Complexité, Approximation et Hasard
Optimisation et Vérification
en univers incertain

Richard Lassaigne
IMJ/Logique mathématique
CNRS-Université Paris Diderot

- Problèmes de **planning** et **vérification**
  pour les systèmes avec **non déterminisme** et **probabilités**

- Méthodes classiques :
  algorithmes en temps **polynomial** ?

- Problème : la **modélisation** donne un modèle
  dont la **représentation** peut ne pas tenir en mémoire

- La complexité en **espace** est primordiale
  dans un problème comme la **vérification**

- Peut-on utiliser des méthodes d'**approximation**
  et des algorithmes **probabilistes** ?

Pb	SAT	VC	CLIQUE	TSP	KNAPSACK
Déci.	NP-com.	NP-com.	NP-com.	NP-com.	NP-com.
Opt.	MaxSAT	MinVC	MaxCLIQ.	MinTSP	Pseudo-Poly
Approx.	$\varepsilon = 1/2$	$\varepsilon = 1/2$			FPTAS
Non Approx.	FPTAS $\Rightarrow$ P=NP		ε-approx. $\Rightarrow$P=NP	ε-approx. $\Rightarrow$P=NP	

Fully Polynomial-Time Approximation Scheme (FPTAS) : poly($|x|$, $1/\varepsilon$)

- Processus de décision markoviens (**MDP**)
  Problème de décision markovien (**planning**)

- Méthodes classiques :
  Itération sur la fonction de valeur (**Value Iteration**)
  Itération sur la stratégie (**Policy Iteration**)

- Vérification **probabiliste**
  Schémas probabilistes d'**approximation**

- Approximation pour le **planning** et la **vérification**

Quelques acteurs majeurs

Richard Bellman

Ronald Howard

Christos Papadimitriou

Michael Kearns

Modèles dynamiques :

- Processus **stochastiques** bien connus
- Critères de **performance** utilisés en
  recherche opérationnelle, économie,
  optimisation combinatoire, théorie du contrôle...

Applications (parmi beaucoup d'autres) :

- Modélisation de protocoles de communication
- Modélisation du contrôle dans les réseaux
- Optimisation dans les réseaux de capteurs sans fil
- Contrôle dans le systèmes embarqués

Modèle de décision :

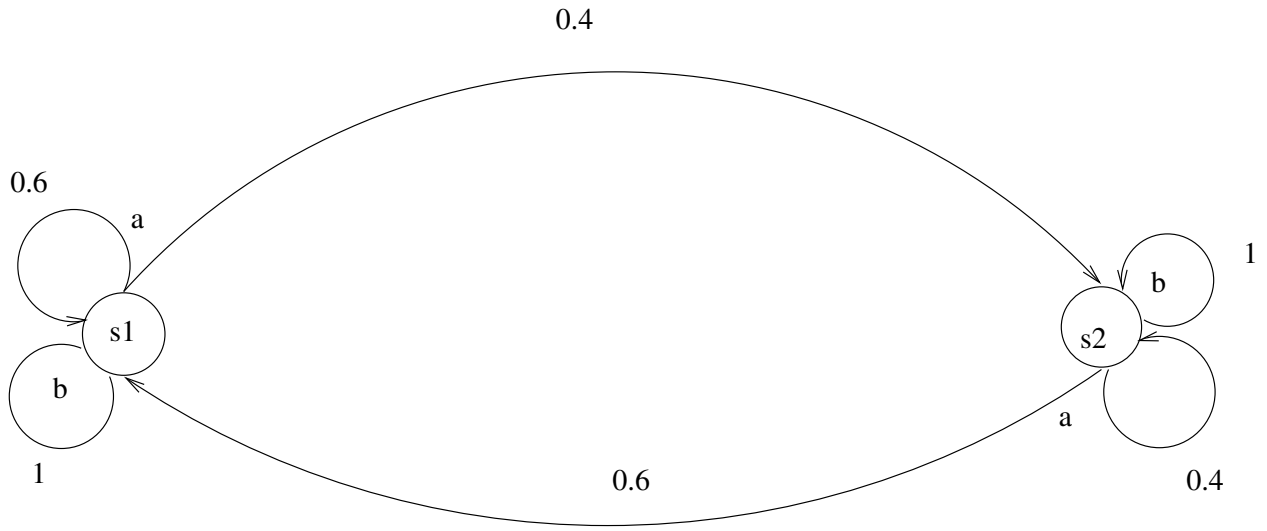
- à chaque étape, un agent (ou un contrôleur) observe l'**état** du système et choisit une **action** (non déterminisme)
- il reçoit une **récompense** (ou subit un **coût**) et le système évolue vers un nouvel **état** en suivant une **distribution de probabilités** déterminée par le choix

