

# Processus de décision markovien

Un **processus de décision markovien** (MDP) est un modèle stochastique issu de la théorie de la décision et de la théorie des probabilités et (partiellement) sous le contrôle d'un preneur de décision. Les MDPs sont utilisés pour étudier un grand nombre de problèmes d'optimisation avec l'aide d'algorithmes de programmation dynamique ou d'apprentissage par renforcement. Les MDPs sont connus depuis 1950s (cf. Bellman 1957). Une grande contribution provient du travail de Ronald A. Howard avec son livre de 1960, 'Dynamic Programming and Markov Processes. Ils sont utilisés dans de nombreuses disciplines, incluant la robotique, l'automatisation, l'économie, et l'industrie manufacturière.

Plus précisément, un processus de décision markovien est un processus de contrôle stochastique discret. À chaque étape, le processus est dans un certain état  $s$ , et le décideur peut choisir une action  $a$  accessible dans un état  $s$ . L'action fait passer de l'état  $s$  à l'état  $s'$  récoltant au passage une récompense  $R(s, a, s')$  donnée au décideur.

La probabilité que le processus arrive à l'état  $s'$  est déterminée par l'action choisie. Plus précisément, elle est décrite par la fonction de transition d'états  $T(s, a, s')$ . Donc, l'état  $s'$  dépend de l'état actuel  $s$  et de l'action  $a$  sélectionnée par le décideur. Cependant, pour un  $s$  et un  $a$ , le prochain état est indépendant des actions et états précédents. On dit alors que le processus satisfait la *propriété de Markov*.

Les MDPs sont une extension des chaînes de Markov. La différence est l'addition des actions (permettant de choisir) et des récompenses (donnant la motivation de ces choix). Dans le cas où une action existe pour toutes les étapes et les récompenses sont les mêmes, le processus de décision Markovien est une chaînes de Markov.

## 1 Définition intuitive

Afin de comprendre ce qu'est un MDP, supposons que l'on ait un système évoluant dans le temps comme un automate probabiliste. À chaque instant le système est dans un état donné et il existe une certaine probabilité pour que le système évolue vers tel ou tel autre état à l'instant suivant en effectuant une transition.

Supposons maintenant que l'on doive contrôler ce système *boite noire* de la meilleure façon possible. L'objectif est de l'amener dans un état considéré comme *bénéfique*, en évitant de lui faire traverser des états *néfastes*. Pour cela, on dispose d'un ensemble d'actions possibles sur le sys-

tème. Pour compliquer la chose, on supposera que l'effet de ces actions sur le système est probabiliste : l'action entreprise peut avoir l'effet escompté ou un tout autre effet. L'efficacité du contrôle est mesurée relativement au gain ou à la pénalité reçue au long de l'expérience.

Ainsi, un raisonnement à base de MDP peut se ramener au discours suivant : *étant dans tel cas et choisissant telle action, il y a tant de chance que je me retrouve dans tel nouveau cas avec tel gain.*

Pour illustrer les MDP, on prend souvent des exemples issus de la robotique mobile (avec les positions pour états, les commandes comme actions, les mouvements comme transitions et l'accomplissement/échec de tâches comme gains/pénalités).

## 2 Hypothèse de Markov

Dans les MDP, l'évolution du système est supposée correspondre à un processus markovien. Autrement dit, le système suit une succession d'états distincts dans le temps et ceci en fonction de probabilités de transitions. L'hypothèse de Markov consiste à dire que les probabilités de transitions ne dépendent que des  $n$  états précédents. En général, on se place à l'ordre  $n = 1$ , ce qui permet de ne considérer que l'état courant et l'état suivant.

## 3 Définition formelle

Un MDP est un quadruplet  $\{S, A, T, R\}$  définissant :

- un ensemble d'états  $S$ , qui peut être fini, dénombrable ou continu ; cet ensemble définit l'environnement tel que perçu par l'apprenant (dans le cas d'un robot, on peut voir cela comme l'ensemble produit des valeurs de ses différents capteurs) ;
- un ensemble d'actions  $A$ , qui peut être fini, dénombrable ou continu et dans lequel l'apprenant choisit les interactions qu'il effectue avec l'environnement (dans le cas d'un robot on peut voir cela comme l'ensemble produit des paramètres de ses différentes commandes) ;
- une fonction de transition  $T : S \times A \times S \rightarrow [0; 1]$  ; cette fonction définit l'effet des actions de

