on considère des systèmes à événements implicites. Dans de telles représentation, l'évolution temporelle du système et l'impact des processus extérieurs à l'agent sont directement intégrés dans sa fonction de transition. Ainsi, dans un tel modèle décisionnel à événements discrets, seules les actions de l'agent permettent de faire évoluer l'environnement. La partie II du manuscrit se concentre sur l'analyse de tels modèles de représentation et sur l'algorithmique de résolution des problèmes associés. Cette étude est principalement menée au travers du cadre TMDP de [Boyan and Littman, 2001] et s'ouvre sur des modèles plus généraux.

La partie III en revanche se concentre sur le second paradigme mis en avant par [Boutilier et al., 1999] : les systèmes à événements explicites. Ces systèmes décrivent l'évolution concurrente des processus dynamiques à événements discrets qui composent l'environnement de l'agent, séparément du modèle de ce dernier. Ainsi, de tels systèmes à événements discrets reposent fortement sur l'idée de concurrence d'exécution. Dans la partie III, nous établissons un parallèle entre le formalisme GSMDP de [Younes and Simmons, 2004] et la théorie de spécification des systèmes à événements discrets DEVS. Nous nous intéressons alors aux approches d'apprentissage par renforcement exploitant la simulation des GSMDP afin de construire incrémentalement des politiques de contrôle appropriées. Ce travail nous emmène notamment au sein de problèmes d'apprentissage statistique que nous cherchons à résoudre pour représenter les politiques et fonctions de valeur apprises par notre algorithme ATPI.

Deuxième partie

Planifier en fonction d'un temps continu observable dans le cadre des processus décisionnels de Markov

4

Un lien entre SMDP et TMDP : le modèle SMDP+

La partie précédente a fourni une introduction générale aux modèles de prise en compte des conséquences temporelles d'actions dans le cadre MDP. Le formalisme TMDP de [Boyan and Littman, 2001] semble constituer un cadre naturel et adapté à la modélisation de la dépendance au temps dans un problème MDP. Toutefois, la place d'un tel modèle dans la famille des processus stochastiques et, par exemple, le lien avec les processus décisionnels à événements discrets et à durées continues comme les SMDP est encore mal exploré. Dans ce chapitre, nous nous attachons à établir un tel lien et à répondre à la question "les TMDP sont-ils des SMDP avec temps observable ?". Pour cela, nous introduisons le modèle SMDP+ afin d'illustrer le critère optimiser et de clarifier la question de la passivité dans les TMDP.

4.1 Introduire un temps observable dans le cadre SMDP

Nous définissons un SMDP+ comme un SMDP incluant, dans son ensemble de variables d'état, la variable temporelle. Cela conduit à la définition suivante : un SMDP est constitué de

- Σ, un espace d'état augmenté contenant les paires (s, t) et pouvant être décomposé en :
  - un ensemble d'états discrets $s \in S$,
  - une variable temporelle continue $t \in \mathbb{R}$.
- A, un espace d'actions discrètes.
- $Q(\sigma'|\sigma, a)$, la fonction de répartition du modèle de transition. Cette dernière peut être décomposée en $Q(\sigma'|\sigma, a) = P(s'|s, t, a) \cdot F(t'|s, t, a, s')$. Par commodité, on pourra noter indifféremment le modèle de durée $f(t'|s, t, a, s')$ où $f(\tau|s, t, a, s')$, avec :

$$f(t'|s,t,a,s') = \begin{cases} 0 & \text{if } t' < t \\ f(\tau = t' - t|s,t,a,s') & \text{if } t' \geq t \end{cases}$$

- $R(\sigma', a, \sigma)$, le modèle de récompense.

Une telle définition implique de repréciser le critère γ-pondéré et l'équation de Bellman associée :

$$V^\pi = E\left(\sum_{\delta=0}^{\infty} \gamma^{t_\delta} r_\delta^\pi | \sigma_0\right) \tag{4.1}$$

$$V^{\pi}(\sigma) = \sum_{s' \in S} \int_0^{\infty} \big(R(s', t+\tau, \pi(\sigma), \sigma) + \gamma^{\tau} V^{\pi}(\sigma')\big) \cdot f(\tau|\sigma, \pi(\sigma), s') P(s'|\sigma, \pi(\sigma)) d\tau = L_{\pi}^{t}(V^{\pi})(\sigma) \tag{4.2}$$

$$V^{*}(\sigma) = \max_{a \in A} \left\{ \sum_{s' \in S} \int_0^{\infty} \big(R(s', t+\tau, a, \sigma) + \gamma^{\tau} V^{*}(\sigma')\big) \cdot f(\tau|\sigma, a, s') P(s'|\sigma, a) d\tau \right\} \tag{4.3}$$

$$V^{*}(\sigma) = LV^{*}(\sigma)$$

4.2 Le problème de la passivité dans le modèle SMDP+

Plusieurs choses sont à noter lorsqu'on introduit un temps continu dans un SMDP. D'une part, il vient naturellement à l'esprit d'introduire alors une action "attendre". Cependant cette action ne peut être définie que relativement à des paramètres de durée d'action ou de date de fin d'action. Par ailleurs, définir une telle action implique de définir un modèle de transition correspondant, illustrant le fait que dans un modèle TMDP, l'hypothèse implicite par défaut est que "attendre" est une action déterministe qui n'affecte que la variable temporelle. On ne capture donc pas tous les cas de SMDP à temps continu avec le cadre TMDP.

Définir une notion d'attente dans le cadre SMDP+ le plus général implique nécessairement de définir une action paramétrique continue correspondante, malgré l'intuition qui semble indiquer que l'on peut simplifier le problème en introduisant une action déterministe comme dans le cas TMDP.

4.3 Quelle est alors la différence entre "attendre" et "ne rien faire" ?

L'idée de ne rien faire est absente du concept de système décisionnel à événements discrets implicites car l'évolution d'un tel système est conditionnée par la prise de décision de l'agent. S'il n'y a pas d'action, il n'y a pas d'évolution.

Introduire l'absence d'action sur une certaine durée ne correspond pas non plus à une action "attendre" car cela implique alors de définir un problème de décision hybride en terme de description temporelle du système. "ne rien faire" est une action comparable aux commandes rencontrées en commande optimale à temps continu tandis que toutes les autres actions restent propres aux systèmes à événements discrets.

Au final, c'est bien la définition d'une action "attendre", associée à une gamme de paramètres possibles, qui permet de définir rigoureusement une modèle de transition et de récompense pour l'inclusion dans un système décisionnel à événements discrets.

4.4 Définition des politiques SMDP+

Une politique SMDP+ est donc définie comme une application :

$$\pi \ : \ \left\{ \begin{array}{ccc} \Sigma & \rightarrow & A+ \\ s, t & \mapsto & a \end{array} \right. \tag{4.4}$$

où $A+$ est un espace d'actions augmenté qui comprend toutes les instances possibles de l'action "attendre". Il faut alors mettre à jour les deux équations d'évaluation des politiques et d'optimalité présentées plus haut.

4.5 Le lien entre TMDP et SMDP+

On peut alors montrer que les TMDP, qui décomposent toute transition en l'entreprise d'une action puis la réalisation d'un "outcome" comme illustré à la figure 4.1, sont un cas particulier de SMDP+.

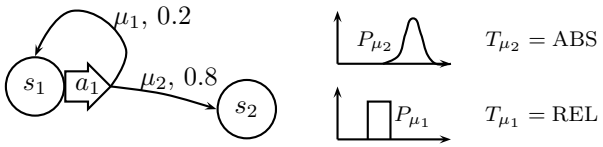


FIG. 4.1 – TMDP - éléments de base

On montre séparément les trois points suivants :
- Equivalence des modèles : il est possible de transformer exactement un TMDP en SMDP+. La transformation inverse est possible sous certaines hypothèses.
- Equivalence des équations d'optimalité sous hypothèse de critère total pour les SMDP+.
- Equivalence de l'exécution de deux politiques, l'une spécifiée dans le cadre TMDP, l'autre dans le cadre SMDP+.

4.6 Conclusion

Nous avons établi le lien entre le cadre TMDP et le cadre SMDP, au travers de l'introduction d'un modèle plus générique nommé SMDP+. Nous avons ainsi fourni ainsi une base théorique plus solide à l'analyse des équations d'optimalité des TMDP et à la compréhension des questions d'"attente" ou de "passivité" dans les systèmes à événements discrets. Par ailleurs, cette analyse a permis de montrer à la fois les limites d'expressivité du modèle TMDP et les raccourcis dans sa formulation qui en font un modèle facile d'usage.

5

Résolution des TMDP par programmation dynamique

Le précédent chapitre établissait le lien entre le cadre TMDP et les processus décisionnels stochastiques classiques. Nous y avons notamment illustré le fait que l'équation d'optimalité TMDP correspondait à un critère total. Nous pouvons à présent nous concentrer sur la résolution de cette équation. Notre objectif est d'analyser les raisons qui ont permis une résolution exacte dans le cas de [Boyan and Littman, 2001], de déterminer si et comment nous pouvons étendre cette résolution et d'introduire les outils de calcul qui nous permettront d'effectuer des itérations de Bellman exactes dans le cadre TMDP.

5.1 Equations d'optimalité et propriétés de la fonction de valeur

La lecture de ce chapitre présuppose une certaine connaissance du modèle TMDP de [Boyan and Littman, 2001]. Notre objectif dans ce chapitre va être de trouver une suite de fonctions V_n appartenant toutes au même espace de fonctions (solution à forme fermée) et solutions du système récurrent d'équations :

$$V_{n+1}(s,t) = \sup_{t' \geq t} \left(\int_t^{t'} K(s,\theta) d\theta + \overline{V}_n(s,t') \right) \tag{5.1}$$

$$\overline{V}_n(s,t) = \max_{a \in A} Q_n(s,t,a) \tag{5.2}$$

$$Q_n(s,t,a) = \sum_{\mu \in M} L(\mu|s,t,a) \cdot U_n(\mu,t) \tag{5.3}$$

$$U_n(\mu,t) = \begin{cases} \int_{-\infty}^{\infty} P_\mu(t')[R(\mu,t,t') + V_n(s'_\mu,t')]dt' & \text{if } T_\mu = \text{ABS} \\ \int_{-\infty}^{\infty} P_\mu(t'-t)[R(\mu,t,t') + V_n(s'_\mu,t')]dt' & \text{if } T_\mu = \text{REL} \end{cases} \tag{5.4}$$

Par ailleurs, nous nous attacherons à chercher une telle famille de fonctions de façon à faciliter sa mise en oeuvre pratique dans un algorithme de type itération de la valeur.

5.2 Les fonctions polynômiales par morceaux

Notre choix s'est tourné vers l'étude des fonctions polynômiales par morceaux. Il y a de nombreuses raisons à cela :

– La résolution exacte de [Boyan and Littman, 2001] s'est effectuée dans le cadre des fonctions constantes et linéaires par morceaux. Passer au cadre polynômial semble donc une extension logique.

– La famille des fonctions polynômiales et polynômiales par morceaux est bien connue et étudiée, notamment dans le cadre de la théorie des splines [Ahlberg et al., 1967].
– Par ailleurs, pour la représentation de densités de probabilités, si les distributions classiques (Bêta, Gaussienne, exponentielle, ...) sont couramment utilisées, il est difficile d'effectuer des calculs formels exacts dessus. Les fonctions polynômiales par morceaux sont, à ce titre, nettement plus faciles à manipuler.
– De plus, afin de modéliser des quantités évoluant de façon discontinue (modèle de récompense par exemple) la définition "par morceaux" fournit un cadre d'étude adapté.
– Enfin, la connaissance des densités de probabilité et autres fonctions nécessaires à la définition de nos modèles n'est précise qu'à un certain écart près. Approcher, par exemple, une Gaussienne, par une fonction polynômiale par morceaux correspondante est acceptable en regard de l'incertitude sur les fonctions du modèle en premier lieu.

5.3 Recherche d'une solution stable à l'équation de Bellman

En suivant la propagation d'une fonction V_n au travers du système d'équations présenté ci-dessus, nous montrons que si le modèle est composé de fonctions polynômiales par morceaux et de distributions discrètes ou polynômiales par morceaux, alors il est préférable de chercher les éléments de la suite $\{V_n\}$ au sein des fonctions polynômiales par morceaux. La figure 5.1 illustre notamment la propagation d'une fonction au travers de l'équation 5.1.

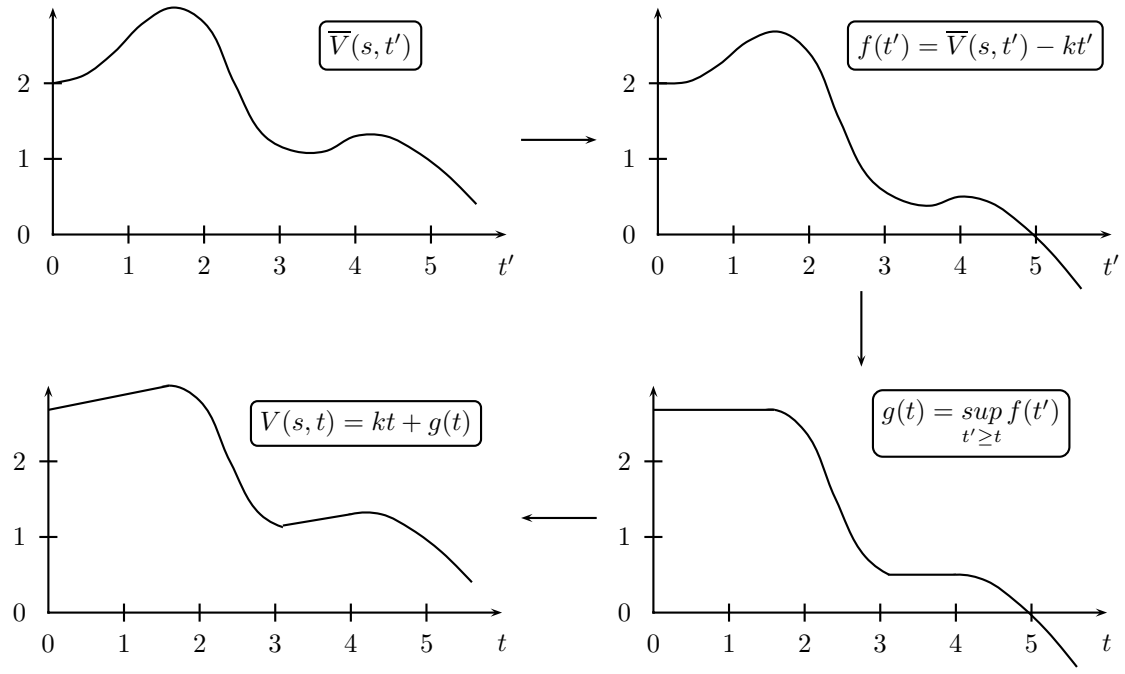


FIG. 5.1 – Illustration de l'équation 5.1

5.4 Borner le degré des polynômes utilisés

Notons $d^\circ(\cdot)$ l'opérateur donnant le degrés d'une fonction polynômiale par morceaux et supposons que :

- $d^\circ(P_\mu) = A$
- $d^\circ(r) = B$
- $d^\circ(L) = C$

Si l'on suppose que $d^\circ(V_0) = 0$, on peut alors écrire que le degré de V_1 est $D_1 = A + B + C + 1$, celui de V_2 est $D_2 = 2A + B + 2C + 2$, etc. Ainsi le degré de V augmente nécessairement avec les itérations à moins que $A + C = -1$.

Or, en regard de la propriété d'évolution du degré des polynômes lors d'une opération de convolution, la famille des distributions de probabilité discrètes se comporte comme un polynôme de degré -1. Nous concluons ainsi que le seul cas possible de stabilité du degré de V_n correspond aux modèles pour lesquels $C = 0$ et $A = -1$.

Nous retrouvons ainsi un résultat qui inclut le résultat de résolution exacte de [Boyan and Littman, 2001]. Le degré de V_n reste stable au travers des itérations si :

- P_μ est une distribution discrète
- L est une fonction constante du temps
- r est une modèle de trois fonctions indépendantes, polynômiales par morceaux de degré quelconque B.

5.5 Est-il possible d'étendre la méthode de résolution exacte ?

Partant du dernier résultat, l'extension de la résolution exacte nécessite au minimum de se situer dans le cadre défini ci-dessus. Cependant, un tel cadre ne garantit que la stabilité du degré de V_n, il faut donc vérifier que chacune des opérations des équations de programmation dynamique peut s'effectuer de façon exacte.

Nous montrons finalement qu'une telle résolution ne peut se faire que si le degré B est inférieur ou égal à 4 en raison de notre incapacité à trouver systématiquement des racines de polynômes de degré supérieur à 5.

Nous avons donc retrouvé quasiment le même résultat que [Boyan and Littman, 2001] en cherchant à l'étendre. Cette exploration des propriétés des équations 5.1 à 5.4 nous permet de cerner où se trouvent les difficultés associées à un cadre polynômial par morceaux et ouvrent ainsi la voie à la généralisation d'un algorithme d'itération de la valeur sur des TMDP dans un cadre exact ou approché.

6

L'algorithme $TMDP_{poly}$: résolution de TMDP généralisés

Pour des TMDP définis avec des fonctions polynômiales par morceaux pour les modèles de transition, de récompense et de durées, les itérations de Bellman peuvent être effectués via un calcul analytique, permettant une recherche de solution optimale dans la famille des fonctions de valeur polynômiales par morceaux. La chapitre précédent a posé les bases et les limites d'une telle résolution exacte et à degré stable. Sur cette base, nous introduisons l'algorithme $TMDP_{poly}$ qui combine les calculs analytiques des itérations de Bellman (calculs impliquant des résolutions exactes ou approchées), une approximation en norme infini des fonctions de valeur et une recherche par programmation dynamique ordonnée afin de résoudre le cas général des TMDPs à fonctions polynômiales définies par morceaux.

6.1 Extensions à la résolution exacte des TMDP : conclusions et propriétés

1. Si le TMDP à résoudre satisfait les conditions énoncées au chapitre précédent alors l'algorithme d'itération de la valeur produit une séquence de fonctions polynômiales par morceaux de degré stable.
2. La résolution d'un TMDP peut découpler la recherche d'actions de la durée d'attente et ainsi alterner des phases d'attente et d'action car l'action "attendre" n'a pas d'effet sur l'état discret du système et est déterministe vis-à-vis de la variable temporelle.
3. Alterner ces phases permet de découpler l'équation d'optimalité et de simplifier le problème comme indiqué par les équations 5.1 à 5.4.
4. Tenir compte de la variable temporelle observable correspond alors à organiser les passes de programmation dynamique afin de tenir compte du principe de causalité. L'idée de mettre à jour en premier les états proches de l'état souhaité final est généralisé par l'algorithme de Prioritized Sweeping de [Moore and Atkeson, 1993].