Problème de décision markovien :

- Processus de décision markovien
- Critère de **performance**

Stratégie :

- Fonction associant une **action** à chaque **état**
- Détermine une suite de transitions dont la **valeur** est la **récompense** totale ou le **coût** total



$$\mathsf{P}_a = \begin{pmatrix} 0.6 & 0.4 \\ 0.6 & 0.4 \end{pmatrix} \qquad \mathsf{P}_b = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

s	act	R(s,act)
s_1	a	1
s_1	b	0
s_2	a	0
s_2	b	1

$$\mathcal{M} = (S, s_0, A, P, R)$$

- S ensemble fini d'**états** ($\mid S \mid = n$)
- s_0 état initial (ou μ distribution de probabilités sur les états)
- A ensemble d'**actions** ($\mid A \mid = m$)
- $P : S \times A \longrightarrow Distr(S)$ Relation de **transition**
  $Distr(S)$ ensemble des **distributions de probabilités** sur S
- $R : S \times A \longrightarrow \mathcal{R}^+$ Fonction de **récompense** (ou de **coût**)

Pour chaque couple état-action $(s, a) \in S \times A$:

- $P_{s,a}(.)$ **distribution de probabilités** sur les états
- $R_{s,a}$ **récompense** (ou **coût**) associée à l'action a
  à partir de l'état s

Stratégie (stationnaire) :

- **déterministe** et sans mémoire $\pi : S \longrightarrow A$
- **probabiliste** et sans mémoire $\pi : S \longrightarrow Distr(A)$

Arbre des **trajectoires** :

Chaque branche correspond à l'application d'une stratégie

Remarque :

- Si l'on se restreint à une stratégie π, on obtient la **chaîne de Markov** induite par la stratégie π
- On peut alors faire une étape de **vérification** de propriétés intéressantes (accessibilité, sûreté)

Problème de décision markovien :
trouver une stratégie **optimale** suivant un critère de **performance**

- Horizon H **fini** : récompense totale **espérée**

$$V^{\pi}(s) = \mathbb{E}_s^{\pi}(\sum_{i=1}^{H} R(s_i, \pi(s_i)))$$

  s_i étant l'état résultant à l'étape i suivant
  la chaîne de Markov induite par la stratégie π
- Horizon **infini** : récompense totale **espérée**
  avec taux de discount $0 < \gamma < 1$

$$V^{\pi}(s) = \mathbb{E}_s^{\pi}(\sum_{i=1}^{\infty} \gamma^{i-1} R(s_i, \pi(s_i)))$$

- Horizon **infini** : récompense **espérée moyenne**

$$V^{\pi}(s) = \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}_s^{\pi}(\sum_{i=1}^{T} R(s_i, \pi(s_i)))$$

Problème de décision markovien : Horizon **infini**, avec **discount**

Fonction **auxiliaire** (Q-fonction) :
$Q^{\pi}(s,a) = R_{s,a} + \mathbb{E}_s^{\pi}(\sum_{s' \in S} \gamma P_{s,a}(s') V^{\pi}(s'))$

Fonction de récompense **optimale** (ou coût optimal) :
$V^*(s) = max_{\pi} V^{\pi}(s)$ et $Q^*(s,a) = max_{\pi} Q^{\pi}(s,a)$ pour tout $(s,a) \in S \times A$
Stratégie **optimale** π^* : $\pi^*(s) = argmax_a Q^*(s,a)$ pour tout $s \in S$