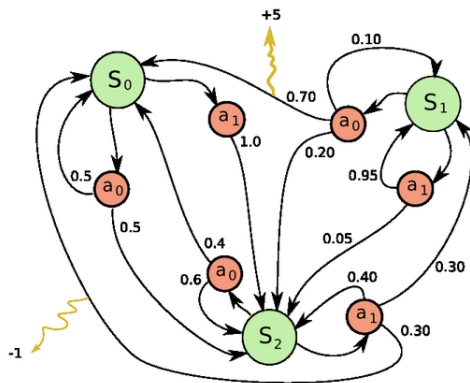
l'apprenant sur l'environnement :  $T(s, a, s')$  représente la probabilité de se retrouver dans l'état  $s'$  en effectuant l'action  $a$ , sachant que l'on était à l'instant d'avant dans l'état  $s$ .

$T$  ainsi définie représente le cas le plus général ; dans un environnement déterministe, on aura plutôt  $T : S \times A \rightarrow S$ .

- une *fonction de récompense*  $R : S \times A \times S \times \mathbb{R} \rightarrow [0; 1]$  ; elle définit la récompense (positive ou négative) reçue par l'agent :  $R(s, a, s', v)$  est la probabilité d'obtenir une récompense  $v$  pour être passé de l'état  $s$  à  $s'$  en ayant effectué l'action  $a$ . Ici encore cette définition est très générale, bien souvent on se contentera par exemple des cas particuliers suivants :
  - $R : S \times A \times S \rightarrow \mathbb{R}$  (récompense déterministe, c'est le choix que nous adopterons dans la suite) ;
  - $R : S \times A \rightarrow \mathbb{R}$  (récompense déterministe rattachée à l'action en ignorant son résultat) ;
  - $R : S \rightarrow \mathbb{R}$  (récompense déterministe rattachée à un état donné).

NB : nous ne considérons ici que les modèles dans lesquels le temps est discrétisé, c'est-à-dire que la « trajectoire » d'un apprenant dans l'environnement est décrivable par une suite d'états  $s_t$  ( $t \in \mathbb{N}$ ), et non par une fonction  $s(t)$  avec  $t \in \mathbb{R}$ . De même on notera  $a_t$  la suite des actions prises par l'agent. On pourra consulter<sup>[1]</sup> pour une description des MDP à temps continu.

## 4 Exemple de MDP



Exemple simple de **Processus de Décision Markovien** à trois états et à deux actions.

L'exemple donné ci-contre représente un **Processus de Décision Markovien** à trois états distincts  $\{s_0, s_1, s_2\}$

représentés en vert. Depuis chacun des états, on peut effectuer une action de l'ensemble  $\{a_0, a_1\}$ . Les nœuds rouges représentent donc une décision possible (le choix d'une action dans un état donné). Les nombres indiqués sur les flèches sont les probabilités d'effectuer la transition à partir du nœud de décision. Enfin, les transitions peuvent générer des récompenses (dessinées ici en jaune).

- La matrice de transition associée à l'action  $a_0$  est la suivante :

$$\begin{pmatrix} 0.50 & 0 & 0.50 \\ 0.70 & 0.10 & 0.20 \\ 0.40 & 0 & 0.60 \end{pmatrix}$$

- La matrice de transition associée à l'action  $a_1$  est la suivante :

$$\begin{pmatrix} 0 & 0 & 1.0 \\ 0 & 0.95 & 0.05 \\ 0.30 & 0.30 & 0.40 \end{pmatrix}$$

En ce qui concerne les récompenses,

- on perçoit une récompense de +5 lorsque l'on passe de l'état  $s_1$  à l'état  $s_0$  en accomplissant l'action  $a_0$
- on perçoit une récompense de  $-1$  (aussi appelée pénalité) lorsque l'on passe de l'état  $s_2$  à l'état  $s_0$  en accomplissant l'action  $a_1$

## 5 Remarques

Le modèle **MDP** présenté ici est supposé stable dans le temps, c'est-à-dire que les composants du quadruplet sont supposés invariants. Il n'est donc pas applicable en l'état pour un système qui évolue, par exemple pour modéliser un système qui apprend contre un autre agent.

## 6 Politiques, fonctions de valeurs et équations de Bellman

La politique d'un agent est la manière avec laquelle il choisit l'action qu'il va effectuer lorsqu'il se trouve dans un état donné. Formellement il s'agit donc d'une fonction  $\pi : S \rightarrow A$  dans le cas d'une politique déterministe (ce que nous supposerons dans la suite), ou  $\pi : S \times A \rightarrow [0; 1]$  dans le cas **stochastique** : on a donc  $a_t = \pi(s_t)$ .