6.2 Calcul exact d'une itération de la valeur

Cette section décrit en détail chacune des opérations nécessaires en pratique à la résolution successive des équations 5.4 à 5.1. Nous nous attardons en particulier sur l'équation 5.2 pour laquelle nous introduisons l'algorithme 6.1.

Algorithm 6.1: Construire $\overline{V}$ et $\overline{\pi}$ à partir de fonctions Q polynômiales par morceaux

$t_0 = 0$ `/* Initialization */`
$t_{\text{inter}} = 0$
a_{sup} is the dominating* action in t_0
$a_{sup\_new} = a_{sup}$
$\overline{\pi}[t_0] = a_{sup}$**
while $t_{\text{inter}} \neq \infty$ **do** `/* While intersections are found */`
  $t_{\text{inter}} \leftarrow \infty$ `/* Earliest intersection found so far after` t_0 `*/`
  for $a \in A \setminus \{a_{sup}\}$ **do**
    $t_{cand} = +\infty$ `/* Candidate for earlier intersection */`
    $test(t) = Q(s,t,a) - Q(s,t,a_{sup})$
    $t_{0\_shifted} = t_0$
    $\mathcal{I}$ = definition interval of $test$ to which t_0 belongs
    if $test(t)$ is equal to zero on $\mathcal{I}$ **then** `/* Case : equivalent actions in` t_0 `*/`
      if $\mathcal{I}$ is the last definition interval of $test(t)$ **then**
        $t_{0\_shifted} = +\infty$
      else if the next interval of $test$ starts before t_{inter} **then**
        $\mathcal{I}_{next}$ = the next interval
        $t_{0\_shifted} = \mathcal{I}_{next}$.lower_bound()
        if a dominates a_{sup} in $t_{0\_shifted}$ **then**
          $t_{cand} = t_{0\_shifted}$
          $t_{0\_shifted} = +\infty$
      else
        $t_{0\_shifted} = +\infty$
    if $t_{0\_shifted} \leq t_{\text{inter}}$
    then `/* Case :` Q `functions intersection */`
      t_{cand} = first point of sign change of $test(t)$ in $[t_{0\_shifted}, t_{\text{inter}}]$ ($+\infty$ if none)
    $t_{new} = +\infty$
    if $t_{cand} < t_{\text{inter}}$ **then** `/* Successful candidate found */`
      $t_{\text{inter}} = t_{cand}$
      $a_{sup\_new} = a$
    else if $t_{cand} = t_{\text{inter}}$ **then** `/* Triple Intersection */`
      This is the case of three Q functions intersecting at t_{cand} : a, a_{sup} and a_{sup_new}.
      a_{sup_new} dominates a so we only need to check if a dominates a_{sup_new}.
      if a dominates a_{sup_new} in t_{cand} **then**
        $t_{\text{inter}} = t_{cand}$
        $a_{sup\_new} = a$
  if $t_{\text{inter}} \neq +\infty$ **then**
    $\overline{\pi}[t_{\text{inter}}] = a_{sup\_new}$
    $a_{sup} = a_{sup\_new}$
    $t_0 = t_{\text{inter}}$

* dominating : highest value or, in the case of value equality, highest first non zero derivative.
** $\overline{\pi}$ is defined on the interval starting in t_0 as a_{sup}, this interval's upper bound will be provided by the next interval's lower bound.

6.3 La méthode de “Prioritized sweeping”

L’idée derrière l’approche de Prioritized Sweeping est que lors d’une résolution par programmation dynamique, l’ordre de mise à jour des états importe peu tant que chacun est visité une infinité de fois quand le nombre d’itérations tend vers l’infini. Cela permet de construire des algorithmes d’itération de la valeur asynchrones qui concentrent les efforts de mise à jour sur certains états puis sur d’autres.

Nous introduisons une version de l’algorithme de [Moore and Atkeson, 1993], adaptée au cadre TMDP. Cette version présente notamment quelques améliorations concernant le calcul des Q-valeurs, l’introduction possible d’un biais dans la pondération, l’initialisation de la file de priorité et de la fonction de valeur ainsi que sur les critères d’arrêt. La synthèse de cette méthode est décrite à l’algorithme 6.2.

Algorithm 6.2: Prioritized Sweeping pour TMDP

Init : $V \leftarrow h$, $priority\_queue \leftarrow$ UnprioritizedVI(), $continue = true$.
while $continue = true$ **do**
  while $priority\_queue \neq \emptyset$ **do**
    Remove the top state from $priority\_queue$. Call it s'
    $V(s', t)$.BellmanBackup() `/* equations 5.2 and 5.1 */`
    foreach $(s, a) \in predecessors(s')$ **do**
      $Q(s, t, a)$.BellmanUpdate() `/* equations 5.4 and 5.3 */`
      $Prio(s, a) = \|Q(s, t, a) - Q_{old}(s, t, a)\|_{t \in [0,T], \infty}$
      if $Prio(s, a) > \epsilon$ and $Prio(s, a) > Prio(s)$ **then**
        Insert s in $priority\_queue$ with $Prio(s) = Prio(s, a)$
  $priority\_queue \leftarrow$ UnprioritizedVI()
  if max_priority($priority\_queue$) $< \epsilon$ **then**
    Either take a smaller ϵ or set $continue = false$.

6.4 Optimisation approchée pour les TMDP

Enfin, afin de faciliter les calculs concernant les manipulations de polynômes définis par morceaux, nous nous fondons sur les résultats connus d’itération de la valeur approchée, notamment sur les mesures d’optimalité en norme infini, et introduisons l’algorithme 6.3 qui permet à la fois :

- d’approcher n’importe quel polynôme défini par morceaux par un polynôme défini par morceaux de degré inférieur,
- de borner l’erreur d’approximation en norme infini,
- de limiter au maximum le nombre d’intervalles de définition de la fonction.

L’idée de base de l’algorithme 6.3 est illustrée par la figure 6.1. Il s’agit à la fois de découper la fonction pour mieux l’approcher mais aussi de regrouper les intervalles sur lesquelles une approximation en un morceau est possible. Tout cela étant mis en oeuvre afin d’obtenir un bon compromis entre qualité de l’approximation et temps de calcul.

Algorithm 6.3: L'algorithme d'approximation polynômiale de $TMDP_{poly}$

input: p_{in} /* the pwp to approximate */
input: L /* the approximation's degree */
input: ϵ /* the tolerance on the L_∞ error bound */
input: $[l, u]$ /* the approximation interval */
$p_{out} = p_{in}$
$t_0 = l$
$continue = true$
f, f', g, g_{temp} are polynomials
while $continue = true$ **do**
  I = interval to which t_0 belongs
  $f = p_{out}$.polynomial(I)
  Refinement phase : /* refining the bounds to fit the function */
  if f.degree() $> L$ **then**
    Erase the interval I in p_{out}.
    $refine = true$
    $t_{up} = I$.upper()
    while $refine = true$ **do**
      $f' =$interpolation(f, L, t_0, t_{up})
      if $\|f - f'\|_{[t_0,t_{up}]} > \epsilon$ **then**
        $t_{worse} = \underset{t \in I}{argsup} |f'(t) - f(t)|$
        $t_{up} = t_{worse}$
      else if $\|f - f'\|_{[t_0,t_{up}]} \leq \epsilon$ and $t_{up} \neq I$.upper() **then**
        Set p_{out} to f' on $[t_0, t_{up}[$.
        $t_0 = t_{up}$
        $t_{up} = I$.upper()
      else
        $refine = false$
  Simplification phase : /* trying to swallow successive intervals into one */
  I = interval to which t_0 belongs
  $I_{next} = I$.next_interval()
  $t_{temp} = I_{next}$.upper()
  $g = p_{out}$.polynomial(I)
  $g_{temp}(t) =$interpolation$(p_{out}, L, t_0, t_{temp})$
  while $\|p_{out} - g_{temp}\|_{[t_0,t_{up}],\infty} \leq \epsilon$ **do**
    $g = g_{temp}$
    $t_{up} = t_{temp}$
    $I_{next} = I_{next}$.next_interval()
    $t_{up} = I_{next}$.upper()
    $g_{temp}(t) =$interpolation$(p_{out}, L, t_0, t_{up})$
  Replace all polynomials of p_{out} over $[t_0, t_{up}[$ by g.
  Stopping condition :
  $t_0 = t_{up}$
  if $t_0 \geq u$ **then** $continue = false$

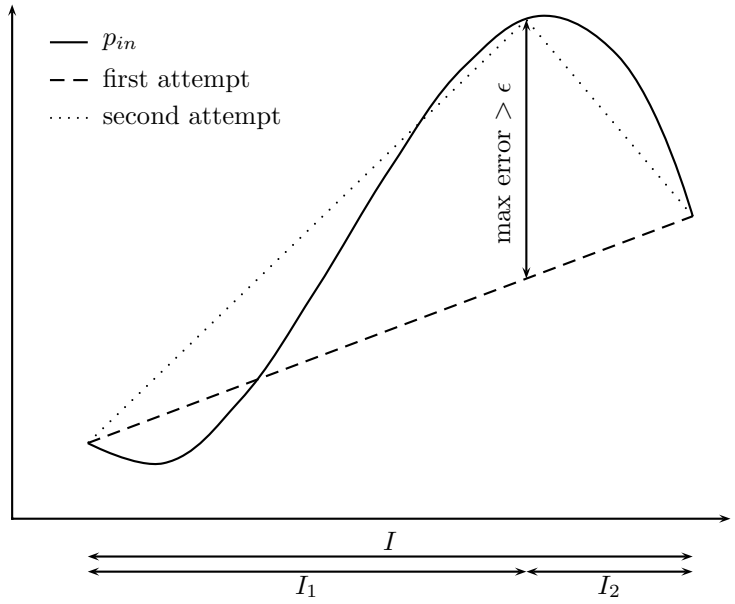


FIG. 6.1 – Illustration de l'algorithme 6.3

6.5 L'algorithme $TMDP_{poly}$

Finalement, l'algorithme $TMDP_{poly}$ combine les contributions précédemment présentées. Il se fonde sur :

- Le calcul analytique des itérations de Bellman pour l'évolution des fonctions de valeur.
- L'algorithme 6.3 pour la réduction de degré et du nombre d'intervalles.
- L'algorithme 6.1 pour la construction et l'assemblage des fonctions de valeur, en version exacte (degré du modèle de récompense inférieur à 5) ou approchée.
- La mise à jour prioritaire des états cruciaux telle que présentée sur l'algorithme 6.2.

7

Implantation et évaluation expérimentale de l'algorithme $TMDP_{poly}$

Ce chapitre présente deux instances particulières de problèmes résolus par le planificateur $TMDP_{poly}$ implémentant l'algorithme $TMDP_{poly}$. Le problème est une adaptation du problème classique du rover Martien, avec incertitudes sur les résultats d'actions et ressource temporelle continue, tel que présenté dans [Bresina et al., 2002]. Nous comparons la résolution de plusieurs versions de ce problème. Le second exemple traite d'une interprétation originale du modèle TMDP. Il s'agit d'un problème de planification d'une mission de surveillance pour un drone aérien. L'action "attendre" n'est plus une action passive mais au contraire l'unique action fournissant des récompenses. En effet, elle correspond à l'action de patrouille sur une zone donnée. Cet exemple généralise l'usage des TMDP à un cadre plus général d'actions continues et ouvre un nouveau point de vue sur les applications des TMDP.

7.1 Choix d'implantation

L'implantation des algorithmes précédents a donné lieu à la création des bibliothèques de fonctions *POLYTOOLS* pour le traitement des fonctions polynômiales par morceaux et $TMDP_{poly}$ pour la définition et la résolution des problèmes TMDP. Cette implantation a été faite en C++ et le code source des bibliothèques est disponible à l'adresse `http://emmanuel.rachelson.free.fr/en/software/`.

Les expériences qui suivent ont été menées sur un ordinateur présentant les caractéristiques suivantes :

Processeur	AMD Athlon 3200+ (single core, 1.8 GHz)
Mémoire vive	1019 Mo
Système d'exploitation	GNU/Linux Ubuntu version 8.04
Compilateur C/C++	gcc 4.2.4

7.2 Exemples simples et premiers résultats du planificateur $TMDP_{poly}$

Le planificateur $TMDP_{poly}$ a tout d'abord été testé sur des exemples simples afin de vérifier son comportement et de s'assurer que des différentes améliorations introduites au chapitre précédent. L'exemple typique de test est un cas d'étude à trois états illustré à la figure 7.1.

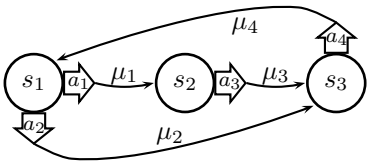


FIG. 7.1 – Problème à trois états - 2nde version

L'étude de ces problèmes simples nous a notamment permis de définir plusieurs métriques quant à l'efficacité de nos algorithmes :

- Nous utilisons le graphe représentant l'évolution des priorités maximales comme une graphe de performance. Ce choix est justifié par le lien étroit entre erreur de Bellman et priorités des états.
- Nous illustrons l'évolution de l'espérance des gains en nous focalisant sur certains états de départ en particulier. Ces états sont choisis afin d'illustrer le comportement de l'agent exécutant la politique trouvée.
- Nous traçons enfin l'évolution du temps nécessaire à la mise à jour d'un état afin d'illustrer la complexité limitée de l'algorithme en pratique.

7.3 Le problème du rover Martien

Le problème de planification de la mission d'observation et d'échantillonage d'un rover Martien est décrit dans [Bresina et al., 2002]. Notre version implique l'incertitude sur les résultats d'action et l'observabilité d'une ressource temporelle continue. Le graphe de navigation du rover est illustré figure 7.2.

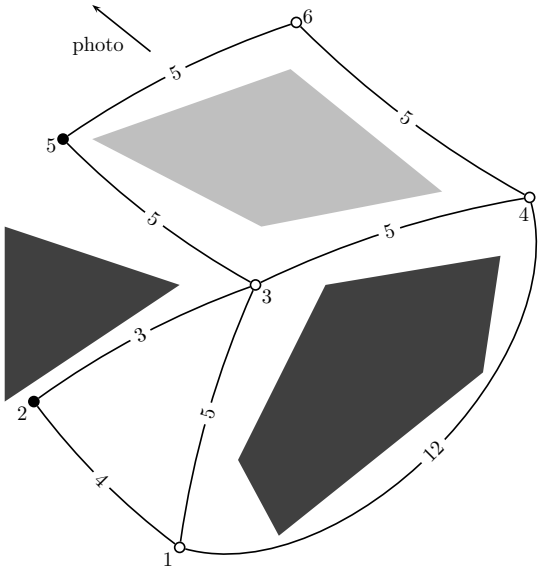


FIG. 7.2 – Problème du rover martien — présentation de la mission

Il s'agit ici d'un problème assez complexe présentant 1968 états discrets et une variables continue. Un problème entièrement discrétisé avec un pas de temps de 1 (pas de temps raisonnable pour représenter les fonction mises en jeu) impliquerait un espace de 139728 états

discrets.

Le rover peut naviguer entre différents sites et a pour mission de prendre une photo et ramasser deux échantillons. L'action "attendre" lui permet de recharger ses batteries.

La figure 7.3 présente l'évolution des priorités associées aux états, directement liées à l'erreur de Bellman. La figure 7.4 présente quant à elle la faible variation des temps de calcul associés aux mises à jour d'un état, malgré la complexification des fonctions de valeur au fur et à mesure des itérations.

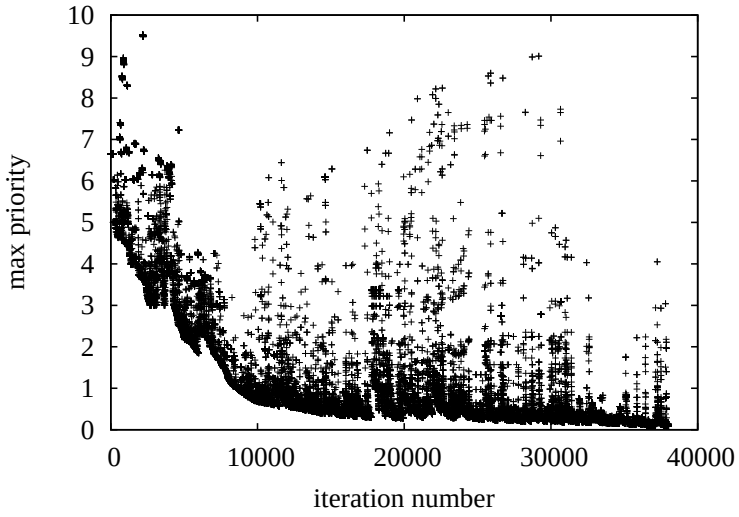


FIG. 7.3 – Evolution des priorités maximum pour le problème du rover Martien

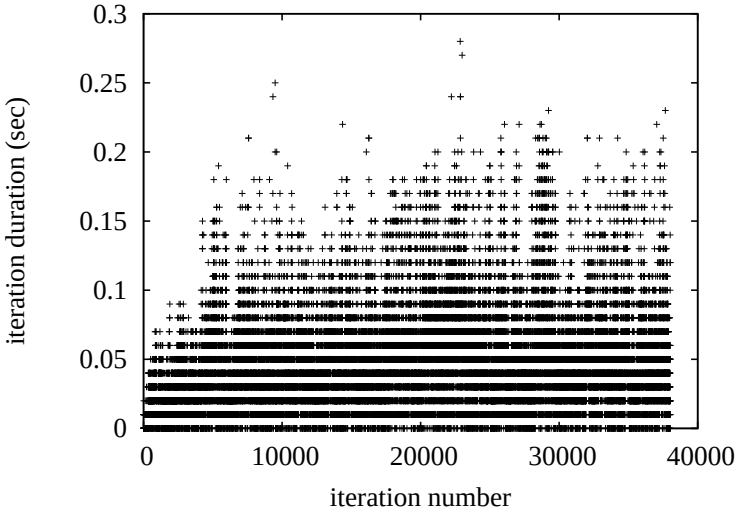


FIG. 7.4 – Evolution de la durée d'une itération pour le problème du rover

Il est possible de suivre l'évolution d'une exécution du rover et l'évolution des fonctions de valeur associées à chaque état en jouant la politique. Nous référons le lecteur à la version

anglaise de la thèse pour plus de détails à ce sujet.