Equation de **Bellman** :

$$V^*(s) = max_{a \in A}(R(s,a) + \gamma \sum_{s' \in S} P_{s,a}(s').V^*(s'))$$

Résolution :

- Méthodes itératives (value iteration, policy iteration)
- Programmation linéaire

Algorithme :

- Pour chaque $s \in S$, initialiser $V_0(s)$
- $h := 1$
- Tant que $n <$ nb maximum d'itérations
  pour chaque $(s, a) \in S \times A$,
  $Q_h(s, a) := R(s, a) + \gamma \sum_{s' \in S} P_{s,a}(s').V_{h-1}(s')$
  $V_h(s) := max_{a \in A} Q_h(s, a)$
  $h := h + 1$
- Pour chaque $s \in S$, $\pi^*(s) := argmax_{a \in A} Q_h(s, a)$
- Retourner π^*

Nombre maximum d'itérations :

- soit l'horizon **fini** H
- soit déterminé par un **critère d'arrêt** (horizon infini)
  $max_{s \in S} \mid V_h(s) - V_{h-1}(s) \mid \leq \varepsilon' = \varepsilon \frac{(1-\gamma)}{2\gamma}$

Ce critère garantit que la stratégie résultante π^* est ε-**optimale**

Le temps de calcul pour chaque itération est $O(mn^2)$

Algorithme :

- Soit π_0 une stratégie stationnaire déterministe
- Boucle :

  $\pi := \pi_0$

  Déterminer, pour chaque $s \in S$, $V^{\pi}(s)$ par résolution de l'équation de **Bellman** (à n inconnues)

  Pour chaque $s \in S$, s'il existe $a \in A$ t.q. $(R(s,a) + \gamma \sum_{s' \in S} P_{s,a}(s').V^{\pi}(s')) < V^{\pi}(s)$, alors $\pi_0(s) := a$, sinon $\pi_0(s) := \pi(s)$

  Répéter la boucle si $\pi \neq \pi_0$
- Retourner π

Remarque :

- Il existe au plus m^n stratégies distinctes
- Comme chaque stratégie améliore la précédente (Puterman, 1994) l'algorithme termine en au plus un nombre **exponentiel** d'étapes

Algorithme en 2 phases :

- Détermination de la fonction **valeur** (résolution d'un système d'équations linéaires) en $O(n^3)$ étapes
- Amélioration (éventuelle) de la **stratégie** en $O(mn^2)$ étapes

Complexité :