Afin que l'agent puisse choisir une politique, il faut lui fournir un critère de choix. Ce critère est lié à la fonction de récompense  $R$ . Notons  $r_t = R(s_t, \pi(s_t), s_{t+1})$  la récompense effective obtenue au temps  $t$  (après avoir effectué son action) par l'agent qui suit la politique  $\pi$ . Voici plusieurs choix possibles de critères d'intérêts que l'agent peut chercher à maximiser :

- $E\left(\sum_{t=0}^h r_t\right)$  : espérance de la somme des récompenses à un horizon fini ;
- $\lim_{h \rightarrow +\infty} E\left(\frac{1}{h} \sum_{t=0}^h r_t\right)$  : récompense moyenne à long terme ;
- $E\left(\sum_{t=0}^{\infty} \gamma^t r_t\right)$  ( $0 \leq \gamma < 1$ ) : récompense amortie à horizon infini.

Le dernier critère est courant et c'est celui que nous adoptons dans la suite. La valeur de  $\gamma$  permet de définir l'importance que l'on donne au futur. Quand  $\gamma = 0$  nous sommes face à un agent « pessimiste » qui ne cherche qu'à optimiser son gain immédiat. À l'opposé si  $\gamma \rightarrow 1$ , l'agent est « optimiste » puisqu'il tient de plus en plus sérieusement compte du futur lointain.

Lorsqu'une politique et un critère sont déterminés, deux fonctions centrales peuvent être définies :

- $V^\pi : S \rightarrow \mathbb{R}$  : c'est la fonction de valeur des états ;  $V^\pi(s)$  représente le gain (selon le critère adopté) engendré par l'agent s'il démarre à l'état  $s$  et applique ensuite la politique  $\pi$  ad infinitum.
- $Q^\pi : S \times A \rightarrow \mathbb{R}$  : c'est la fonction de valeur des états-actions ;  $Q^\pi(s, a)$  représente le gain engendré par l'agent s'il démarre à l'état  $s$  et commence par effectuer l'action  $a$ , avant d'appliquer ensuite la politique  $\pi$  ad infinitum.

Les deux fonctions sont intimement liées. On a toujours  $V^\pi(s) = Q^\pi(s, \pi(s))$  et, dans le cas du gain amorti à horizon infini, on peut également écrire que :

$$Q^\pi(s, a) = \sum_{s' \in S} [R(s, a, s') + \gamma V^\pi(s')] T(s, a, s').$$

Cette dernière relation montre que la fonction  $V^\pi$  vérifie une relation de récurrence appelée équation de **Bellman** :

$$V^\pi(s) = \sum_{s' \in S} [R(s, \pi(s), s') + \gamma V^\pi(s')] T(s, \pi(s), s').$$

## 7 Problèmes possibles

- Planification : il s'agit, étant donné un **MDP**  $\{S, A, T, R\}$ , de trouver quelle est la politique  $\pi$  qui maximise la récompense.
- Améliorer une politique connue : soit une politique  $\pi_0$ , on souhaite trouver une meilleure politique.

Ce problème est notamment au cœur des algorithmes de recherche de la politique optimale.

- Apprentissage d'une politique sans connaître le modèle :

- à partir de traces d'exécution : c'est le problème de l'apprentissage par renforcement hors ligne.
- au cours d'expériences sur le modèle, on parle alors d'apprentissage par renforcement en ligne.

## 8 Algorithmes

Une politique étant fixée, l'équation de Bellman peut se résoudre d'au moins deux manières, permettant donc de déterminer les valeurs de  $V^\pi$ , et par suite, celles de  $Q^\pi$  également.

- on peut déjà remarquer que, dans le cas où le nombre d'états  $n$  est fini, l'équation de Bellman cache en fait un système linéaire de  $n$  équations à  $n$  inconnues.

On peut donc le résoudre, une fois traduit en une équation matricielle, par une technique telle que le **pivot de Gauss**.

- on peut également remarquer qu'en posant

$$K(f)(s) = \sum_{s' \in S} [R(s, \pi(s), s') + \gamma f(s')] T(s, \pi(s), s'),$$

on définit un opérateur  $K$ , appelé opérateur de Bellman, pour lequel  $V^\pi$  est un point fixe. On peut montrer que  $K$  est une **contraction**, ce qui garantit d'une part l'existence d'un unique **point fixe**, et d'autre part que la suite récurrence  $V_{n+1} = K(V_n)$  converge vers ce point fixe exponentiellement vite.