Enfin, afin d'illustrer la structure de la politique et de la fonction de valeur finale, nous traçons sur la figure 7.5, pour un état géographique donné et dans la situation où aucun objectif n'a encore été réalisé, la surface correspondant à la fonction de valeur en fonction des variables "temps" et "énergie".

7.4 Le problème de patrouille d'un UAV

Ce problème de patrouille considère un drone aérien (UAV) devant explorer une carte et passer du temps à patrouiller sur certaines zones en particulier. La spécification de mission peut faire apparaître des conflits d'emploi du temps et une valeur d'importance est accordé à chaque zone en fonction de l'heure de la journée. Ce problème est illustré par la figure 7.6.

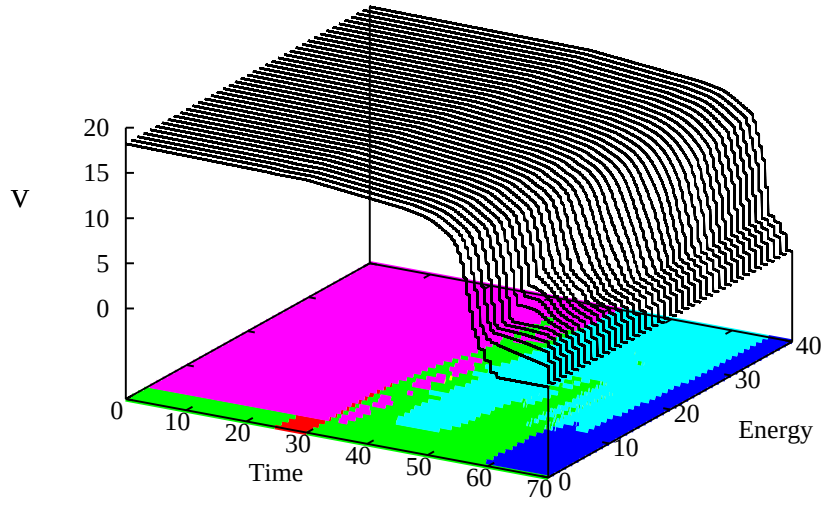
La performance du planificateur $TMDP_{poly}$ sur ce problème a été excellente, lui permettant de trouver des politiques optimales en quelques centaines de mises à jour d'états tandis qu'un algorithme d'itération de la valeur comme celui de [Boyan and Littman, 2001] converge difficilement ou ne converge pas après plusieurs milliers de mises à jour d'états.

Afin d'illustrer l'évolution des mises à jour et les scénarios de résolution, une interface graphique a été implantée pour ce problème et est présentée à la figure 7.7.

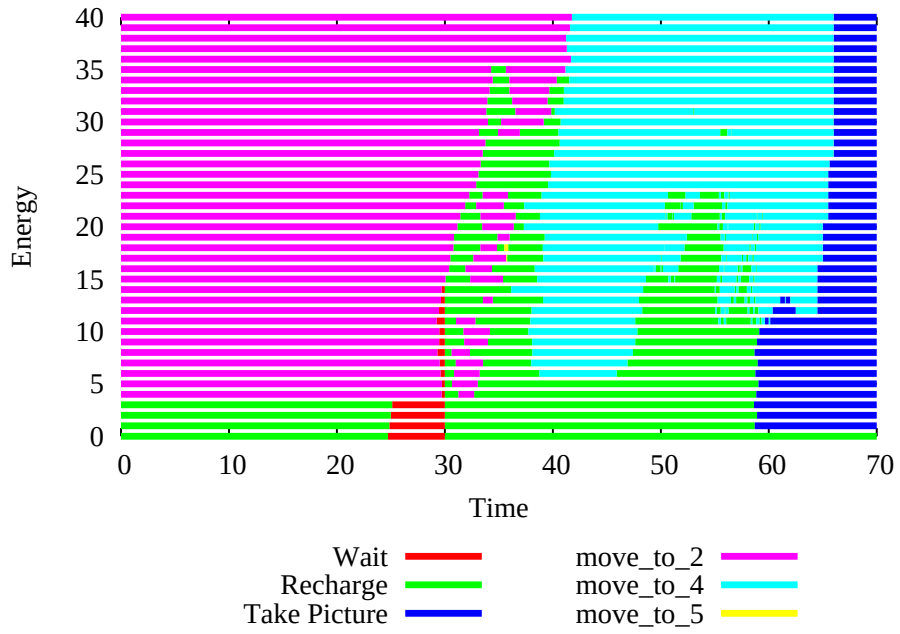
Nos différentes expériences ont notamment montré que sur une grille de navigation à 100 cases, l'algorithme trouvait une fonction de valeur quasi optimale en environ 500 mises à jour d'états. Les états les plus souvent mis à jours étant alors visités moins de dix fois.

7.5 Conclusion

Il reste de nombreuses possibilités d'amélioration de l'algorithme $TMDP_{poly}$ et de son implantation. Naturellement, l'amélioration des temps de calcul associés à la bibliothèque *POLYTOOLS* bénéficiera directement à l'implantation de $TMDP_{poly}$. Par ailleurs, d'autres voie ont été suggérées plus haut et n'ont pu être extensivement testées. Il parait notamment pertinent de coupler la méthode actuelle de mise à jour des états avec une initialisation heuristique efficace pour la fonction de de valeur (les expériences ci-dessus étant menées avec une initialisation uniforme à zéro). Le fait de biaiser les priorités de mise à jour des états tout en conservant la garantie d'optimalité afin de mieux tenir compte du principe de causalité est également un axe d'amélioration prometteur. Ce ne sont toutefois que des exemples des possibilités ouvertes par cette étude.



(a) Fonction de valeur et politique en $p = 3$ lorsqu'aucun objectif n'a encore été rempli



(b) Politique en $p = 3$ lorsqu'aucun objectif n'a encore été rempli — vue 2D

FIG. 7.5 – Structure de la politique pour $p = 3$ pour le problème du rover

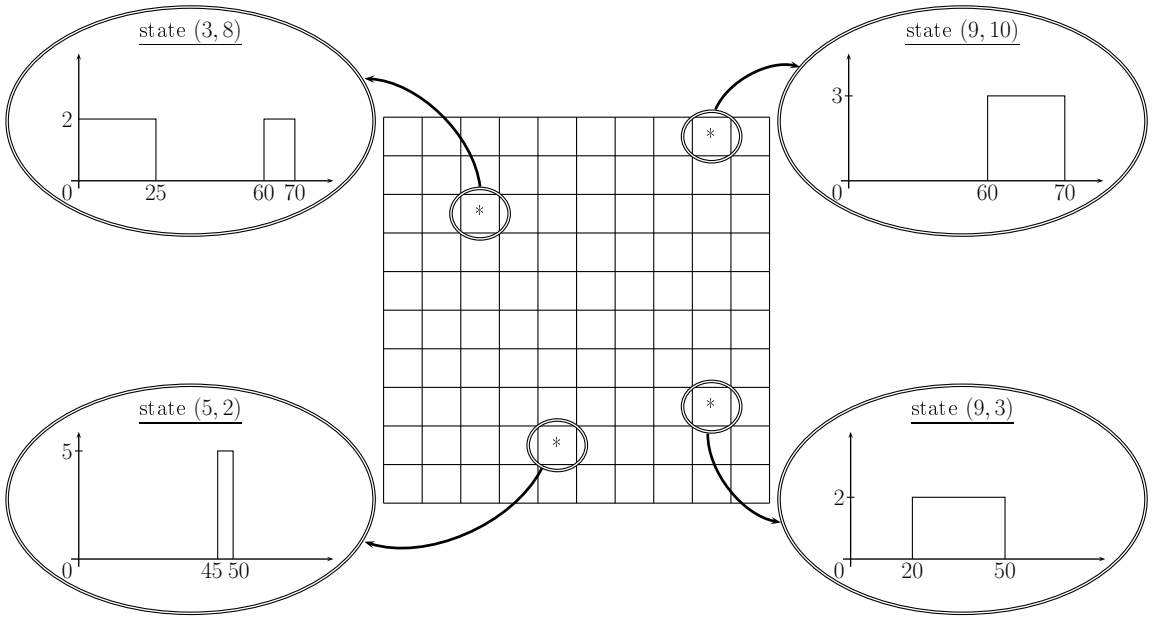


FIG. 7.6 – Problème de patrouille de l'UAV — Illustration et modèle de récompense

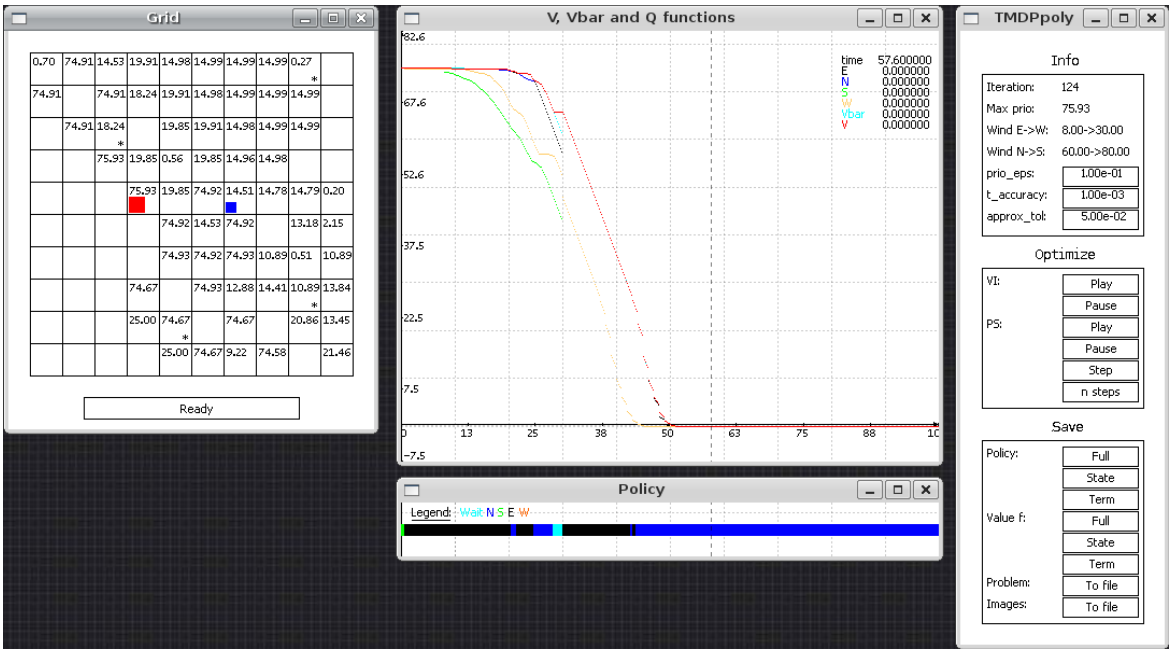


FIG. 7.7 – Problème de patrouille de l'UAV — Interface graphique

8

Généraliser les MDPs à temps continu observable : le cadre XMDP

Inclure un temps observable continu dans l'espace d'états d'un MDP amène naturellement l'introduction de l'action $attendre(\tau)$ dans l'espace des actions discrètes. Plus généralement, introduire des variables continues amène souvent la question des actions continues ou hybrides. Nous avons toutefois vu dans les chapitres d'introduction que la variable temporelle tenait une place particulière dans le cadre du critère γ-pondéré. Dans le présent chapitre, nous étendons le cadre MDP classique afin d'y rendre la variable temporelle observable. Nous nous plaçons alors dans le cadre d'un temps et d'un espace d'états continus et des actions paramétriques associées. Notre objectif est d'étudier les hypothèses de fond qui permettent de garantir l'existence et la validité d'une équation d'optimalité similaire à l'équation de Bellman.

8.1 Prise de recul sur le modèle TMDP : qu'est-ce que l'action "attendre" ?

Souvent, nos espaces d'action discrétisés le sont par commodité, afin de simplifier le traitement des problèmes associés. Cependant, ce niveau de discrétisation n'est pas toujours suffisant et ces représentations ont leurs limites. Par exemple, une action "avancer" n'est pas universelle et considérer qu'elle n'avance que d'une distance unitaire n'est valide que lorsqu'on considère un espace d'états discrétisé de la même manière. Si l'on considère des variables continues dans l'état, il devient naturel de considérer aussi les actions continues correspondantes.

L'action "attendre" du cadre TMDP se place en grande partie dans cette réflexion : il s'agit d'un action continue paramétrique de paramètre τ pour laquelle l'équation 5.1 permet de trouver la meilleure valeur de τ. La généralisation de cette considération à plusieurs actions paramétriques nous permet d'introduire le cadre XMDP.

8.2 Un modèle formel à temps observable continu et espaces d'états et d'actions hybrides

Nous définissons ainsi un XMDP de la façon suivante :

Définition (XMDP)**.** *Un XMDP est un quadruplet* $\langle S, A(X), p, r\rangle$ *où :*

S *est un espace d'états constitué de la tribu des Boréliens définie sur l'espace des variables d'état. Cet espace peut-être continu, discret ou hybride. Il inclut notamment l'horloge du processus.*
A *est un espace d'action décrivant un ensemble finie d'actions* $a_i(x)$ *où* x *est un vecteur de paramètres prenant ses valeurs dans* X*. Ainsi l'espace d'actions peut également être hybride ; il est factorisé par des actions* a_i *de nature différente. En pratique, chaque action ne dépend généralement que d'un sous-ensemble des variables de* X*.*
p *est la densité de probabilité du modèle de transition* $p(s'|s, a(x))$*.*
r *est un modèle de récompense* $r(s, a(x))$*.*

Nous pouvons alors définir naturellement des politiques XMDP et étendre la définition d'un critère γ-pondéré.

Nous effectuons par ailleurs les hypothèses suivantes sur le modèle :
– $|r((s,t), a(x))|$ est borné par M,
– $\forall \delta \in T, \quad t_{\delta+1} - t_\delta \geq \alpha > 0$, où α est la plus petite durée d'action possible[1]
– $\gamma < 1$.

8.3 Equation de Bellman étendue

On peut alors définir un opérateur d'évaluation des politiques L^π et un opérateur de Bellman étendu L :

Définition (Opérateur L^π)**.** *L'opérateur d'évaluation des politiques* L^π *associe, à chaque fonction* V *de* $\mathcal{V}$*, la fonction de valeur :*

$$L^\pi V(s,t) = r(s,t,\pi(s,t)) + \int\limits_{\substack{t' \in \mathbb{R} \\ s' \in S}} \gamma^{t'-t} p(s',t'|s,t,\pi(s,t)) V(s',t') ds' dt' \tag{8.1}$$

Définition (Opérateur L)**.** *L'opérateur (de Bellman) de programmation dynamique* L *associe, à toute fonction* V *de* $\mathcal{V}$*, la fonction de valeur :*$LV = \sup\limits_{\pi \in \mathcal{D}} \{L^\pi V\}$

$$LV(s,t) = \sup_{\pi \in \mathcal{D}} \left\{ r_\pi(s,t) + \int\limits_{\substack{t' \in \mathbb{R} \\ s' \in S}} \gamma^{t'-t} p_\pi(s',t'|s,t) V(s',t') ds' dt' \right\} \tag{8.2}$$

Et nous montrons que :

Proposition (Evaluation des politiques)**.** *Soit* π *une politique déterministe Markovienne.* $V = V^\pi$ *est l'unique solution de l'équation* $L^\pi V = V$*.*

Proposition (Equation d'optimalité)**.** *Pour un XMDP à critère* γ*-pondéré, la fonction de valeur optimale* V^* *est l'unique solution de l'équation d'optimalité* $V = LV$*.*

[1] les preuves qui suivent illustreront le fait qu'on peut en fait relâcher cette hypothèse pour ne conserver que la contrainte plus souple qui implose qu'il y ait une probabilité nulle d'un nombre infini de transition instantanées successives.

On peut alors proposer une formulation “paramétrique” de cette équation d'optimalité :

$$LV(s,t) = \max_{a \in A} \sup_{x \in X} \left\{ r(s,t,a(x)) \; + \int_{\substack{t' \in \mathbb{R} \\ s' \in S}} \gamma^{t'-t} p(s',t'|s,t,a(x)) V(s',t') ds' dt' \right\} \tag{8.3}$$

Cette formulation s'inspire du cadre TMDP et alterne optimisation des paramètres d'action et choix des actions elles-mêmes.

8.4 Retour au cadre TMDP

Nous pouvons alors montrer qu'un TMDP est exactement un XMDP à critère total et que les équations 5.1 à 5.4 sont équivalentes — sous les hypothèses rendues explicites du modèle TMDP — à l'équation d'optimalité 8.3.

8.5 Conclusion sur le cadre XMDP

L'objectif de l'introduction du cadre XMDP n'est pas de présenter *encore* un nouveau cadre de résolution inspiré du cadre MDP. A l'inverse, nous souhaitions fournir un modèle générique, adaptable aux hypothèses des différentes formulations existantes en termes d'espaces d'états et d'action hybrides, qui permette de garantir l'existence et la cohérence d'une équation de Bellman avec temps continu observable. Ce cadre XMDP peut être vu comme une généralisation du modèle MDP Borélien, du cadre TMDP ou de bien d'autres modèles.

9

Perspectives : discrétisation évolutive de la droite temporelle

La principale difficulté rencontrée lors des calculs analytiques de V^* dans le cas $TMDP_{poly}$ — outre les difficultés purement calculatoires et informatiques — provient du grand nombre d'intervalles de définition nécessaires à la description de la fonction de valeur. Lorsque l'on compare ce nombre à celui nécessaire à la description de la politique, on peut imaginer une autre approche qui ne nécessiterait pas un partitionnement aussi fin de la droite temporelle. La méthode que nous présentons dans ce chapitre s'appuie une intuition simple : le problème crucial est d'identifier les bornes des intervalles de définition temporels de la politique. Toutefois, trouver ces bornes et trouver les actions optimales à effectuer entre ces bornes correspondent à un unique processus d'optimisation. Nous essayons alors de trouver les valeurs des variables de décision (bornes et actions) en résolvant une séquence de problèmes discrets et en effectuant une évolution incrémentale des bornes temporelles locales.

Ce chapitre est appelé "perspectives" car il décrit un travail inachevé qu'il nous a cependant paru intéressant de présenter car il établit le lien entre différentes idées. Les idées développées ici sont liées à la fois à la problématique des chapitres précédents sur la résolution des MDP à temps continu observable et aux première intuitions d'itération de la politique approchée qui seront développées dans la suite du manuscrit. Assez ironiquement, ces idées d'évolution incrémentales des bornes ont été introduites très tôt durant la thèse et ont engendré beaucoup des développements ici présentés. Bien que cette idée n'ait pas donné lieu à une implantation ni à une étude complète, elle constitue une abstraction intéressante du problème de la recherche des intervalles optimaux pour la résolution. Elle introduit également l'idée de recherche directe d'une politique, idée au coeur de la partie III de la thèse.