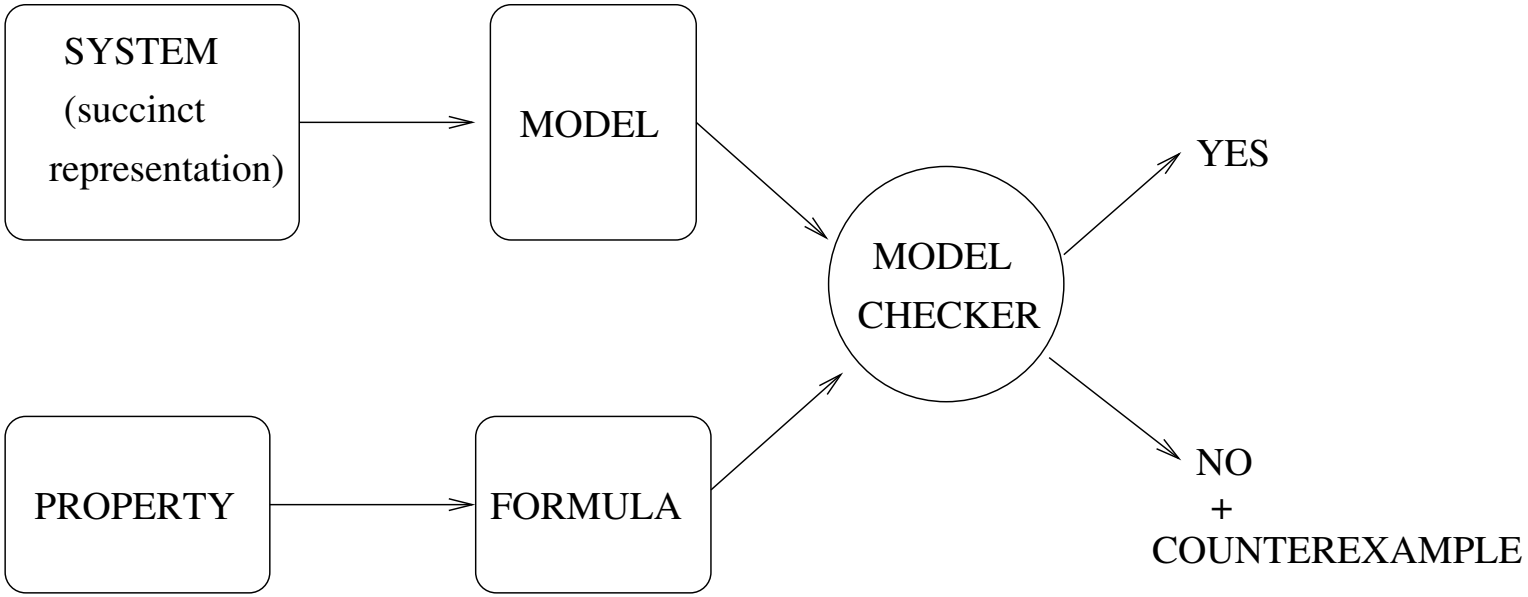
- Le nombre d'itérations est en $O(\frac{mn}{1-\gamma} log(\frac{n}{1-\gamma}))$ (Ye, 2010)
- Le nombre d'itérations est en $O(\frac{m}{1-\gamma} log(\frac{n}{1-\gamma}))$ (Hansen, Miltersen et Zwick, 2011)
- Borne exponentielle pour le problème sans discount, ou lorsque le taux de discount fait partie de l'entrée (Friedmann, 2009, Fearnley, 2010)

Théorème (C. Papadimitriou, J.N. Tsitsiklis, 1987) :
Le problème de décision markovien est **P-complet** dans les 3 cas (horizon fini, horizon infini avec discount ou coût moyen)

Idée de la preuve :
Réduction du "Circuit Value Problem" (CVP) au problème de décision markovien

Remarque : Cas "déterministe" (les probas sont égales à 0 ou 1)
Les trois problèmes correspondants sont dans la classe **NC** (problèmes décidables en temps polynomial sur une RAM parallèle avec un nombre de processeurs polynomial)

Problème : **P** $\neq$ **NC** ?



Entrée :

- Modèle $\mathcal{M} = (S, R)$ $R \subseteq S^2$ (relation de transition)
- Etat initial s_0
- Formule φ (Logique Temporelle Linéaire : **LTL**)

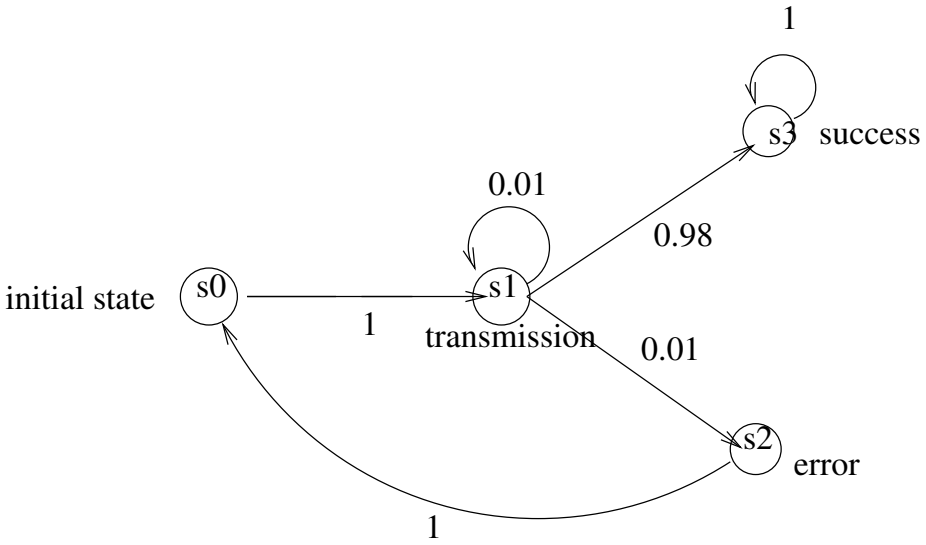
Sortie :