### 8.1 Équations d'optimalité de Bellman

Le but de l'agent est de trouver la politique optimale  $\pi^*$  qui lui permet de maximiser son gain, c'est-à-dire celle qui vérifie, pour tout état  $s \in S$ ,  $V^{\pi^*}(s) \geq V^\pi(s)$  quelle que soit l'autre politique  $\pi$ . On peut montrer que la fonction de valeurs optimale  $V^*$  vérifie l'équation d'optimalité de Bellman :

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} [R(s, a, s') + \gamma V^*(s')] T(s, a, s').$$

De manière analogue, la fonction  $Q$  vérifie elle aussi une équation d'optimalité :

$$Q^*(s, a) = \sum_{s' \in S} [R(s, a, s') + \gamma \max_{a' \in A} Q^*(s', a')] T(s, a, s').$$

## 8.2 Résolution des équations d'optimalité de Bellman

Les équations d'optimalité de Bellman ne sont pas linéaires, il faut donc abandonner l'idée de les résoudre algébriquement. En revanche, l'opérateur de Bellman  $K^*$  défini par

$$K^*(f)(s) = \max_{a \in A} \sum_{s' \in S} [R(s, a, s') + \gamma f(s')] T(s, a, s'),$$

définit encore une contraction dont  $V^*$  est un point fixe. La fonction de valeurs optimale peut donc à nouveau s'approcher par un processus itératif à convergence exponentielle.

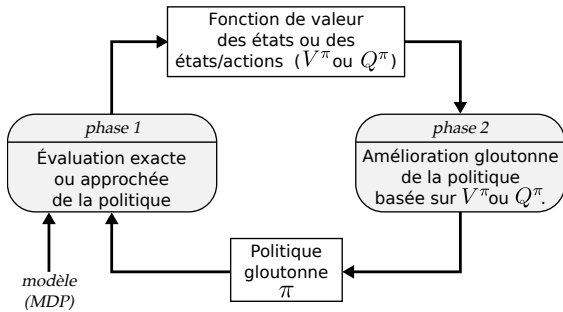
## 8.3 Déterminer la politique optimale : algorithme d'itération sur la valeur (VI)

La méthode itérative que nous venons de voir pour les équations d'optimalité de Bellman fournit un premier algorithme, appelé *itération sur la valeur* (VI : Value-Iteration) permettant de déterminer  $\pi^*$ . Il suffit en effet de déterminer  $V^*$  avec une précision donnée, et on peut en déduire la politique optimale par :

$$\pi(s) = \arg \max_{a \in A} Q^*(s, a) = \arg \max_{a \in A} \sum_{s' \in S} [R(s, a, s') + \gamma V^*(s')] T(s, a, s').$$

Une difficulté dans cet algorithme est de déterminer la précision avec laquelle calculer  $V^*$  de manière à être sûr d'en déduire effectivement la politique optimale.

## 8.4 Déterminer la politique optimale : algorithme d'itération sur la politique (PI)



Description générale d'un algorithme d'itération de la politique

Un autre algorithme, appelé *itération de la politique* (PI : Policy-Iteration) essaye d'obtenir la politique optimale sans nécessairement calculer « jusqu'au bout » les valeurs de  $V^*$ . L'idée est de partir d'une politique quelconque  $\pi_0$ , puis d'alterner une phase d'évaluation, dans laquelle

la fonction  $V^{\pi_n}$  est déterminée (avec une des techniques vues plus haut), et une phase d'amélioration, où l'on définit la politique suivante  $\pi_{n+1}$  par :

$$\pi_{n+1}(s) = \arg \max_{a \in A} \sum_{s' \in S} [R(s, a, s') + \gamma V^{\pi_n}(s')] T(s, a, s').$$

Cet algorithme prend fin lorsqu'aucune évolution de la politique n'est observée, ie, lorsque  $\pi_{n+1}(s) = \pi_n(s)$  pour tout  $s$ .

Si dans l'algorithme précédent l'on utilise une méthode itérative pour évaluer  $V^{\pi}$ , alors se pose la question de savoir à quelle précision s'arrêter. Ce problème n'en est en réalité pas un, car on peut montrer que même si l'on tronque l'évaluation de  $V^{\pi}$ , l'algorithme converge tout de même vers l'optimal. À l'extrême, c'est-à-dire lorsqu'une seule itération est utilisée pour évaluer  $V^{\pi}$ , et après avoir réuni en une seule étape de calcul la phase d'amélioration et la phase d'évaluation, on retombe sur l'algorithme VI.