9.1 Définitions et idée générale

L'idée que nous développons ici est de rechercher, par état, le partitionnement de la ressource "temps" qui permette de décrire la politique optimale. Nous nous plaçons dans le cadre des notations SMDP+ en conservant l'hypothèse d'une action "attendre" déterministe. Nous définissons alors le concept d'intervalle de décision, correspondant à un intervalle temporel durant lequel la politique reste la même.

L'objectif de notre approche est alors de considérer ces intervalles de décision comme des variables de décision au même titre que les actions et de rechercher incrémentalement leur valeur optimale, état par état.

9.2 Evolution des intervalles de décision par résolution d'une séquence de problèmes discrets

Notons $\mathcal{T}_s$ le jeu d'intervalles de décision en s, correspondant à la dernière politique définie. Avec cet ensemble discret d'intervalles, nous pouvons considérer l'espace d'états abstrait :

$$\tilde{\Sigma} = \{(s, T) / s \in S, T \in \mathcal{T}_s\}$$

Supposons à présent que nous démarrions notre recherche avec une première estimation du partitionnement de la ressource temporelle, par état. Notre idée est alors de procéder en quatre étapes :

- **Discrétisation.** Avant tout, nous calculons les modèles de transition et de récompense d'un MDP discret $\tilde{M}$, défini sur l'espace d'états $\tilde{\Sigma}$ et approchant le comportement du problème M original.
- **Recherche d'actions optimales.** Nous calculons alors, en faisant appel au méthodes de résolution de MDP de la littérature, la politique optimale pour le problème $\tilde{M}$. Notons $\tilde{\pi}$ cette politique. Nous fusionnons alors toute paire d'intervalles consécutifs de $\mathcal{T}_s$ pour lesquels l'action optimale reste la même.
- **Evaluation de la politique.** Puis nous évaluons $\tilde{\pi}$ dans le modèle continu en définissant la politique π correspondant à $\tilde{\pi}$.
- **Evolution des bornes des intervalles de décision.** Enfin, nous utilisons la fonction de valeur obtenue à la dernière étape afin d'effectuer une unique itération de Bellman, fournissant ainsi, par état, une date à laquelle nous pouvons améliorer la politique actuelle. Nous passons pour cela par la définition de la t-erreur de Bellman [Rachelson et al., 2006]. Nous insérons alors cette date dans le jeu d'intervalles $\mathcal{T}_s$.

On peut voir une telle approche comme un algorithme proche de l'itération de la politique, où la phase d'évaluation correspond à une évaluation approchée fondée sur un modèle discret optimiste du problème.

10

Conclusion

Ce chapitre récapitule les résultats et contributions présentés dans les chapitres précédents. Nous discutons notamment de la possibilité d'étendre l'algorithme $TMDP_{poly}$ au cas plus général des XMDP hybrides, en essayant de mettre en évidence les avantages et les difficultés inhérentes à cette approche. Enfin, nous concluons sur cette partie de la thèse et présentons le lien avec la partie suivante.

10.1 Messages "à emporter"

Cette première grande partie de la thèse s'est concentrée autour de l'introduction d'un temps observable dans le cadre MDP. Ce problème a soulevé les questions du lien avec le critère γ-pondéré, du cadre algorithmique de résolution et de la représentation formelle des problèmes décisionnels de Markov temporels. Voici un bref résumé des conclusions tirées des chapitres précédents :

- Les fait de considérer une variable temporelle continue et observable implique le traitement de MDP à espaces d'états hybrides. De plus, ce temps observable affecte directement l'expression du critère γ-pondéré.
- L'introduction de variables continues comme le temps mène souvent à l'introduction d'actions continues comme "attendre".
- Le cadre XMDP formalise ces caractéristiques des problèmes temporels et établit une équation d'optimalité pour les politiques correspondantes. Ce cadre XMDP englobe les formalismes MDP, SMDP+ et TMDP.
- En pratique, lorsque le temps est l'unique variables continue et qu'"attendre" est l'unique action continue, certaines hypothèses supplémentaires peuvent être faites. Notamment, "attendre" peut être considérée déterministe vis-à-vis des variables d'état et la récompense associée à une durée d'attente nulle est zéro. Ces aspects ont été illustrés par les liens entre TMDP et SMDP+.
- Les équations d'optimalité présentées dans [Boyan and Littman, 2001] pour le cadre TMDP correspondent à un critère total pour le XMDP équivalent.
- Les tentatives d'extension de la résolution exacte des TMDP au cas des fonctions polynômiales par morceaux se heurte au problème des manipulations formelles sur de telles représentations. En pratique, la résolution exacte n'a pu être étendue au delà des densités de probabilité discrètes, des modèles de transition constants par morceaux et des modèles de récompense de degré inférieur à 5.
- L'analyse des équations d'optimalité TMDP a toutefois permis de définir une méthode de résolution approchée plus globale dans le cadre des fonctions polynômiales par

morceaux en général. Cette approche se fonde sur :
  - Le calcul exact ou approché des manipulations de coefficients pour les fonctions polynômiales par morceaux.
  - L'adaptation de l'algorithme de Prioritized Sweeping aux TMDP.
  - Les résultats de convergence de l'itération de la valeur approchée.

  Ces éléments ont permis de construire l'algorithme et le planificateur $TMDP_{poly}$
- La principale faiblesse des méthodes d'itération de la valeur pour les problèmes décisionnels de Markov temporels provient de la difficulté à définir avec exactitude les fonctions de valeur. Dans le cas des représentations polynômiales par morceaux, ceci se traduit par l'augmentation du nombre d'intervalles de définition nécessaires à la description exacte de la fonction de valeur. Nous avons, dans ce cadre, proposé une approche de simplification de cette représentation via un partitionnement évolutif de la droite temporelle. Ceci a permis de définir un méthode type itération de la politique qui présente les premières idées des algorithmes se passant de modèle présentés dans la partie suivante de la thèse.

10.2 Perspectives

En plus des perspectives concernant la discrétisation évolutive du temps présentée au chapitre 9, il est intéressant de s'intéresser à la façon dont les idées développées pour $TMDP_{poly}$ peuvent s'appliquer à des classes de MDP plus générales. En particulier, on peut se poser la question d'adapter l'algorithme $TMDP_{poly}$ aux XMDP.

Bien que de nombreuses briques de base semblent disponibles, assembler un algorithme complet ne semble pas évident. Il y a plusieurs raisons à cela. D'une part, il est à noter que peu de méthodes ont été développées dans la littérature afin d'effectuer des itérations de Bellman formelles comme nous l'avons fait dans le cadre TMDP. Généralement, l'option retenue pour les résolutions similaires est de résoudre un problème linéaire cherchant à projeter V_{n+1} sur un jeu approprié de fonctions de base, comme pour les approches ALP ou LSPI. L'approche initiale de Neuro-Dynamic Programming utilise des réseaux de neurones comme base de représentation commune des fonctions de valeur, d'autres approches récentes utilisent des opérateurs de régression ou d'estimation plus ou moins sophistiqués mais — à notre connaissance — dans la plupart des cas, le problème se ramène à un problème d'apprentissage supervisé. La recherche directe d'une famille de fonctions stable par l'opérateur de Bellman est un problème difficile et a fourni peu de résultats jusqu'à présent.

Lorsqu'on introduit plusieurs variables continues en plus des variables discrètes dans un problème MDP, trouver une bonne représentation devient de plus en plus ardu. Le cas des fonctions constantes ou linéaires par morceaux a été exploité dans les travaux de [Feng et al., 2004], [Li and Littman, 2005] ou encore [Benazera et al., 2005]. Nous pouvons aussi nous référer à une méthode alternative intéressante présentée dans [Marecki et al., 2006]. Une implantation $XMDP_{poly}$ pourrait probablement utiliser des fonctions de valeur constantes ou linéaires par morceaux, associées à des états discrets comme dans [Benazera et al., 2005].

Toutefois, au delà de ces premiers obstacles, la principale difficulté provient de la définition des actions continues ou hybrides. Dans le cas des TMDP, l'optimisation exploite le caractère déterministe de l'action "attendre" et le fait qu'elle soit la seule action continue. Ceci permet notamment le découplage des équations d'optimalité. Dans le cas général, il faut résoudre le problème posé par la formulation paramétrique de l'équation de Bellman introduit par l'équation 8.3. En d'autres termes, il faut trouver un moyen efficace de trouver les paramètres

d'action avant de comparer les actions ensemble.

Ici encore, une bonne représentation de la fonction de valeur permet de faciliter la recherche de ces paramètres optimaux. Les procédures d'éliminations d'action, comme présentées dans [Puterman, 1994] ou utilisées dans [Mausam and Weld, 2006], permettent également de réduire la quantité de calculs nécessaires à la prise en compte de ces actions hybrides.

Bien que ces approches ne soient pas directement liées à l'optimisation de politiques sur des MDP de modèle connu, il est toutefois important de mentionner les travaux de [Hasselt and Wiering, 2007] ou [Antos et al., 2007] au sujet des espaces d'actions continus.

Enfin, l'aspect critique qui rend efficaces la plupart des planificateurs MDP actuels concerne leur stratégie de recherche. Bien que Prioritized Sweeping constitue une procédure efficace d'ordonnancement des itérations de Bellman pour la résolution complète de problèmes MDP, la recherche heuristique permet d'obtenir un important gain d'efficacité pour la résolution partielle de tels problèmes. Ainsi, selon le type de problèmes XMDP que l'on souhaite résoudre, in planificateur XMDP n'effectuera pas nécessairement ses étapes de programmation dynamique de la même manière que l'algorithme $TMDP_{poly}$.

Pour résumer ces différentes idées, les XMDP et $TMDP_{poly}$ ouvrent la voie à une classe plus générale de méthodes pour les MDP à espaces d'états et d'actions hybrides, cependant :

- La recherche d'un bon cadre de représentation pour la fonction de valeur demeure un problème difficile pour lequel les représentations polynômiales présentent clairement des limites.
- Le calcul formel des itérations de Bellman peut se rendre utile dans le cadre de la formulation paramétrique de l'équation de programmation dynamique.
- Ces problèmes souffrent toutefois toujours du *curse of dimensionality*. Cela implique de mettre en oeuvre des méthodes efficaces de sélection d'actions, de recherche heuristique et d'approximation afin de construire un algorithme pertinent.

10.3 Ouvertures

En considérant comment nos travaux peuvent s'appliquer ou s'adapter au cas général des XMDP, nous avons déjà pris un certain recul vis-à-vis du problème académique de la résolution des TMDP. Eloignons-nous encore un peu afin de considérer la façon dont notre problème a été posé en premier lieu.

Depuis le début de cette partie, nous avons considéré comme acquis qu'un modèle était fourni et que nous pouvions l'approcher en utilisant par exemple des distributions discrètes ou polynômiales par morceaux et des fonctions polynômiales par morceaux. Toutefois, en pratique, les difficultés rencontrées lors du traitement des problèmes que nous avons cherchés à résoudre ne concernent pas uniquement la construction d'algorithmes de résolution efficaces : elles nécessitent en particulier un grand travail de formalisation et d'écriture du problème en premier lieu.

Comme dans le cas de la plupart des problèmes MDP, écrire les matrices de transition ou les densités de probabilité constitue une tâche nécessitant un lourd travail d'ingénierie. De nombreux problèmes que nous cherchons à résoudre ne sont pas aisément décrits par des distributions de probabilités explicitement fournies simplement parce que le fait de trouver la forme exacte de telles distribution est difficile en premier lieu. Prenons l'exemple du métro introduit au chapitre 2. Les variables d'état sont alors le nombre de passagers dans les

stations et les trains, la position courante de chaque train et la variables temporelle. Etant donné l'état courant, écrire la densité de probabilité sur l'état suivant demande de grandes ressources de calcul tandis que la spécification d'un simulateur pour un tel problème est une tâche bien plus simple.

Cet exemple simple illustre le besoin de s'attacher à spécifier des représentations simples et efficaces pour la formalisation de processus temporels stochastiques complexes. Une telle spécification permettra alors de décrire le comportement de tels systèmes et de construire des simulateurs.

L'optimisation de politiques de contrôle, sans l'usage d'un modèle explicite du processus, en exploitant les signaux de récompense issus de l'environnement, constitue le domaine d'étude de l'Apprentissage par Renforcement. La seconde partie de la thèse se concentre sur l'idée de représenter la complexité des problèmes décisionnels de Markov temporels sous forme de simulateurs afin de construire des modèles génératifs du processus à contrôler. Ces modèles sont alors utilisés avec une approche d'itération de la politique approchée, fondée sur la simulation du processus.

Troisième partie

Contrôler des systèmes stochastiques dépendant du temps en présence d'événements exogènes incontrôlables

11

La concurrence d'événements, origine de la complexité du processus

De nombreux problèmes temporels présentent une structure complexe, notamment en regard de la difficulté d'écrire le modèle en premier lieu. Un tel modèle nécessite généralement un très gros travail d'ingénierie en amont, souvent aussi difficile que la résolution du problème lui-même. Ce chapitre s'intéresse à l'une des principales causes de complexité des processus temporels : la concurrence de phénomènes concurrents. Une représentation efficace et compacte de la concurrence dans les processus stochastiques semble être une des clés à l'écriture et au traitement de problèmes fortement couplés et de grande taille. Le cadre des processus semi-Markoviens généralisés (GSMP) capture élégamment la complexité du processus temporel global. Nous passons en revue les propriétés des GSMP puis nous intéressons plus précisément à la question de la modélisation de tels processus stochastiques dans le cadre de la théorie de spécification des systèmes à événements discrets DEVS. Nous introduisons alors un possible choix d'actions dans le modèle GSMP afin de parvenir à un problème de décision dans l'incertain avec événements exogènes et temps continu observable.

11.1 La difficulté d'écrire un modèle décisionnel stochastique temporel

La difficulté d'écriture associée aux problèmes présentés en exemples et aux problèmes temporels en général est une conséquence directe de la difficulté à synthétiser les contributions concurrentes de plusieurs processus stochastiques couplés par un espace d'états commun. Ainsi, une description à événements explicites du système doit représenter à la fois les processus locaux mais aussi leur couplage via l'espace d'états.

11.2 Les processus décisionnels semi-markoviens généralisés

Un GSMP correspond à une collection de SMP affectant tous le même ensemble de variables. On peut décrire un GSMP comme un ensemble d'états et un ensemble d'événements concurrents. A chaque événement correspond une horloge qui indique la durée avant déclenchement de l'événement. Lorsqu'un événement se déclenche, il emmène le processus dans un nouvel état étant donné une probabilité de transition propre à cet événement. Certain événements sont alors désactivés, d'autres activés, et c'est de nouveau le premier événement à déclencher

qui conditionne la transition suivante comme illustré à la figure 11.1.

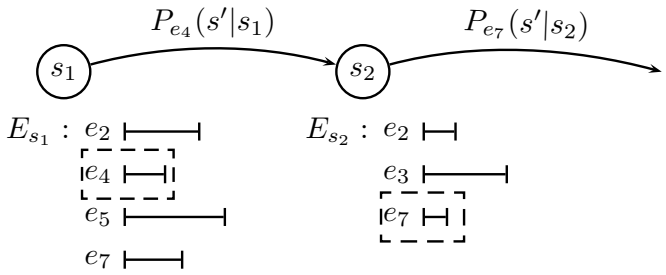


FIG. 11.1 – Illustration d'un GSMP

Il est important de noter que le processus décrivant l'évolution de l'état s du GSMP n'est pas un processus Markovien. En revanche, le processus décrivant l'évolution des variables (s, c) est, lui, semi-Markovien.

11.3 Modélisation DEVS

La théorie de spécification des systèmes à événements discrets DEVS permet de définir un cadre commun à la représentation et au couplage de tout système à événement discret. Ce modèle implique les éléments suivants :

Définition (Modèle atomique DEVS, [Zeigler, 1976])**.** *Un modèle atomique DEVS est décrit comme un octuplet* $\langle X, Y, S, \delta_{ext}, \delta_{int}, \delta_{con}, \lambda, ta \rangle$ *avec :*

- X*, un ensemble de ports d'entrée et leurs domaines de valeurs associés,*
- Y*, un ensemble de ports de sorties et de valeurs,*
- S*, un ensemble d'états internes,*
- $\delta_{ext} : S \times X \to S$*, une fonction de transition externe décrivant l'évolution de l'état interne du modèle lorsqu'un événement externe se produit sur l'un des ports d'entrée,*
- $\delta_{int} : S \to S$*, une fonction de transition interne décrivant l'évolution naturelle de l'état interne du modèle,*
- $\delta_{con} : S \times X \to S$*, une fonction de résolution de conflits, spécifiant le comportement à tenir en cas de conflit de simultanéité entre un événement interne et externe (spécifiant généralement un choix entre* δ_{int} *et* δ_{ext}*),*
- $\lambda : S \to Y$*, une fonction de sortie, mettant à jour les valeurs des ports de sortie,*
- $ta : S \to \mathbb{R}^+$*, une fonction d'avancement du temps, utilisée afin de planifier la date de la prochaine transition interne.*

On peut adopter une représentation graphique à base de ports comme sur la figure 11.2.

Il est également possible de construire des modèles DEVS par couplage et par hiérarchisation de modèles comme illustré sur la figure 11.3.

11.4 GSMP et modèles DEVS

Ecrire un GSMP dans le cadre DEVS nécessite d'une part de surmonter un certain nombre d'obstacles dus à un recouvrement des éléments de vocabulaire, puis à décider du niveau de

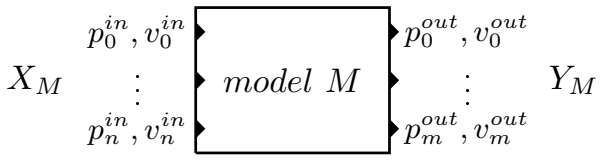


FIG. 11.2 – Modèle atomique DEVS à ports

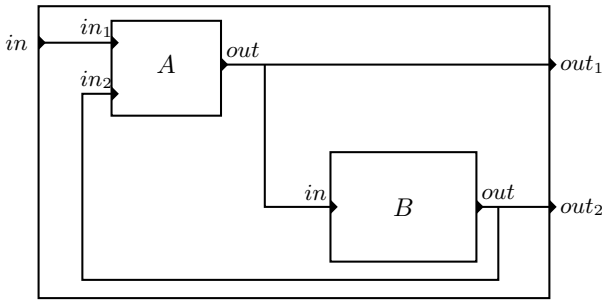


FIG. 11.3 – Modèle DEVS couplé

finesse de la représentation. Choisir de construire un modèle DEVS par événement GSMP semble une option naturelle. Une telle représentation fournit des modèles tels celui présenté figure 11.4(b) et nécessitant un modèle "observateur" permettant le couplage via l'espace d'état.