- OUI si $(\mathcal{M}, s_0) \models \varphi$
- NON avec trace d'erreur si $(\mathcal{M}, s_0) \not\models \varphi$

Chaînes de Markov

Entrée :

- Modèle $\mathcal{M} = (S, P, L)$ et état initial s_0
- $P : S^2 \longrightarrow [0, 1]$ Fonction de probabilité
- $L : S \longrightarrow 2^{AP}$ (étiquetage des états)
- Formula ψ (**LTL**)



Sortie : $Prob_{\Omega}[\psi]$

Exemple : $\psi \equiv transmission$ Until $success$

(Ω espace probabiliste des chemins d'exécution d'origine s_0)

Complexité : (Courcoubetis et Yannakakis, 1995)

Vérification qualitative (i.e. $prob > 0$?)

Même **complexité** que **model checking pour LTL**

$$O(|M|.2^{|\psi|})$$

Vérification Quantitative (i.e. $prob =$?)

$$O(poly(|M|).2^{|\psi|})$$

Problème :

- Phénomène d'explosion combinatoire (dû à la modélisation)
- Le problème n'est pas le temps, mais l'espace utilisé pour la représentation explicite du modèle

Exemple : PRISM (Probabilistic Model Checker)

Protocole (probabiliste) du diner des philosophes

Une solution :

- Méthode d'approximation probabiliste
- Utilisation d'une représentation succincte du modèle

Algorithme probabiliste A

- Entrée : instance x d'un problème de comptage, $\varepsilon, \delta > 0$
- Sortie : valeur $A(x, \varepsilon, \delta)$ telle que

$$Pr[(1-\varepsilon)\#(x) \leq A(x,\varepsilon,\delta) \leq (1+\varepsilon)\#(x)] \geq 1-\delta$$

Schéma probabiliste d'approximation polynomial : FPRAS

Le temps de calcul est $poly(|x|, (1/\varepsilon), \log(1/\delta))$

Théorème (LP08) :
L'existence d'un FPRAS pour calculer $Prob_\Omega(\psi)$ ($\psi \in LTL$)
entrainerait $RP = NP$

Restrictions sur l'approximation probabiliste :

- approximation absolue
- propriétés monotones (ex : accessibilité)
  ou anti-monotones (ex : sûreté)

Schéma probabiliste d'approximation en espace logarithmique

Classe RP (Randomised Polynomial Time) :
Problèmes P pour lesquels il existe un algorithme probabiliste A fonctionnant en temps polynomial t.q. pour toute entrée x

- si $P(x)$, alors $Prob_\Omega(A \text{ accepte } x) \geq \frac{1}{2}$
- si $\neg P(x)$, alors $Prob_\Omega(A \text{ accepte } x) = 0$

Ω est l' espace probabiliste des tirages de la machine

Exemple : Le problème de primalité est dans $coRP$

Algorithmes de Monte-Carlo (avec erreur d'1 seul côté) :

$$Prob_\Omega(P(x) \text{ et } A \text{ rejette } x) < \frac{1}{2}$$

Réduction exponentielle de la probabilité d'erreur
avec un nombre linéaire d'itérations de l'algorithme

Méthode :

Estimation (Monte-Carlo) + borne de Chernoff-Hoeffding

X variable de Bernoulli $(0, 1)$ avec probabilité de succès p

- Faire N tirages aléatoires independants $X_1, X_2, \ldots, X_N$
- Estimer p par $\mu = \sum_{i=1}^{N} X_i/N$ avec erreur absolue ε
- La taille de l'échantillon N est telle que la probabilité d'erreur (de l'algorithme) $< \delta$