L'algorithme PI peut également se formuler dans les termes de la fonction d'états-actions  $Q$  plutôt que  $V$ . On voit donc qu'un grand nombre de variantes peuvent être imaginées, mais tournant toutes autour d'un même principe général qui est schématisé à la figure ci-contre.

## 9 Articles connexes

Voir aussi :



- Les chaînes de Markov, dont dérivent les MDP,
- Les processus de décision markoviens partiellement observables (POMDP), qui permettent de modéliser l'incertitude sur l'état dans lequel on se trouve
- Le calcul stochastique, qui est à la base les modèles stochastiques
- L'apprentissage par renforcement, méthode permettant de résoudre des processus de décision markoviens.
- Les métaheuristiques, méthodes utilisant parfois des processus markoviens.

## 10 Bibliographie

- (Putterman 1994) Putterman M. L., Markov Decision Processes. Discrete stochastic dynamic programming. Wiley-Interscience, New York 1994, 2005.
- (Sutton et Barto 1998) Sutton R. S. et Barto A.G. Reinforcement Learning : An introduction. MIT Press, Cambridge, MA, 1998.

## 11 Références

- [1] (en) Xianping Guo, Onesimo Hernandez-Lerma, *Continuous-Time Markov Decision Processes : Theory and Applications*, Springer-Verlag Berlin Heidelberg, 2009, (ISBN 978-3-642-02546-4)

-  Portail des probabilités et de la statistique
-  Portail de l'informatique théorique

## 12 Sources, contributeurs et licences du texte et de l'image

### 12.1 Texte

- **Processus de décision markovien** *Source* : [https://fr.wikipedia.org/wiki/Processus\\_de\\_d%C3%A9cision\\_markovien?oldid=121112111](https://fr.wikipedia.org/wiki/Processus_de_d%C3%A9cision_markovien?oldid=121112111) *Contributeurs* : Nojhan, Archeos, Bob08, KhunMKham, MistWiz, Jmini, Sylenius, Ji-Elle, PierreSelim, JAnDbot, Salebot, Gdupont, Jean-Louis Lascoux, Metaxal, Ptbotgourou, Yannis1962, DumZiBoT, WikiCleanerBot, Luckas-bot, ArthurBot, Lomita, TobeBot, Dinamik-bot, Aurelien.poisson, BendelacBOT, Roll-Morton, Naereen, Addbot, Agerussi, NathanBerah et Anonyme : 8

### 12.2 Images

- **Fichier:Logo\_proba\_4.svg** *Source* : [https://upload.wikimedia.org/wikipedia/commons/f/f7/Logo\\_proba\\_4.svg](https://upload.wikimedia.org/wikipedia/commons/f/f7/Logo_proba_4.svg) *Licence* : CC BY-SA 3.0 *Contributeurs* : Travail personnel *Artiste d'origine* : Ipipipourax
- **Fichier:Markov\_Decision\_Process\_example.png** *Source* : [https://upload.wikimedia.org/wikipedia/commons/2/21/Markov\\_Decision\\_Process\\_example.png](https://upload.wikimedia.org/wikipedia/commons/2/21/Markov_Decision_Process_example.png) *Licence* : Public domain *Contributeurs* : Travail personnel *Artiste d'origine* : MistWiz
- **Fichier:Max-cut.svg** *Source* : <https://upload.wikimedia.org/wikipedia/commons/c/cf/Max-cut.svg> *Licence* : CC BY-SA 3.0 *Contributeurs* : Travail personnel *Artiste d'origine* : Miym
- **Fichier:Policy-iteration.svg** *Source* : <https://upload.wikimedia.org/wikipedia/commons/6/65/Policy-iteration.svg> *Licence* : CC BY-SA 4.0 *Contributeurs* : Travail personnel *Artiste d'origine* : Agerussi
- **Fichier :\_Max-cut.svg** *Source* : <https://upload.wikimedia.org/wikipedia/commons/c/cf/Max-cut.svg> *Licence* : CC BY-SA 3.0 *Contributeurs* : Travail personnel *Artiste d'origine* : Miym

### 12.3 Licence du contenu

- Creative Commons Attribution-Share Alike 3.0