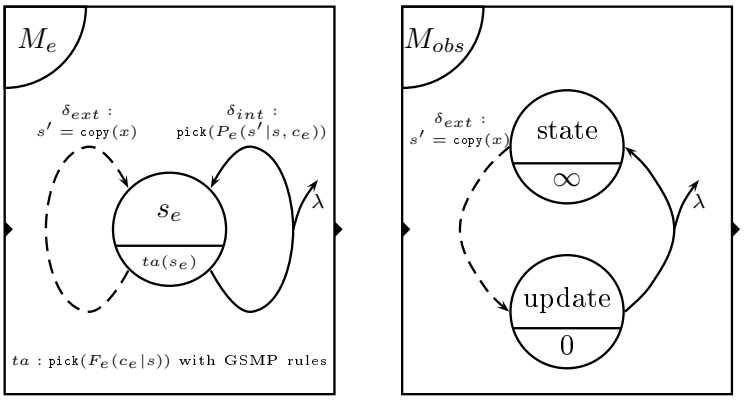
Ces modèles peuvent alors être couplés comme indiqué figure 11.5.

Il est possible de se passer d'un modèle observateur lors de l'écriture du modèle global, cependant cela implique un réseau très dense de communications entre modèles afin d'assurer la synchronisation des variables d'états entre modèles.

Cette analyse permet de mieux comprendre la complexité d'établir une loi d'évolution du processus global d'un GSMP. D'autre part, elle permet d'établir que pour écrire un patron de modèle GSMP pour un moteur de simulation DEVS, il peut être préférable de choisir une solution "tout intégré" en un unique modèle atomique, vu comme une boîte noire, qu'un modèle éclaté et distribué en autant de modèles atomiques que d'événements GSMP. C'est sur la base de cette analyse que nous avons implanté l'extension aux GSMP de la plate-forme de simulation VLE [Quesnel et al., 2007]. Cela permet enfin de mieux comprendre la complexité de modélisation et de résolution des processus stochastiques décisionnels temporels comme nous le montrons ci-dessous.

11.5 MDP, temps continu et concurrence d'événements

Un GSMDP correspond à un GSMP où l'on distingue un sous-ensemble d'événements particuliers, les appelant contrôlables et permettant à un agent de choisir lequel activer à chaque changement d'état. On peut alors définir un tel ensemble état par état et appeler cet



(a) Modèle graphique DEVS d'un événement GSMP

(b) Modèle graphique DEVS de l'observateur d'état GSMP

FIG. 11.4 – Modèles atomiques DEVS pour la description des GSMP

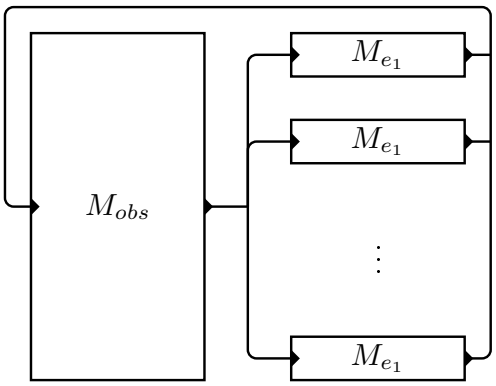


FIG. 11.5 – Modèle DEVS couplé pour les GSMPs

ensemble “espace d’actions”. Cet ensemble n’est jamais vide puisqu’il contient au minimum une action “ne rien faire” qui laisse simplement les événements exogènes contrôler le système pour un pas. Une telle action correspond à un événement ayant une horloge de valeur $+\infty$ en permanence.

On peut définir la notion d’un chemin d’exécution ρ pour un GSMDP donné et y associer une évaluation via un critère γ-pondéré :

$$V_\gamma^\pi(\rho) = \sum_{i=0}^{n-1} \gamma^{T_i} \left(\gamma^{t_i} k(s_i, e_i, s_{i+1}) + \int_0^{t_i} \gamma^t c(s_i, e_i) dt \right) \tag{11.1}$$

Cela permet alors d’établir formellement la fonction de valeur d’une politique GSMDP :

$$V_\gamma^\pi(s) = E_s^\pi \left[V_\gamma^\pi(\rho) \right] \tag{11.2}$$

Toutefois, le problème de contrôle d’un GSMDP présente bien moins de garanties que celui d’un MDP. En effet, parce que le processus d’évolution des variables d’état s est non-Markovien, il n’est pas possible de garantir l’existence d’une politique optimale Markovienne. Par ailleurs, l’augmentation de l’ensemble des variables d’états par les horloges des événements permettrait de récupérer la propriété de Markov pour le modèle de transition et donc la garantie d’une politique optimale Markovienne. Cependant ces horloges sont rarement observables en pratique et une politique dépendant d’elles n’aurait donc que peu de sens physique. Nous discutons de plusieurs autres manières de compléter cet état, dit “naturel”, afin de récupérer la propriété de Markov sur l’espace d’états augmenté.

Au final, nous optons pour une solution ne permettant pas de garantir l’optimalité d’une politique Markovienne. Il s’agit d’inclure le temps comme variable d’état dans un GSMDP classique afin de pallier à l’inobservabilité des horloges et d’éviter de stocker les chemins d’exécution complets. La validité d’une équation de Bellman dans ce cas avec temps observable est garantie par les développements du modèle XMDP.

11.6 Conclusion

De ce chapitre dédié à la modélisation des problèmes temporels de Markov, on peut retenir les choses suivantes. Le parallèle fait avec la théorie des systèmes à événements discrets permet d’illustrer pourquoi les problèmes temporels de Markov s’écrivent difficilement dans un cadre à événements implicites comme les MDP. Les GSMDP en revanche constituent un modèle élégant et efficace pour les représenter, ceci étant obtenu au prix de la perte de la propriété de Markov pour le processus de l’état naturel du processus. Enfin, nous faisons l’hypothèse que l’introduction d’un temps observable dans le modèle permettra de compenser — au moins pour partie — la perte d’optimalité associée à la recherche d’une politique Markovienne dans le cadre GSMDP. Un modèle synthétique et se prêtant à une résolution par programmation dynamique n’étant pas disponible, il faut chercher une autre façon d’exploiter les propriétés des GSMDP. Ces derniers, s’ils sont complexes à ramener à un MDP, sont en revanche relativement faciles à simuler. Nous nous tournons ainsi vers des approches fondées sur l’exploitation des simulations de GSMDP pour essayer d’incrémentalement améliorer les politiques Markoviennes que nous y définissons.

12

L'algorithme Real-Time Policy Iteration

L'idée d'effectuer directement une recherche de politique, en utilisant les éléments de simulation introduits au chapitre précédent afin d'effectuer une évaluation type "Monte-Carlo", se rapporte aux algorithmes Itération de la Politique. Nous laissons de côté le cadre GSMDP le temps d'un chapitre afin d'introduire un algorithme qui présente les propriétés essentielles de la méthodes que nous développerons dans les chapitres suivants. Ainsi qu'on l'a vu au travers de l'exemple de Prioritized Sweeping, la Programmation Dynamique devient particulièrement efficace lorsqu'elle est intelligemment guidée dans son exploration de l'espace d'états. Real-Time Dynamic Programming (RTDP, [Barto et al., 1995]) désigne une des familles d'algorithmes de résolution de MDP les plus efficaces. Cet algorithme s'appuie sur une exploration heuristique et sur des mises à jour asynchrones de la fonction de valeur. Dans ce chapitre, nous reprenons les bases de la programmation dynamique asynchrone et introduisons une version de RTDP inspirée de l'Itération de la Politique, que nous nommons RTPI. Tandis qu'RTDP et ses variantes sont touts basés sur l'itération de la valeur asynchrone, RTPI effectue une recherche directement dans l'espace des politiques.

L'intuition que nous développons dans ce chapitre — très proche de la philosophie d'RTDP — peut-être formulée ainsi : pour un algorithme de recherche incrémentale de politique, étant donné un état initial ou un ensemble d'états initiaux, afin d'obtenir des améliorations rapides de la politique, les états à mettre à jour vis-à-vis de la politique sont les états fréquemment rencontrés lors de l'exécution de la politique courante, et donc les états visités par la simulation de cette politique.

La progression des idées dans ce chapitre peut se résumer comme suit. Nous commençons (section 12.1) par reprendre l'idée de mises à jour de Bellman asynchrones et nous concentrons en particulier sur l'itération de la politique. Ceci nous mène à discuter de la question de l'approximation en itération de la politique (section 12.2). Puis nous revenons à notre sujet principal : nous présentons la méthode d'exploration gloutonne d'RTDP à la section 12.3 et montrons comment l'on peut adapter cet algorithme à un cadre fondé sur la simulation, avec l'algorithme RTPI de la section 12.4.

12.1 Programmation Dynamique Asynchrone

L'idée d'origine de la programmation dynamique asynchrone repose sur le fait que l'on peut effectuer les mises à jour de Bellman dans n'importe quel ordre : tant que chaque état est mis à jour une infinité de fois lorsque le nombre d'itérations tend vers $+\infty$, la politique converge vers la politique optimale.

[Bertsekas and Tsitsiklis, 1996] pose les bases de l'itération de la politique asynchrone de la façon suivante. A l'itération n, on sélectionne un sous-ensemble S_n de S et on effectue les mises à jour de Bellman uniquement sur S_n. Cela fournit la politique π_{n+1} telle que :

$$\pi_{n+1}(s) = \begin{cases} \underset{a \in A}{\operatorname{argmax}}\, r(s,a) + \gamma \sum_{s \in S} P(s'|s,a) V^{\pi_n}(s') & \text{if } s \in S_n \\ \pi_n(s) & \text{if } s \notin S_n \end{cases} \tag{12.1}$$

On peut, de manière similaire, effectuer un certain nombre de mises à jour $V_{k+1} = L^{\pi_n} V_k$ sur les différents états de S_n.

Cette formulation permet de regrouper en une seule description tous les algorithmes précédents : si l'on alterne mise à jour de la politique et mise à jour de la fonction de valeur sur tout l'espace d'états alors cet algorithme est équivalent à l'itération de la valeur. De façon similaire, si le nombre de mises à jour de la fonction de valeur n'est pas borné, on retrouve l'itération de la politique. Enfin, si on alterne une mise à jour de la politique et m_n mises à jour de la fonction de valeur, on trouve l'algorithme d'itération de la politique modifié.

Si la fonction de valeur et a politique initiales vérifient $V_0 \leq L^{\pi_0} V_0$ et si les mises à jour sont effectuées un nombre de fois tendant vers l'infini lorsque n tend vers l'infini, alors [Bertsekas and Tsitsiklis, 1996] montre que la politique converge vers la politique optimale.

12.2 L'approximation pour l'algorithme d'Itération de la Politique

Le défaut majeur des approches d'itération de la politique réside dans la phase d'évaluation qui grève fortement le temps d'exécution. Il est alors commun de remplacer cette phase exacte par une évaluation approchée de la valeur de la politique courante. [Anderson, 2000] présente une analyse empirique de la différence de robustesse entre les méthodes approchées d'itération de la politique et de la valeur, illustrant que les premières sont moins enclines à osciller autour d'une solution optimale.

Il est établi que si l'on peut borner l'erreur d'approximation d'un opérateur Ap par :

$$\exists \epsilon \in \mathbb{R}^+ \ / \ \forall f \in \mathcal{F}(S, \mathbb{R}),\ \|Ap(f) - f\|_\infty \leq \epsilon \tag{12.2}$$

Alors l'itération de la politique approchée fournit une suite de politiques π_k telle que :

$$\limsup_{k \to \infty} \|V^* - V^{\pi_k}\| \leq \frac{2\gamma\epsilon}{(1-\gamma)^2} \tag{12.3}$$

Ceci permet de borner les erreurs faites lors d'une itération de la politique approchée. Sur la base de ce résultat, de nombreuses méthodes d'approximation ont été développées. Nous regroupons ces méthodes en trois grandes familles : les architectures d'approximation linéaires (ALP ou LSPI), les méthodes utilisant la simulation (roll-outs, méthodes Monte-Carlo), les méthodes de représentation structurées (arbres, méthodes à noyaux, méthodes évolutionnaires).

12.3 Recherche heuristique en avant pour l'itération de la valeur asynchrone

L'algorithme RTDP [Barto et al., 1995] est un algorithme d'itération de la valeur asynchrone dont le principe de base est d'effectuer une recherche en avant depuis un état de départ, cette recherche étant guidée par une fonction heuristique. Cette démarche est illustrée par l'algorithme 12.1.

Algorithm 12.1: Real Time Dynamic Programming

RTDP(state s, value function h)
$V(s) \leftarrow h(s)$
repeat RTDPtrial(s) **until** convergence of the value function

RTDPtrial(state s)
while $s \notin GoalStates$ **do**
  $a \leftarrow \underset{a \in A}{\operatorname{argmax}}\, r(s,a) + \sum_{s' \in S} P(s'|s,a) \cdot s'.\text{value}$
  $s.\text{value} \leftarrow s.\text{Qvalue}(a)$
  $s \leftarrow \text{pickNextState}(s,a)$

Avec l'algorithme lRTDP [Bonet and Geffner, 2003b], cette recherche en avant est complétée par un mécanisme de mise à jour "en arrière". Ce mécanisme met à jour les parents des états visités lors du dernier épisode, permettant ainsi d'accélérer la convergence de RTDP. Cette approche de recherche en avant et en arrière a donné naissance aux méthodes actuelles les plus performantes en planification dans l'incertain, notamment lors de la dernière compétition internationale de planification [Teichteil-Königsbuch and Infantes, 2008].

12.4 L'algorithme Real Time Policy Iteration

L'idée que nous développons ici est d'adapter le guidage par simulation d'une politique gloutonne adopté par RTDP, au cadre de l'itération de la politique. Une telle approche se fonde sur les précédents résultats d'itération de la politique asynchrone et approchée.

Si nous supposons disposer d'un oracle permettant d'obtenir facilement la mise à jour de la fonction de la valeur (représenté sous la forme de la fonction updateQvalues), alors l'algorithme RTPI peut-être présenté tel qu'à l'algorithme 12.2.

Algorithm 12.2: Real-Time Policy Iteration

RTPI(state s, policy π)
repeat RTPItrial(s) **until** convergence of the policy

RTPItrial(state s)
while $s \notin GoalStates$ **do**
  s.updateQvalues()
  $\pi(s) \leftarrow \underset{a \in A}{\operatorname{argmax}}\, s.\text{Qvalue}(a)$
  $s \leftarrow \text{pickNextState}(s, \pi(s))$

Le calcul des Q-valeurs en pratique peut se faire de nombreuses manières, de façon exacte ou approchée comme mentionné plus haut. Il est toutefois particulièrement intéressant de noter que pour les problèmes à temps explicites, une évaluation de π_n reste utilisable lors d'un épisode de RTPI jusqu'à ce qu'on ait atteint l'horizon, cela même si certaines actions de

la politique sont mises à jour au fur et à mesure. Cette propriété tient au principe de causalité et à la stratégie d'exploration de l'espace d'états : si l'on met à jour la politique en l'état s^1 et si l'on choisit s^2 via l'application d'une politique gloutonne, alors s^2 est nécessairement postérieur à s^1 et sa valeur ne dépend pas de l'action décidée en s^1.

12.5 Conclusion

Ce cadre RTPI constitue la base de l'algorithme dédié aux problèmes décisionnels de Markov temporels que nous allons développer au prochain chapitre. Cet algorithme repose sur l'idée de recherche en avant pour RTPI, associée à une évaluation par simulation et à l'utilisation d'un couple politique - fonction de valeur.

13

Recherche locale et incrémentale de politiques, fondée sur la simulation, pour les GSMDP à temps observable : l'algorithme ATPI

La résolution des problèmes temporels de Markov en grande dimension pose plusieurs difficultés importantes. La première concerne l'écriture du problème en premier lieu. Le chapitre 11 a permis d'analyser la structure sous-jacente à la complexité de tels processus temporels. Ceci a illustré le fait que ces processus n'étaient pas nécessairement Markoviens, qu'ils avaient généralement un grand nombre de variables dans l'espace d'états et étaient facilement représentables sous la forme de processus atomiques concurrents. Toutefois, le couplage entre ces processus concurrents demeure au coeur de la complexité du problème global. De ce fait, ces processus sont simples à simuler et à décrire via un modèle génératif mais difficiles à intégrer sous forme d'un modèle prédictif à événements implicites.
En se fondant sur la formulation GSMDP à temps continu et sur l'idée de Real-Time Policy Iteration, nous concevons un algorithme d'itération de la politique approchée afin de chercher de bonnes politiques de contrôle. Cet algorithme s'appuie sur une exploration gloutonne basée sur la simulation, sur une évaluation de type rollout et sur des méthodes d'apprentissage statistique afin de construire les fonctions de valeur. Nous présentons les premiers résultats obtenus avec cet algorithme baptisé ATPI et illustrons son principal défaut, motivant ainsi la version améliorée du chapitre suivant.

13.1 Idée générale

Pour des GSMDP à temps observable et à état initial connu, nous effectuons des améliorations locales de la politique, dans les états rencontrés par simulation de la meilleure politique à jour. Puisque notre problème utilise un critère total, chaque trajectoire dans l'espace d'états jusqu'à l'horizon temporel fournit une réalisation de la variable aléatoire décrivant la récompense cumulée espérée, dans chacun des états rencontrés. Nous utilisons des outils généraux d'apprentissage statistique (régression) afin de généraliser cette information et de construire une fonction de valeur approchée pour la politique courante. Cette fonction de valeur est alors utilisée pour les itérations de Bellman lors de la trajectoire suivante.

En résumé :
- Nous effectuons une exploration partielle de l'espace d'états, guidée par la simulation.
- Nous stockons les récompenses obtenues comme des échantillons des variables aléatoires

$R^\pi(s)$ (récompense à venir).
- Nous utilisons ces échantillons pour construire une régression de la fonction de valeur de la politique correspondant à la dernière trajectoire.
- Nous utilisons cette fonction de valeur afin d'améliorer la politique lors de la trajectoire suivante.

13.2 L'algorithme d'Approximate Temporal Policy Iteration

L'algorithme 13.1 décrit les principales étapes de notre algorithme ATPI.