Borne de Chernoff-Hoeffding :

$$Pr[\mu < p - \varepsilon] + Pr[\mu > p + \varepsilon] < 2e^{-2N\varepsilon^2}$$

Si $N \geq \ln(\frac{2}{\delta})/2\varepsilon^2$, alors

$$Pr[p - \varepsilon \leq \mu \leq p + \varepsilon)] \geq 1 - \delta$$

Algorithme générique d'approximation $\mathcal{GAA}$

entrée : générateur, $\phi, \varepsilon, \delta$

sortie : ε-**approximation** de $Prob_k(\psi)$

$A := 0$

$N := \log(\frac{2}{\delta})/2\varepsilon^2$

Pour $i := 1$ à N

- Engendrer de manière aléatoire un chemin σ de longueur k
- Si ψ est vraie sur σ alors $A := A + 1$

Retourner (A/N)

Algorithme basé sur une estimation de type Monte-Carlo et la borne de Chernoff-Hoeffding

Générateur : représentation succincte du système
(par exemple programme dans le langage d'entrée de PRISM)

Théorème :

$\mathcal{GAA}$ est un FPRAS pour $Prob_k(\psi)$

Méthodologie : Pour approximer $Prob_\Omega(\psi)$

- Choisir $k \approx log|M| \cdot ln(1/\varepsilon)$
- Itérer l'approximation de $Prob_k(\psi)$

Corollaire :

L'algorithme de point fixe obtenu en itérant $\mathcal{GAA}$ est un schéma probabiliste d'approximation en espace logarithmique pour $Prob(\psi)$

Remarque :

- La complexité en espace est logarithmique...
- La vitesse de convergence est régie par l'ordre de multiplicité de la 2e valeur propre (théorème de Perron-Frobenius)

Idée :

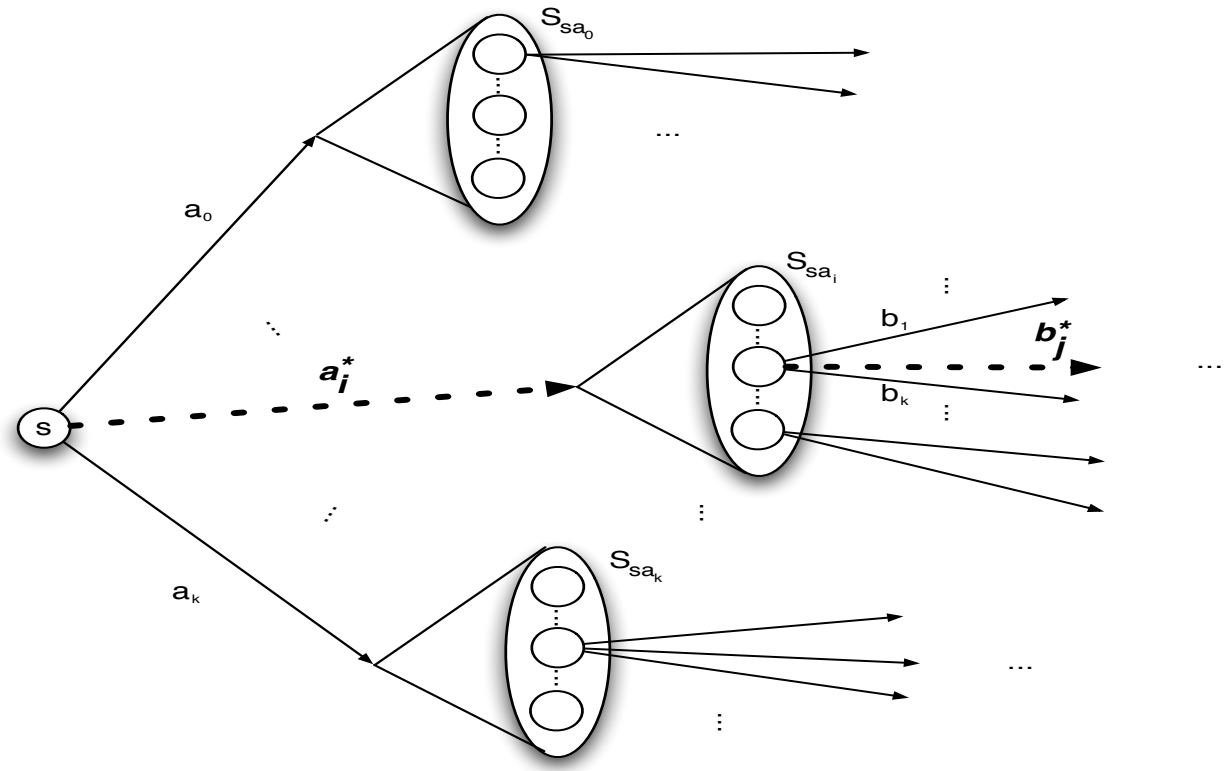
Etant donné un générateur (probabiliste) pour MDP $\mathcal{M}$