Algorithm 13.1: Online-ATPI

main :
input: π_0 or $\tilde{V}_0$, s_0
repeat
  $TrainingSet \leftarrow \emptyset$
  for $i = 1$ to N_{sim} **do**
    $\{(s, v)\} \leftarrow \texttt{simulate}(\tilde{V}, s_0)$
    $TrainingSet \leftarrow TrainingSet \cup \{(s, v)\}$
  $\tilde{V} \leftarrow \texttt{TrainApproximator}(TrainingSet)$
until termination

$\texttt{simulate}(\tilde{V}, s_0)$:
$ExecutionPath \leftarrow \emptyset$
$s \leftarrow s_0$
while horizon not reached **do**
  $a \leftarrow \texttt{ComputePolicy}(s, \tilde{V})$
  $(s', r) \leftarrow \texttt{GSMDPstep}(s, a)$
  $ExecutionPath \leftarrow ExecutionPath \cup (s', r)$
convert execution path to value function $\{(s, v)\}$
return $\{(s, v)\}$

$\texttt{ComputePolicy}(s, \tilde{V})$:
for $a \in A$ **do**
  $\tilde{Q}(s, a) = 0$ **for** $j = 1$ to $N_{samples}$ **do**
    $(s', r) \leftarrow \texttt{GSMDPstep}(s, a)$
    $\tilde{Q}(s, a) \leftarrow \tilde{Q}(s, a) + r + \gamma^{t'-t}\tilde{V}(s')$
  $\tilde{Q}(s, a) \leftarrow \frac{1}{N_{samples}}\tilde{Q}(s, a)$
return $\underset{a \in A}{\operatorname{argmax}}\, \tilde{Q}(s, a)$

Cet algorithme utilise les aspects d'exploration et d'évaluation par simulation. Il utilise alors les échantillons de fonction de valeur obtenus pour construire un régresseur sous la forme d'une machine à vecteurs supports (SVR). Il est intéressant de noter que cet algorithme effectue au final une amélioration incrémentale de la politique courante sans jamais utiliser explicitement cette dernière.

Une hypothèse importante concernant cette résolution concerne l'aspect non-Markovien. Nous maintenons l'hypothèse que les horloges (variables cachées du processus) ont un im-

pact négligeable sur la politique optimale. Toutefois, cet aspect non-Markovien nous impose de faire une hypothèse de "reproductibilité" du simulateur afin d'obtenir suffisamment d'échantillons pour évaluer des Q-valeurs.

Enfin, il est intéressant de noter que l'approche ATPI traite indifféremment les variables continues et discrètes.

13.3 Premiers résultats d'ATPI sur le problème du métro

Les tests de l'algorithme ATPI ont été effectués sur une instance du problème de contrôle d'un réseau de métro. Dans ce problème, l'état observable est constitué du nombre de passagers sur les quais et dans les rames, ainsi que de la position des trains et de l'heure courante de la journée. L'espace d'actions consiste en la possibilité de mettre en service ou de retirer des rames. Les récompenses sont acquises à chaque sortie d'un passager, tandis que les coûts correspondent à la consommation électrique du réseau.

La figure 13.1 présente l'évolution de la valeur de l'état initial étant donné la politique courante au fur et à mesure des itérations.

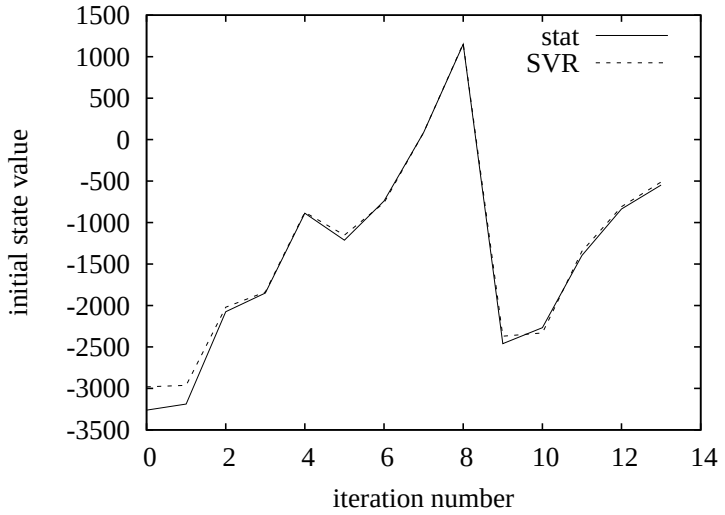


FIG. 13.1 – Problème du métro — Evolution de la politique

L'utilisation des SVR pour la régression apporte certains avantages mais est grandement handicapée par le biais que constitue l'opérateur SVR en lui-même. Une bonne politique est trouvée à l'itération huit mais la politique suivante semble catastrophique. Nous proposons l'explication illustrée par la figure 13.2 tenant au fait que la formulation ATPI est incomplète : dans certaines parties inexplorées de l'espace d'états, la fonction de valeur est non-pertinente et ne doit pas être utilisée, au risque de se ré-engager dans des régions à faibles récompenses déjà explorées.

Cette version "naïve" d'ATPI appelle donc à être complétée par une mesure de confiance en la fonction de valeur.

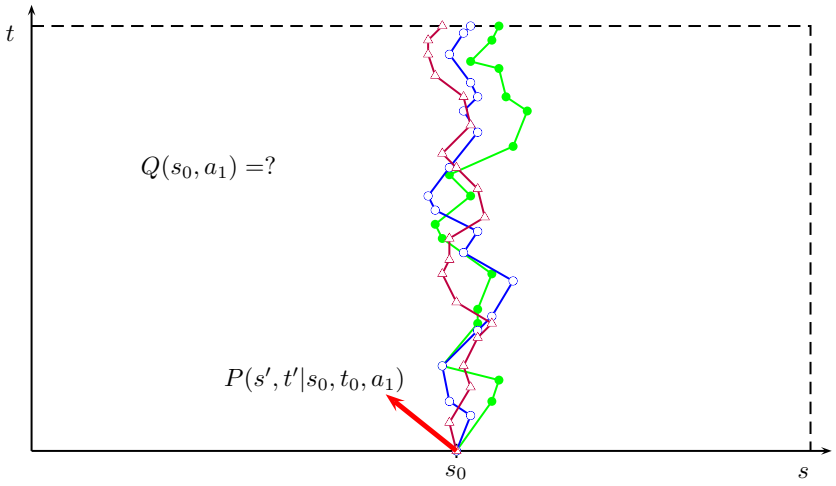


FIG. 13.2 – La pathologie de l'exploration pour l'évaluation

14

L'algorithme ATPI amélioré

Ce chapitre présente la version finale de notre travail sur le contrôle de systèmes temporels à événements explicites. Il s'appuie sur les résultats de chacun des chapitres précédents et introduit une description de plus haut niveau des systèmes que nous cherchons à contrôler. Cette description s'appuie sur le paradigme de description des systèmes à événements discrets et définit la classe des systèmes contrôlables temporels à événements discrets (DECTS). Sur la base de cette formulation et des conclusions du précédent chapitre sur l'approche ATPI, nous introduisons l'algorithme *improved ATPI*, qui corrige le problème de fiabilité de la fonction de valeur d'ATPI. Cet algorithme synthétise les contributions précédentes dans un cadre unifié. Nous présentons plusieurs exemples d'implantation d'*improved ATPI* et discutons des premiers résultats sur le problème du métro.

14.1 Définition des systèmes temporels contrôlables à événements discrets (DECTS)

Nous définissons les systèmes temporels contrôlables à événements discrets (DECTS) dans le cadre DEVS comme des systèmes à événements discrets possédant un port d'entrée correspondant à de possibles commandes qui conditionnent leur évolution. Cette dernière n'est pas nécessairement Markovienne. Par ailleurs, chacune des transitions d'un DECTS présente une extension temporelle. On peut résumer ce modèle par la figure 14.1.

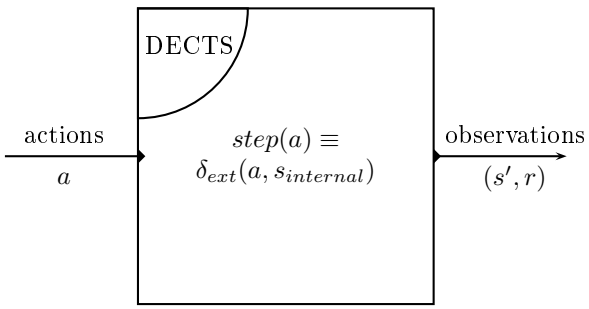


FIG. 14.1 – Représentation schématique d'un DECTS dans le cadre DEVS

Nous cherchons à décrire un algorithme d'apprentissage comme un système à événements

discrets. Pour ce faire nous nous appuyons sur le concept de simulations récursives et sur le cadre de la création et du contrôle dynamique du graphe des modèles de simulation. Cela permet de définir un "learner" DECTS comme un modèle exécutif DS-DEVS [Barros, 1997] et de lui permettre de créer des expériences sous forme de DECTS comme illustré par la figure 14.2.

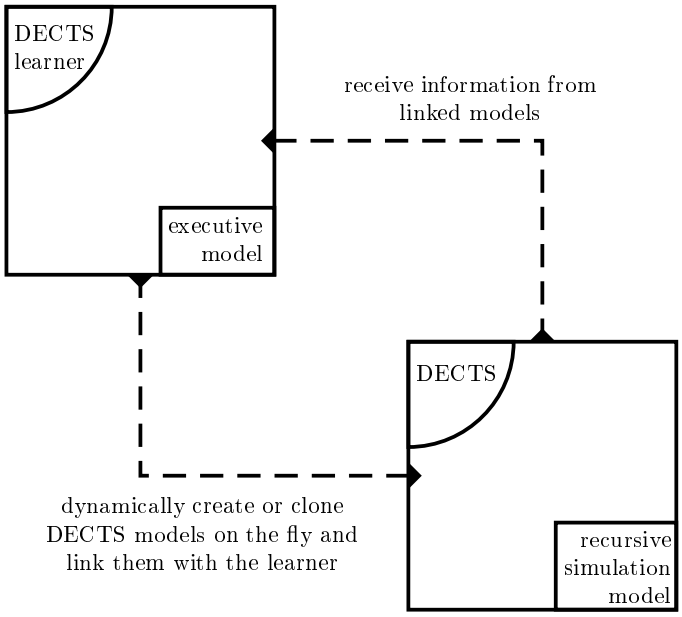


FIG. 14.2 – Modélisation d'un DECTS-learner dans le cadre événements discrets

Nous illustrons cette représentation unifiée de l'algorithme et du système de simulation sous la forme du DECTS-learner de l'algorithme naive ATPI, figure 14.3.

14.2 Retour sur les idées de base d'ATPI

L'idée de base d'ATPI est commune avec les approches de type Monte-Carlo en général : simuler pour explorer et simuler pour évaluer. Cette approche s'appuie sur la connaissance de l'état initial du processus et d'un modèle génératif.

Une première originalité d'ATPI est l'exploitation de la présence d'un temps observable pour pouvoir utiliser un critère total et surtout pour pouvoir garantir des simulations de durée finie. Plus précisément, ATPI se fonde sur la probabilité nulle d'observer une séquence infinie de transitions entre états ayant la même date.

Par ailleurs, l'exploration dans les espaces métriques hybrides est un problème crucial d'efficacité et nous prenons l'option d'effectuer des explorations partielles. Ces explorations appellent alors une notion de généralisation.

Cependant, le couple "exploration partielle" - "généralisation" implique de définir ce qui a été précédemment introduit comme une notion de confiance. Il est nécessaire de pouvoir définir quels éléments, dans l'espace d'état (ou dans l'espace des couples états-actions), ont été explorés ou sont assez proches d'éléments explorés pour pouvoir utiliser la généralisation.

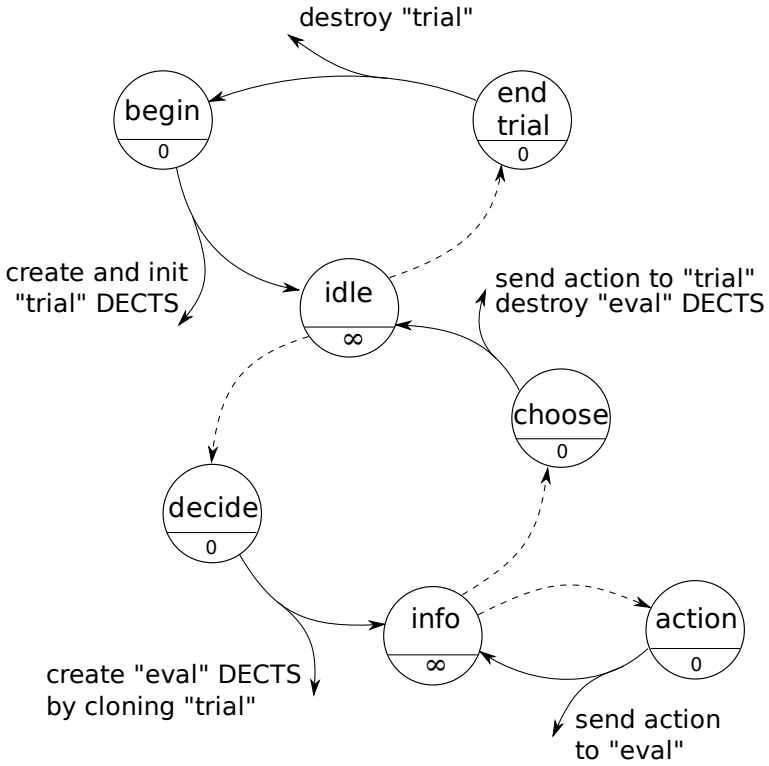


FIG. 14.3 – Le DECTS-learner de *naive* ATPI

Cette notion de confiance se ramène, d'un point de vue statistique, à l'évaluation d'une statistique suffisante pour les paramètres que l'on cherche à mesurer (notamment la fonction de valeur). Nous rapprochons ce problème de celui de la recherche de la densité de probabilité de la distribution décrivant les états visités par la politique courante.

Enfin, nous utilisons cette notion de confiance afin de reconstruire l'algorithme ATPI qui se conçoit alors comme un problème d'apprentissage statistique complet comprenant :

- Un problème de régression, en ligne, incrémentale pour la fonction de valeur.
- Un problème d'estimation de densité, en ligne, incrémentale pour la confiance en la fonction de valeur.
- Un problème de classification hors-ligne pour la politique.
- Un problème d'estimation de densité hors-ligne pour la fonction de confiance de la politique.

Ces quatre objets constituent les quatre *objets de décision* (decision objects) utilisés dans le DECTS-learner de l'algorithme *improved ATPI*.

14.3 L'algorithme *improved ATPI*

Nous pouvons alors présenter l'algorithme *iATPI* sous la forme du pseudo code de l'algorithme 14.1, ou encore sous la forme plus compacte du DECTS-learner correspondant présenté figure 14.4.

On peut notamment représenter le processus de création et de destruction de simulations d'*iATPI* ainsi que les différentes références temporelles associées par la figure 14.5.

Algorithm 14.1: Improved online-ATPI : *iATPI*

main :
input: π_0, s_0
repeat
  $valueDB$.`reset`() `/* new trial */`
  $actionDB$.`reset`()
  for $i = 1$ to N_{sim} **do**
    σ.`reset`()
    $trialProcess$.`init`(s_0)
    while horizon not reached **do**
      $a =$`bestAction`(s)
      $trialProcess$.`activateEvent`(a)
      $(s', r) \leftarrow trialProcess$.`step`()
      σ.`add`(s, a, r)
    $valueDB$.`convertExecutionPathToValueFunction`(σ)
    $actionDB$.`add`(σ)
  $V_n, C_{V_n} \leftarrow$ `trainRegressor`($valueDB$) `/* decision objects */`
  $\pi_n, C_{\pi_n} \leftarrow$ `trainClassifier`($actionDB$)
until termination

`bestAction`(s) :
for $a \in A_s$ **do**
  $\tilde{Q}(s, a) = 0$
  for $j = 1$ to N_a **do**
    $\tilde{Q}(s, a) = \tilde{Q}(s, a) +$ `simulateWithStop`(s, a) `/* test the actions */`
  $\tilde{Q}(s, a) = \frac{\tilde{Q}(s,a)}{N_a}$
return $\arg\max_{a \in A} \tilde{Q}(s, a)$

`simulateWithStop`(s) :
$\sigma_{eval} \leftarrow \emptyset$
$evalProcess = trialProcess$.`clone`()
$evalProcess$.`activateEvent`(a)
$(s', r) \leftarrow evalProcess$.`step`()
$Q \leftarrow r$
$s \leftarrow s'$
while horizon not reached and $C_{V_n}(s) =$ `false` **do** `/* explore until confident */`
  $a = \pi_n(s)$
  $evalProcess$.`activateEvent`(a)
  $(s', r) \leftarrow evalProcess$.`step`()
  $Q \leftarrow Q + r$
  $s \leftarrow s'$
  σ_{eval}.`add`(s, r)
$Q = Q + V_n(s)$
$valueDB$.`convertExecutionPathToValueFunction`(σ_{eval})
$V_n, C_{V_n} \leftarrow$ `reTrainRegressor`($valueDB$) `/* update value function */`
return Q

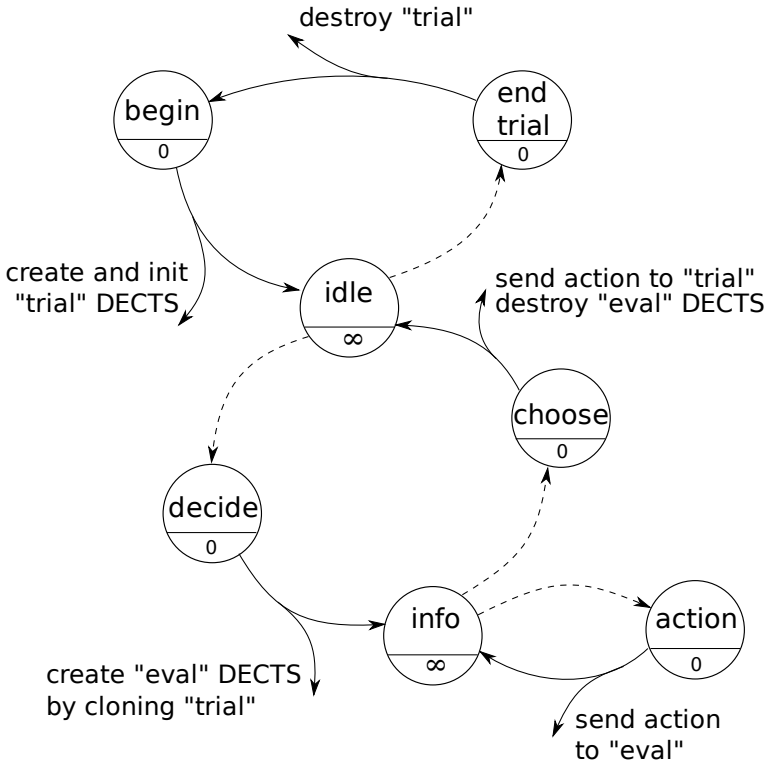


FIG. 14.4 – Le DECTS-learner d'*improved ATPI*

14.4 Premiers retours d'expérience avec *iATPI* — résultats et difficultés

L'algorithme *iATPI* se prête à de nombreuses implantations possibles dépendant des méthodes d'apprentissage statistique utilisées. Nous discutons de plusieurs essais d'implantation résumés par le tableau 14.1 puis nous focalisons sur les deux méthodes permettant un apprentissage réellement en ligne. Ces deux méthodes sont désignées dans le tableau par "compact iATPI" et "full storage iATPI".

π_n + C_{π_n} \ V_n + C_{V_n}	None + $s \mapsto$ false	SVR + OC-SVM	LWPR + RF activ$^{\text{n}}$ level	Parzen regr. + window.
MC-SVM + OC-SVM	Monte-Carlo iATPI	"non-naive" iATPI	compact iATPI	—
local vote + Parzen w.	—	—	—	full storage iATPI

TAB. 14.1 – Plusieurs version d'*iATPI*

La version "compact iATPI" nous permet de mettre en évidence d'intérêt du filtrage par la fonction de confiance. Elle met également en lumière les avantages et inconvénients de l'usage d'une méthode de régression locale comme LWPR [Vijayakumar et al., 2005].

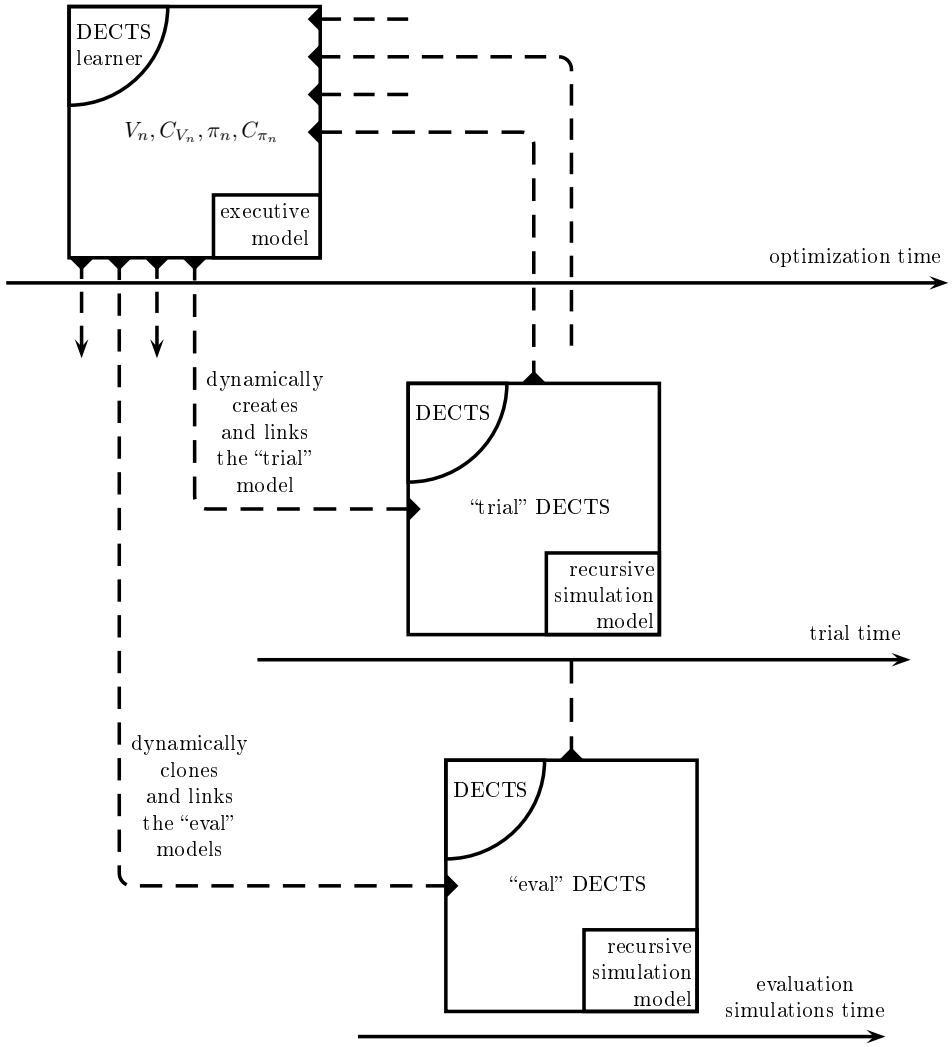


FIG. 14.5 – Illustration des différentes échelles temporelles du DECTS-learner d'*iATPI*

Cette méthode met également en évidence une difficulté propre au problème du métro qui tient à la forte variance des variables aléatoires observées et qui affecte fortement la difficulté de construction en ligne d'un bon régresseur.

Certains aspects sont corrigés par la version "full storage" d'*iATPI*. Cette version s'appuie sur le stockage des échantillons collectés par simulation sans post-traitement. Pour cela, un système de recherche par coordonnées dans notre espace d'états de grande dimension évite les faiblesses des représentations tabulaires.

Les différentes fonctions (de valeur, de confiance et la politique) sont alors calculées en ligne par régression de Parzen. La fonction de valeur utilise une méthode de fenêtrage local en norme L_1 puis une pondération uniforme ou de type L_2 pour définir une moyenne. Les fonctions de confiance sont définies par la densité locale des échantillons. La politique est obtenue par un système de vote majoritaire entre échantillons.
Le problème des fortes variances et l'ajustement automatique de la fonction de confiance est obtenu en dérivant des bornes de confiance à partir du théorème central-limite.

Cette version d'*iATPI* est à ce jour la version la plus prometteuse testée, malgré son fort coût en terme de stockage d'échantillons.

14.5 Conclusion

Ce chapitre a posé des fondations saines à l'algorithme *iATPI*. Cet algorithme d'itération de la politique asynchrone effectuant une exploration partielle de l'espace d'états et une amélioration incrémentale de la politique en tirant parti de la présence d'une variable temporelle fournit maintenant de meilleures garanties que l'algorithme ATPI naïf du chapitre précédent.

Par ailleurs l'utilisation des techniques d'apprentissage statistique — en ligne ou hors ligne — reste un sujet ouvert sur lequel plusieurs axes de recherche sont actuellement explorés par d'autres équipes. *iATPI* tire une grande partie de son originalité du couplage exploration partielle - évaluation partielle et de la définition des zones explorées (zones de confiance) et des zones inconnues.

Bien entendu, il est encore trop tôt pour tirer des conclusions définitives quant à cette approche et de nombreux tests et améliorations sont encore nécessaires et constituent un objectif essentiel des travaux post-thèse.

Enfin, la représentation DECTS permet de représenter dans un cadre commun les agents apprenants et les processus à contrôler. Du point de vue de la représentation et de la multi-modélisation des systèmes, cela ouvre une porte nouvelle à la représentation et au couplage de processus de simulation et de processus d'optimisation.

15 Conclusion

Ce chapitre de conclusion résume la progression des idées suivies au cours de cette partie III. Il rassemble nos contributions de façon synthétique et s'ouvre sur les futurs domaines de travail.

15.1 Bilan

Cette partie III s'est concentrée sur la résolution de problèmes décisionnels de Markov temporels à événements explicites. Partant de la constatation que ce tels processus sont souvent trop complexes pour être directement modélisés comme des MDP, nous nous sommes tournés vers le cadre générique de la théorie DEVS afin de représenter et construire les modèles génératifs des systèmes à contrôler. Ceci a motivé nos premières contributions : nous avons exprimé le problème de contrôle comme un GSMDP hybride à temps continu et illustré la complexité de construction d'un modèle prédictif du système ainsi que son aspect non-Markovien. Cela nous a permis d'établir un pont entre modèles DEVS et GSM(D)P et d'obtenir une description générale du système à contrôler, compatible de façon générale avec tout système à événements discrets.

Nous avons alors exploité l'idée de simulation gloutonne pour la recherche en avant dans le cadre le l'itération de la politique asynchrone, créant ainsi le cadre RTPI. Cet algorithme est fortement inspiré par les idées de programmation dynamique asynchrone de [Bertsekas and Tsitsiklis, 1996] et par l'exploration gloutonne de [Barto et al., 1995] et [Bonet and Geffner, 2003b]. Cette idée constitue notre seconde contribution et est motivée par le fait que — pour les problèmes temporels à résoudre — nous connaissions l'état de départ, disposions d'une méthode de simulation de politique et souhaitions limiter l'exploration de l'espace d'états aux états les plus pertinents.

Ces deux premières briques de simulation et de procédure algorithmique ont alors permis de définir l'algorithme ATPI qui exploite le temps observable introduit dans les GSMDP afin d'éviter de stocker une politique pendant l'exécution de RTPI et afin de compense la perte de la propriété de Markov et de faciliter les itérations de Bellman. L'algorithme "*naive* ATPI" constitue notre troisième contribution et le premier pas vers sa version améliorée. Nous avons évalué ATPI sur le problème du métro, illustrant ainsi à la fois ses résultats prometteurs et ses faiblesses fondamentales.

Ces premiers résultats nous ont alors permis de tirer plusieurs conclusions. Avant tout,

nous avons pris un certain recul avec le cadre des GSMDP et nous sommes attachés à caractériser les propriétés essentielles des systèmes que nous souhaitions contrôler. Ceci nous a amenés à définir, comme quatrième contribution, les systèmes temporels contrôlables à événements discrets, ou DECTS, qui s'insèrent dans le cadre de la théorie DEVS et englobent les propriétés de dynamique par événements des GSMDP, leur comportement non-Markovien et la classe générale des problèmes de contrôle temporel à événements discrets. L'un des points cruciaux de cette définition réside dans la représentation du système simulé et du processus d'optimisation dans un langage commun utilisant le principe des modèles dynamiques et des simulations récursives et introduisant la définition générale d'objets de décision. Ceci permet alors de considérer une procédure d'optimisation en décision séquentielle comme un système à événements discrets et permet, par exemple de construire des procédures d'optimisation de façon hiérarchique, dans un langage commun.

Enfin, sur la base de la description DECTS, nous avons reformulé l'algorithme ATPI afin d'en éliminer les précédentes faiblesses. Ceci mène à notre dernière contribution : *iATPI*. Cet algorithme rassemble des résultats de simulation, la méthode d'RTPI et les outils empruntés à l'apprentissage statistique. Il cherche à résoudre des problèmes décisionnels de Markov temporels de grande dimension, à espace d'état continu, en présentant les caractéristiques suivantes :

- Exploration par simulation (recherche gloutonne en avant)
- Evaluation par simulation (évaluation de Monte-Carlo)
- Généralisation locale dans l'espace d'états de l'expérience reçue (régularité du régresseur / classificateur pour les objets de décision que nous avons considérés).

Bien que le temps nous ait manqué afin de tester extensivement les dernières versions d'*iATPI*, les premiers résultats ont ouvert de nombreuses possibilités d'amélioration et ont soulevé de nouvelles questions. En particulier, ils ont fournis des indications quant à la pertinence statistique des objets de décision utilisés. Ils ont également permis de raffiner la notion de confiance, qui permet de guide l'exploration pour l'optimisation. Ils ont enfin montré les propriétés nécessaires des outils de régression, classification et estimation de densité appropriés lors d'une utilisation conjointe avec *iATPI*.

15.2 Perspectives

Bien entendu, la première étape des travaux futurs consiste à tester extensivement les différentes implantations d'*iATPI*. Ces tests devront impliquer différentes versions des outils d'apprentissage statistique mais également porter sur d'autres cas d'évaluation que les problèmes abordés dans la thèse.

La construction d'une version efficace d'*iATPI* dans certains cas pourra impliquer la définition d'autres outils dédiés afin de construire de bonnes politiques et fonctions de valeur. Une étude approfondie de l'algorithme LWPR semble notamment une bonne piste de recherche, entre autres pour compenser son important besoin d'échantillons et pour l'adapter au problèmes locaux de classification par exemple.

Une autre perspective intéressante se situe dans le cadre du modèle DECTS. Avoir écrit le système simulé et l'algorithme d'apprentissage en utilisant un langage commun de description nous permet de considérer l'ensemble système-contrôleur comme un nouveau système à événements discrets et de construire hiérarchiquement d'autres systèmes d'optimisation

à un niveau plus élevé. Entre autres choses, ceci ouvre une perspective aux techniques de vérification, validation et test dans un cadre unifié à événements discrets.

Quatrième partie

Conclusion

Conclusion générale

Partant de la problématique initiale de décider, sous contraintes d'incertitude, dans le cadre de problèmes non-stationnaires, nous avons exploré deux cadres de modélisation distincts, correspondant chacun à une extension des processus décisionnels de Markov.

Nous avons tout d'abord fait l'hypothèse qu'un modèle stochastique "tout-intégré" du processus à contrôler était disponible sous la forme d'un TMDP. Ces TMDP formalisent la notion de modèles de décision à événements implicites. Sur la base de cette modélisation, nous nous sommes appliqués à fournir un cadre solide à l'équation d'optimalité, à améliorer le processus de résolution et à étendre l'expressivité du modèle. Cela s'est fait au travers de l'introduction de deux modèles plus généraux : SMDP+ et XMDP, qui ont mis en lumière les avantages et les limites des TMDP, généralisé leur équation d'optimalité et ouvert de nouvelles pistes pour leur résolution.

Mais les modèles de décision à événements implicites sont rarement disponibles directement et il est bien plus aisé de décrire un problème de décision temporelle via un modèle à événements explicites. L'inconvénient de ces derniers est qu'il devient impossible d'y optimiser des politiques avec les mêmes garanties d'optimalité générales que dans le cadre MDP. Par conséquent, nous sommes attachés en premier lieu à comprendre et formaliser ces systèmes temporels à événements explicites. Cette étude nous a menés vers le champs de la théorie de spécification des systèmes à événements discrets DEVS et vers l'usage de méthodes d'apprentissage statistique dans un but commun : construire un algorithme générique d'optimisation de politique pour ces problèmes décisionnels de Markov temporels à événements explicites.

Les différents domaines explorés et les contributions apportées dans les parties II et III peuvent être résumés de la façon suivante :

- La partie II s'est concentrée sur la formalisation des modèles de décision à événements implicites, étendant le modèle TMDP à un cadre de représentation et d'optimisation plus général et plus expressif (les représentations polynômiales par morceaux) et introduisant le cas général d'un temps continu avec la formalisation XMDP. Les efforts de résolution de tels problèmes temporels ce sont tournés vers des algorithmes de recherche en arrière, effectuant une optimisation de type itération de la valeur de façon asynchrone.
- La partie III s'est, elle, tournée vers la modélisation des systèmes à événements explicites où l'inclusion de la variable temporelle a en partie compensé la parte de la propriété de Markov. Les contributions algorithmiques se sont alors portées sur

des méthodes de recherche en ligne et en avant, utilisant les propriétés générales des méthodes d'apprentissage statistique et traitant des espaces d'états de grande dimension.

Nous ne nous attarderons pas plus ici sur les contributions amenées par l'étude de ces deux domaines distincts et renvoyons pour cela le lecteur aux chapitres de conclusion de leurs parties respectives (chapitres 10 et 15). A la place, il semble pertinent de prendre du recul vis-à-vis de notre variable temporelle afin de tirer des conclusions plus générales.

La discussion du chapitre 2, soulignant les différences et similarités entre MDP standards et problèmes décisionnels de Markov temporels, a souligné trois différentes notions de temps dans le contexte des processus stochastiques : le temps du processus (les numéros des périodes de décision successives), le temps de transition ou de séjour (l'extension temporelle des transitions) et le temps physique ou horloge du système (la mesure à laquelle l'agent à accès en regardant sa montre). Nous sommes restés dans le cadre des processus à événements discrets ce qui nous a menés à nous concentrer sur les deux dernières notions, leurs relations et dépendances. Au final il apparaît que :

- L'horloge du système *peut* être traitée comme toute autre variable non renouvelable du processus. La généralisation des TMDP aux XMDP en est probablement la meilleure illustration. Même s'ils définissent un temps non borné a priori, les TMDP reposent sur l'hypothèse implicite que la connaissance de l'instationnarité est limitée dans le temps. Ainsi, on pourrait considérer un TMDP comme un MDP ayant une variable continue. Toutefois,
- l'horloge du système de *devrait pas* être traitée comme n'importe quelle variable continue dans un problème décisionnel de Markov temporel. La première raison en est que cette variable structure l'évolution du processus : elle conditionne les modèles de transition, durée et récompense et, par conséquent, la politique optimale. Donner au temps une place particulière dans le processus d'optimisation — comme le font les cadres TMDP et XMDP — permet de structurer la résolution en tirant parti du principe de causalité[1]. De plus, l'introduction d'une variable temporelle continue observable dans le cadre GSMDP a permis, quand l'état initial est connu, de compenser en partie la perte de la propriété de Markov due à la non-observabilité de la dynamique interne décidant des temps de séjour.

D'un point de vue pratique, les algorithmes $TMDP_{poly}$ et *iATPI*, qui forment la principale contribution algorithmique de la thèse, peuvent bénéficier d'améliorations, d'une meilleure compréhension de leurs mécanismes et de meilleures implantations. Toutefois, ils constituent tous deux des propositions nouvelles dans leurs domaines respectifs et en regard de la question de la planification et de l'apprentissage dans l'incertain.

Finalement, cette question de gestion du temps est loin d'être close. D'autres communautés de recherche on adopté des formalismes différents pour s'attaquer à la question : logique temporelle, planification déterministe temporelle, TMDP, équations différentielles de propagation, ... tous constituent des tentatives distinctes de représentation de la dépendance temporelle des problèmes de décision et de leur structure. Nous avons présenté dans cette thèse une contribution au cadre spécifique de la décision dans l'incertain, essayant, à un niveau modeste, de construire des ponts entre communautés (décision dans l'incertain et modélisation à événements discrets par exemple). Cependant, à la fois dans nos domaines et dans le cas général, cette variable temporelle conserve une certaine aura d'incompris, conformant que son étude nécessite plus de ... temps.

[1] conséquence directe du fait que le temps soit une variable non renouvelable.

Bibliographie

Ahlberg, J. H., Nielson, E. N., and Walsh, J. L. (1967). *The Theory of Spline Functions and Their Applications*. Academic Press, New York.

Ahn, M. S. and Kim, T. G. (1993). Analysis on Steady State Behavior of DEVS Models. In *International Conference on AI, Simulation and Planning in High Autonomy Systems*.