- déterminer une stratégie π^* ε-optimale
- construire une chaîne de Markov $\mathcal{M}$ par restriction du MDP $\mathcal{M}$ à la stratégie π^*

Générateur probabiliste :

Algorithme probabiliste qui pour chaque entrée $(s, a) \in S \times A$ fournit

- un état t engendré suivant la distribution de probabilités $P_{s,a}$
- la récompense (ou coût) associée $P_{s,a}$



Construction (inspirée de Kearns, 2001) :

Arbre des trajectoires d'un "plus petit" MDP $\mathcal{M}'$ de profondeur H'

- Arbre peu dense construit (look-ahead) à partir de l'état initial
- Chaque transition à partir d'un état s et d'une action a est restreinte à C successeurs suivant la distribution $P_{s,a}$

Remarque : $H' = 2H$ (H déterminé plus tard...)

Objectif :

Estimer la récompense espérée (avec discount) à l'horizon H

$$V_H^*(s) = \mathbb{E}_s^{\pi^*}(\sum_{i=1}^{H} \gamma^{i-1} R(s_i, \pi^*(s_i)))$$

Récompense espérée (avec discount) à l'horizon $h \geq 1$:
$V_h^*(s) = R_{s,a^*} + \gamma \sum_{s' \in S} P_{s,a^*}(s').V_{h-1}^*(s'))$ et $V_0^*(s) = 0$
a^* est l'action choisie par la statégie optimale à partir de s

Estimation $\hat{V}_h^*(s)$ de $V_h^*(s)$:

- pour chaque $a \in A$, utiliser le générateur pour obtenir $R_{s,a}$ et un échantillon $S_{s,a}$ de C états suivant la distribution $P_{s,a}$
- calculer $\hat{V}_{h-1}^*(s')$ pour chaque état s' dans les $S_{s,a}$
- l'estimation de $V_h^*(s)$ est donnée par

$$\hat{V}_h^*(s) = max_{a \in A}(R(s,a) + \gamma \frac{1}{C} \sum_{s' \in S_a} \hat{V}_{h-1}^*(s'))$$

Structure intermédiaire $\widehat{\mathcal{M}}$:

- pour chaque noeud s dans l'arbre, pour chaque action a
  on stocke les ensembles $S_{s,a}$ des C successeurs de s
- on étiquette l'action a correspondant à la valeur max par a^*
  on obtient ainsi une stratégie presque optimale $\hat{\pi}^*$

Arbre des trajectoires de la chaîne de Markov $\mathcal{M}^*$:

- on supprime les noeuds et les transitions correspondant
  à des actions non étiquetées
- on élague les branches à la profondeur H

Propriété :
pour chaque état s du MDP $\mathcal{M}'$ à profondeur $\leq H = \frac{H'}{2}$,
l'action optimale dans $\mathcal{M}'$ à partir de s est une action
presque optimale dans $\mathcal{M}$

Objectif :