Altman, E. (1999). *Constrained Markov Decision Processes*. Chapman & Hall/CRC, London.

Altman, E. and Shwartz, A. (1993). Time-Sharing Policies for Controlled Markov Chains. *Operations Research*, 41(6) :1116–1124.

Alur, R. and Dill, D. L. (1994). A theory of timed automata. *Theoretical Computer Science*, 126(2) :183–235.

Anderson, C. W. (2000). Approximating a Policy can be easier than Approximating a Value Function. Technical Report TR-CS-00-101, Colorado State University.

Andre, D., Friedman, N., and Parr, R. (1998). Generalized Prioritized Sweeping. In *Neural Information Processing Systems*, pages 1001–1007.

Antos, A., Munos, R., and Szepesvari, C. (2007). Fitted Q-iteration in continuous action-space MDPs. In *Neural Information Processing Systems*.

Atkeson, C., Moore, A., and Schaal, S. (1997). Locally Weighted Learning. *Artificial Intelligence*, 11(4) :76–113.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning Journal*, 47(2–3) :235–256.

Barros, F. J. (1997). Modelling Formalisms for Dynamic Structure Systems. *ACM Transactions on Modelling and Computer Simulation*, 7 :501–515.

Barto, A. G., Bradtke, S. J., and Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72 :81–138.

Baxter, J. and Bartlett, P. (1999). Direct gradient-based reinforcement learning : I. Gradient estimation algorithms. Technical report, Computer Science Laboratory, Australian National University.

Bellman, R. E. (1954). The Theory of Dynamic Programming. *Bulletin of the American Mathematical Society*, 60 :503–516.

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey.

Benazera, E., Brafman, R., Meuleau, N., Mausam, and Hansen, E. A. (2005). An AO* Algorithm for Planning with Continuous Resources. In *Workshop on Planning under Uncertainty for Autonomous Systems, at ICAPS*.

Bernstein, D. S. and Zilberstein, S. (2001). Reinforcement Learning for Weakly-Coupled MDPs and an Application to Planetary Rover Control. In *Conference on Uncertainty in Artificial Intelligence*.

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific.

Bertsekas, D. P. and Shreve, S. E. (1996). *Stochastic Optimal Control : The Discrete-Time Case*. Athena Scientific. Originally published in 1978.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Bonet, B. and Geffner, H. (2003a). Faster Heuristic Search Algorithms for Planning with Uncertainty and Full Feedback. In *International Joint Conference on Artificial Intelligence*.

Bonet, B. and Geffner, H. (2003b). Labeled RTDP : Improving the convergence of real-time dynamic programming. In *International Conference on Automated Planning and Scheduling*, pages 12–21.

Boutilier, C., Dean, T., and Hanks, S. (1999). Decision-theoretic Planning : Structural Assumptions and Computational Leverage. *Journal of Artificial Intelligence Research*, 11 :1–94.

Boutilier, C., Dearden, R., and Goldszmidt, M. (2000). Stochastic Dynamic Programming with Factored Representations. *Artificial Intelligence*, 121(1–2) :49–107.

Bouyer, P., Cassez, F., Fleury, E., and Larsen, K. G. (2004). Optimal Strategies in Priced Game Automata. In *Foundations of Software Technology and Theoretical Computer Science*.

Boyan, J. A. and Littman, M. L. (2001). Exact Solutions to Time Dependent MDPs. *Advances in Neural Information Processing Systems*, 13 :1026–1032.

Bradtke, S. J. and Barto, A. G. (1996). Linear Least-Squares Algorithms for Temporal Difference Learning. *Machine Learning*, 22(2) :33–57.

Bresina, J., Dearden, R., Meuleau, N., Ramakrishnan, S., and Washington, R. (2002). Planning under Continuous Time and Resource Uncertainty : a Challenge for AI. In *Conference on Uncertainty in Artificial Intelligence*.

Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer.

Chang, C.-C. and Lin, C.-J. (2001). *LIBSVM : a library for support vector machines*. Software available at `http://www.csie.ntu.edu./tw/~cjlin/libsvm`.

Chang, H. S., Fu, M. C., Hu, J., and Marcus, S. I. (2007). *Simulation-based Algorithms for Markov Decision Processes*. Communications and Control Engineering. Springer-Verlag London.

Chen, T., Morris, J., and Martin, E. (2006). Probability Density Estimation via Infinite Gaussian Mixture Model : Application to Statistical Process Monitoring. *Journal of the Royal Statistical Society (series C)*, 55(1) :699–715.

Coquelin, P.-A. and Munos, R. (2007). Bandit Algorithm for Tree Search. In *Conference on Uncertainty in Artificial Intelligence*.

Cox, D. R. and Miller, H. D. (1965). *The Theory of Stochastic Processes*. John Wiley & Sons, Inc.

Cushing, W., Kambhampati, S., Mausam, and Weld, D. S. (2004). When is Temporal Planning Really Temporal ? In *International Conference on Automated Planning and Scheduling*.

Dai, P. and Goldsmith, J. (2007). Multi-Threaded BLAO* Algorithm. In *FLAIRS Conference*, pages 56–61.

Dean, T. L. and Kanazawa, K. (1990). A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3) :142–150.

Dean, T. L. and Lin, S.-H. (1995). Decomposition Techniques for Planning in Stochastic Domains. In *International Joint Conference on Artificial Intelligence*.

Dearden, R. (2001). Structured Prioritized Sweeping. In *International Conference on Machine learning*.

Dedecker, J. (2008). Inégalités de Hoeffding et théorème limite central pour les fonctions peu régulières de chaînes de markov non irréductibles. *Numéro spécial des Annales de l'ISUP*, 52 :39–46.

d'Epenoux, F. (1963). A Probabilistic Production and Inventory System. *Management Science*, 10(1) :98–108.

Dimitrikakis, C. and Lagoudakis, M. (2008). Algorithms and Bounds for Sampling-based Approximate Policy Iteration. In *European Workshop on Reinforcement Learning*.

Ernst, D., Geurts, P., and Wehenkel, L. (2005). Tree-Based Batch Mode Reinforcement Learning. *Journal of Machine Learning Research*, 6 :503–556.

Farahmand, A., Ghavamzadeh, M., Szepesvári, C., and Mannor, S. (2008). Regularized Policy Iteration. In *Neural Information Processing Systems*.

Feng, Z., Dearden, R., Meuleau, N., and Washington, R. (2004). Dynamic Programming for Structured Continuous Markov Decision Problems. In *Conference on Uncertainty in Artificial Intelligence*.

Ferguson, D. and Stentz, A. (2004). Focussed Dynamic Programming : Extensive Comparative Results. Technical Report CMU-RI-TR-04-13, Robotics Insitute, Carnegie Mellon University.

Ghallab, M., Nau, D., and Traverso, P. (2004). *Automated Planning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Gilmer Jr., J. B. and Sullivan, F. J. (2005). Issues in Event Analysis for Recursive Simulation. In *Winter Simulation Conference*.

Glynn, P. (1989). A GSMP Formalism for Discrete Event Systems. *Proc. of the IEEE*, 77.

Guestrin, C., Hauskrecht, M., and Kveton, B. (2004). Solving Factored MDPs with Continuous and Discrete Variables. In *Conference on Uncertainty in Artificial Intelligence*.

Guestrin, C., Koller, D., and Parr, R. (2001). Max-norm Projections for Factored MDPs. In *International Joint Conference on Artificial Intelligence*, pages 673–682.

Hansen, E. A. and Zilberstein, S. (2001). LAO* : a heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, 129(1-2).

Hasselt, H. and Wiering, M. A. (2007). Reinforcement Learning in Continuous Action Spaces. In *IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning*.

Hauskrecht, M. and Kveton, B. (2004). Linear program approximations for factored continuous-state Markov decision processes. *Advances in Neural Information Processing Systems*, 16 :895–902.

Hauskrecht, M. and Kveton, B. (2006). Approximate Linear Programming for Solving Hybrid Factored MDPs. In *International Symposium on Artificial Intelligence and Mathematics*.

Hauskrecht, M., Meuleau, N., Kaelbling, L. P., Dean, T. L., and Boutilier, C. (1998). Hierarchical Solution of Markov Decision Processes using Macro-actions. In *Conference on Uncertainty in Artificial Intelligence*, pages 220–229.

Hoey, J., St. Aubin, R., Hu, A., and Boutilier, C. (2000). Optimal and Approximate Stochastic Planning using Decision Diagrams. Technical Report TR-2000-05, University of British Columbia - Vancouver, BC, Canada.

Howard, R. A. (1963). Semi-Markovian Decision Processes. In *34th Session of the International Statistical Institute*.

Joslyn, C. (1996). The Process Theoretical Approach to Qualitative DEVS. In *International Conference on AI, Simulation and Planning in High Autonomy Systems*.

Kaelbling, L. P. (1990). *Learning In Embedded Systems*. PhD thesis, Stanford University, Department of Computer Science.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101 :99–134.

Kearns, M. J., Mansour, Y., and Ng, A. Y. (2002). A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes. *Machine Learning*, 49 :193–208.

Kocsis, L. and Szepesvari, C. (2006). Bandit Based Monte-Carlo Planning. In *European Conference on Machine Learning*.

Korf, R. E. (1990). Real-Time Heuristic Search. *Artificial Intelligence*, 42 :189–211.

Kveton, B. and Hauskrecht, M. (2006). Learning Basis Functions in Hybrid Domains. In *AAAI Conference on Artificial Intelligence*.

Lagoudakis, M. and Parr, R. (2003). Least-Squares Policy Iteration. *Journal of Machine Learning Research*, 4 :1107–1149.

Li, L. and Littman, M. L. (2005). Lazy Approximation for Solving Continuous Finite-Horizon MDPs. In *National Conference on Artificial Intelligence*.

Littman, M. L., Dean, T. L., and Kaelbling, L. P. (1995). On the Complexity of Solving Markov Decision Problems. In *Conference on Uncertainty in Artificial Intelligence*, volume 11, pages 394–402.

Liu, Y. and Koenig, S. (2006). Functional Value Iteration for Decision-Theoretic Planning with General Utility Functions. In *National Conference on Artificial Intelligence*.

Marecki, J., Topol, Z., and Tambe, M. (2006). A Fast Analytical Algorithm for Markov Decision Process with Continuous State Spaces. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 2536–2541.

Mausam (2007). *Stochastic Planning with Concurrent, Durative Actions*. PhD thesis, University of Washington.

Mausam, Benazera, E., Brafman, R., Meuleau, N., and Hansen, E. A. (2005). Planning with continuous resources in stochastic domains. In *International Joint Conference on Artificial Intelligence*, pages 1244–1251.

Mausam and Weld, D. S. (2005). Concurrent Probabilistic Temporal Planning. In *International Conference on Automated Planning and Scheduling*.

Mausam and Weld, D. S. (2006). Probabilistic Temporal Planning with Uncertain Durations. In *National Conference on Articifial Intelligence*.

Mausam and Weld, D. S. (2007). Planning with Durative Actions in Stochastic Domains. *Journal of Artificial Intelligence Research*, 31 :33–82.

Maxwell, M. and Woodroofe, M. (2000). Central Limit Theorems for Additive Functionals of Markov Chains. *Annals of Probability*, 28(2) :713–724.

McMahan, H. B., Likhachev, M., and Gordon, G. J. (2005). Bounded real-time dynamic programming : RTDP with monotone upper bounds and performance guarantees. In *International Conference on Machine learning*, pages 569–576.

Melamed, B. (1976). *Analysis and Simplification of Discrete Event Systems and Jackson Queuing Networks*. PhD thesis, University of Michigan.

Meuleau, N., Hauskrecht, M., Kim, K.-E., Peshkin, L., Kaelbling, L. P., Dean, T., and Boutilier, C. (1998). Solving Very Large Weakly Coupled Markov Decision Processes. In *AAAI Conference on Artificial Intelligence*.

Moore, A. W. and Atkeson, C. G. (1993). Prioritized Sweeping : Reinforcement Learning with Less Data and Less Real Time. *Machine Learning Journal*, 13(1) :103–105.

Munos, R. (2003). Error Bounds for Approximate Policy Iteration. In *International Conference on Machine Learning*.

Munos, R. (2007). Performance Bounds for Approximate Value Iteration. *SIAM Journal on Control and Optimization*, 46(2) :541–561.

Munos, R. and Moore, A. W. (2000). Rates of Convergence for Variable Resolution Schemes in Optimal Control. In *International Conference on Machine Learning*.

Munos, R. and Moore, A. W. (2002). Variable Resolution Discretization in Optimal Control. *Machine Learning Journal*, 49(2-3) :291–323.

Neuts, M. R. (1981). *Matrix-geometric solutions in stochastic models : an algorithmic approach*. The John Hopkins University Press, Baltimore.

Nielsen, F. (1998). GMSim : a tool for compositionnal GSMP modeling. In *Winter Simulation Conference*.

Ormoneit, D. and Sen, S. (2002). Kernel-Based Reinforcement Learning. *Machine Learning Journal*, 49 :161–178.

Parr, R. (1998). Flexible Decomposition Algorithms for Weakly Coupled Markov Decision Problems. In *Conference on Uncertainty in Artificial Intelligence*.

Parzen, E. (1962). On the Estimation of a Probability Density Function and the Mode. *Annals of Mathematics and Statistics*, 33 :1065–1076.

Peng, J. and Williams, R. J. (1993). Efficient Learning and Planning Within the Dyna Framework. *Adaptive Behaviour*, 1(4) :437–454.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes : The Art of Scientific Computing, Third Edition*. Cambridge University Press.

Puterman, M. L. (1994). *Markov Decision Processes*. John Wiley & Sons, Inc.

Péret, L. (2004). *Recherche en ligne pour les Processus Décisionnels de Markov : application à la maintenance d'une constellation de satellites*. PhD thesis, Institut National Polytechnique de Toulouse.

Péret, L. and Garcia, F. (2003). On-line Search for Solving Large Markov Decision Processes. In *European Workshop on Reinforcement Learning*.

Péret, L. and Garcia, F. (2004). On-line Search for Solving Markov Decision Processes via Heuristic Sampling. In *European Conference on Artificial Intelligence*.

Quesnel, G., Duboz, R., Ramat, E., and Traore, M. K. (2007). VLE - A Multi-Modeling and Simulation Environment. In *Moving Towards the Unified Simulation Approach, Proc. of the 2007 Summer Simulation Conf.*, pages 367–374.

Rabiner, L. R. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286.

Rachelson, E., Fabiani, P., Farges, J.-L., Teichteil, F., and Garcia, F. (2006). Une approche du traitement du temps dans le cadre MDP : trois méthodes de découpage de la droite temporelle. In *Journées Françaises Planification Décision Apprentissage*. F. Garcia, G. Verfaillie editors.

Rachelson, E., Garcia, F., and Fabiani, P. (2008a). Extending the Bellman Equation for MDP to Continuous Actions and Continuous Time in the Discounted Case. In *International Symposium on Artificial Intelligence and Mathematics*.

Rachelson, E., Quesnel, G., Garcia, F., and Fabiani, P. (2008b). A Simulation-based Approach for Solving Generalized Semi-Markov Decision Processes. In *European Conference on Artificial Intelligence*.

Rachelson, E., Quesnel, G., Garcia, F., and Fabiani, P. (2008c). Approximate Policy Iteration for Generalized Semi-Markov Decision Processes : an Improved Algorithm. In *European Workshop on Reinforcement Learning*.

Roth, V. (2004). The Generalized LASSO. *IEEE Transations on Neural Networks*, 15(1).

Sabbadin, R. (2002). Graph partitioning techniques for Markov Decision Processes decomposition. In *European Conference on Artificial Intelligence*, pages 670–674.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A., and Williamson, R. (2001). Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(1) :1443–1471.

Smith, T. and Simmons, R. G. (2006). Focused Real-Time Dynamic Programming for MDPs : Squeezing more out of a Heuristic. In *AAAI Conference on Artificial Intelligence*.

Smola, A. and Schölkopf, B. (1998). A Tutorial on Support Vector Regression. Technical Report NC-TR-98-030, Royal Holloway College, University of London, NeuroCOLT Technical Report.

Sturm, C. (1835). *Mémoire sur la résolution des équations numériques*. Ins. France Sc. Math. Phys., t. 6.

Sutton, R. S. (1995). TD Models : Modeling the World at a Mixture of Time Scales. In *International Conference on Machine Learning*, pages 531–539.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning : An Introduction*. The MIT Press, Cambridge, MA.

Teichteil-Königsbuch, F. and Infantes, G. (2008). Tr-FSP : Forward Stochastic Planning using Probabilistic Reachability. In *International Symposium on Search Techniques in Artificial Intelligence and Robotics*.

Tesauro, G. and Galerpin, G. R. (1997). On-line Policy Improvement using Monte-Carlo Search. *Advances in Neural Information Processing Systems*, pages 1068–1072.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the royal Statistical Society : series B*, 58(1) :267–288.

Vapnik, V., Golowich, S., and Smola, A. (1996). Support Vector Method for Function Approximation, Regression Estimation and Signal Processing. *Advances in Neural Information Processing Systems*, 9 :281–287.

Vijayakumar, S., D'Souza, A., and Schaal, S. (2005). Incremental Online Learning in High Dimensions. *Neural Computation*, 17 :2602–2634.

Wang, G., Yeung, D.-Y., and Lochovsky, F. H. (2007). The Kernel Path in Kernelized LASSO. In *AISTATS*.

Watkins, C. J. C. (1989). *Learning from Delayed Rewards*. PhD thesis, Cambridge University.

Watkins, C. J. C. and Dayan, P. (1992). Q-learning. *Machine Learning*.

Wellman, M., Ford, M., and Larson, K. (1995). Path Planning under Time-Dependent Uncertainty. In *Conference on Uncertainty in Artificial Intelligence*, pages 532–539.

Whiteson, S. and Stone, P. (2006). Evolutionary Function Approximation for Reinforcement Learning. *Journal of Machine Learning Research*, 7 :877–917.

Williams, R. J. and Baird, L. C. (1993). Tight Performance Bounds on Greedy Policies Based on Imperfect Value Functions. Technical Report NU-CCS-93-14, College of Computer Science, Northeastern University, Boston, Massachussetts.

Younes, H. L. S. and Simmons, R. G. (2004). Solving Generalized Semi-Markov Decision Processes using Continuous Phase-Type Distributions. In *AAAI Conference on Artificial Intelligence*.

Zeigler, B. P. (1976). *Theory of Modeling and Simulation*. Wiley Interscience.

Zeigler, B. P., Kim, D., and Praehofer, H. (2000). *Theory of modeling and simulation : Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press.