Montrer que la stratégie π^* est ε-optimale à l'horizon infini

i.e. que pour tout état s de $\mathcal{M}^*$, $\mid V^*(s) - \hat{V}_H^*(s) \mid \leq \varepsilon$

Borne de Chernoff-Hoeffding :

Soit $\lambda > 0$. Pour tout état s et toute action a,

$\mathcal{P}rob\big(\mid \mathbb{E}_s^{\pi^*}(V^*(s)) - \frac{1}{C} \sum_{i=1}^{C} V^*(s_i) \mid \leq \lambda \big) \geq 1 - e^{-\lambda^2 . C / V_{max}^2}$

La probabilité d'une mauvaise estimation à l'issue de H étapes

est bornée par $(mC)^H e^{-\lambda^2 . C / V_{max}^2}$.

On choisit H pour que l'erreur soit $\leq 2\lambda/(1-\gamma)$

$H = log_\gamma(\frac{\lambda}{V_{max}})$ où $V_{max} = \frac{R_{max}}{1-\gamma}$

Il est possible de choisir $C = \widetilde{O}(\frac{R_{max}^2}{\varepsilon^2 (1-\gamma)^6})$ tel que :

la probabilité d'une mauvaise estimation soit $\leq \delta = \lambda / R_{max}$

Dernière étape (Kearns, d'après Singh et Yee, 1994) :

Soit π une stratégie probabiliste t. q. pour tout état s,

$$\mathcal{P}rob(\mid Q^*(s,\pi^*(s)) - Q^*(s,\pi(s)) \mid < \lambda) \geq (1-\delta)$$

Alors pour tout état s,

$$\mid V^*(s) - V^\pi(s) \mid \leq (\lambda + 2\delta V_{max})/(1-\gamma)$$

Il reste à choisir λ pour obtenir l'ε-approximation :
$\lambda = \varepsilon(1-\gamma)^2/4$

Ouf !

- Les méthodes classiques d'optimisation et de vérification pour les problèmes de décision markoviens sont polynomiales dans la taille du modèle

- Les méthodes d'approximation probabiliste ont permis d'éliminer la complexité en espace sur les chaînes de Markov et ont montré leur efficacité pratique (APMC dans le model-checker probabiliste PRISM)

- Les méthodes d'optimisation par génération probabiliste dans les MDPs sont indépendantes de la taille du modèle

- Cependant la complexité est exponentielle dans l'ε-horizon

- Il reste à montrer leur efficacité dans la pratique...

- [B57] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957
- [CY95] C. Courcoubetis and M. Yannakakis. *The complexity of probabilistic verification*. JACM, 24(4), p. 857-907, 1995
- [HMZ11] T.D. Hansen, P.B. Miltersen and U. Zwick. *Strategy iteration is strongly polynomial*. Proc. Innovations in Computer Science, p.253-263, 2011
- [H63] W. Hoeffding. *Probability inequalities for sums of bounded random variables*. Journal of the American Statistical Association, 58(301), p.13-30, 1963
- [H60] R.A. Howard. *Dynamic programming and Markov process*. MIT Press, 1960
- [KMN02] M. Kearns, Y. Mansour and A.Y. Ng. *A sparse sampling algorithm for near-optimal planning in large Markov decision processes*. Machine Learning, 49(2-3), p. 193-208, 2002

- [LDK95] M. Littman, T. Dean and L. Kaelbling. *On the complexity of solving markov decision problems*. Proc. of the 11th UAI, p. 394-402, 1995
- [LP08] R. Lassaigne et S. Peyronnet. *Probabilistic Verification and Approximation*. Journal of Pure and Applied Logic, vol. 152 (1-3), p. 122-131, 2008
- [LP12] R. Lassaigne et S. Peyronnet. *Approximate planning and verification for large Markov decision processes*. Proc. ACM Symposium on Applied Computing, p. 1314-1319, 2012
- [PT87] C.H. Papadimitriou and J.N. Tsitsiklis. *The complexity of Markov decision processes*. Mathematics of Operations Research, 12(3), p. 441-450
- [P94] M. Puterman. *Markov Decision Processes*. John Wiley and Sons, 1994