Apprentissage par renforcement

Stéphane Airiau

Université Paris Dauphine

Un type d'apprentissage qui est basé sur l'**interaction avec son environnement**.
⇨ dans ce cours, on va étudier une approche computationnelle pour apprendre de son interaction.
Le but de l'apprentissage par renforcement est de savoir quoi faire dans un contexte donné pour maximiser un signal récompense.

- les actions peuvent avoir une influence sur les contextes futurs
- l'apprenti doit découvrir par lui-même les actions qui donnent les plus hautes récompenses en les essayant !
- ces choix peuvent avoir des conséquences sur la récompense directement obtenue, mais aussi sur la situation suivante.

Représenter l’environnement

- l’état du monde n’est pas toujours connu de l’agent
  - ce que l’agent observe n’est pas suffisant pour déterminer l’état exact
    - ex un robot ne connait pas exactement sa position. Sa camera ne lui permet pas de voir le monde avec suffisemment de résolution
    - un agent qui joue au poker ne connait pas les cartes de son adversaire !
  - en plus, il y a peut être de l’information non pertinente !
- l’agent a une structure pour représenter l’état du monde
  (et si possible une représentation succincte !)
  ⇒ de toute façon pas la réalité !

Représenter l'environnement

Deux types de models :

- **Observation totale** : l'agent observe directement l'état de l'environnement
  ➡ pas d'incertitude sur l'état du monde
  ➡ processus decisonnel de Markov (PDM)
  C'est ce qu'on va étudier
- **Observation partielle** : l'agent est uncertain, il n'est pas sûr quel est le véritable état du monde
  il connait l'ensemble des états du monde et peut avoir une estimation de la probabilité de se trouver dans chaque état.
  PDM partiellement Observable

Hypothèse de Markov

Si on exécute l'action a dans l'état s :

- On obtient une récompense r
- On arrive dans un état s'

En principe, r et s' peuvent dépendre de tout l'historique !

Definition (Etat de Markov)

Un état S_t est dit de **Markov** ssi

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, \ldots, S_t]$$

- "Etant donné le présent, le futur est indépendant du passé"
- Une fois que l'état est connu, on peut effacer son historique !
- *ex :* aux échecs, l'état du jeu ne dépend pas de l'historique des coups !

Les composants d'un agent

un agent peu posséder (ou non) un de ces composants :

- **politique** la fonction qui décide le comportement de l'agent
  ex : dans tel état, l'agent effectue telle action
- le signal récompense : c'est ce qui défini le problème d'apprentissage par renforcement.
  ➾ le but est de maximiser ce signal sur le long terme. Le signal est <u>généralement</u> une fonction stochastique de l'état de l'environnement.
- **fonction de valeurs** : la fonction mesure la qualité de chaque état et/ou action <u>sur le long terme</u>
- **modèle** : la représentation que possède l'agent de son environnement.
  On verra des algorithmes qui nécessitent un modèle ("model-based") mais aussi des algorithmes qui vont bâtir un modèle en apprenant ("model-free").

Les composants d'un agent : politique

C'est ce qui gouverne le comportement de l'agent

On nomme

- A l'ensemble des actions
- S l'ensemble des états

La fonction associe à chaque état :

- **politique déterministe** : une action
$$\pi : S \mapsto A$$
- **politique stochastique** : une distribution de probabilité sur les actions possibles
$$\pi : S \mapsto \Delta(A)$$

où $\Delta(N)$ désigne une distribution de probabilité sur l'ensemble (fini) N.
$p \in \Delta(N)$ ssi $\forall i \in N$, $p(i) \in [0,1]$ et $\sum_{i \in N} p(i) = 1$

Les composants d'un agent : fonction de valeurs

- La fonction estime les récompenses que l'agent va obtenir dans le futur à partir de cet état.
- Elle va nous servir à estimer quel état est prometteur ou non
  ➾ elle va nous aider à choisir l'action de l'agent !

Les composants d'un agent : modèles

- Un modèle prédit le comportement de l'environnement.
  un agent n'a donc pas toujours accès à un modèle !
- le modèle est souvent stochastique
  le problème ne serait peut-être pas intéressant sinon

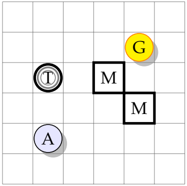
- S_t est l'état du monde à l'instant t
- A_t est l'action choisit par l'agent à l'instant t
- **modèle de transition**

$$T^a_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$

- **modèle de récompense**

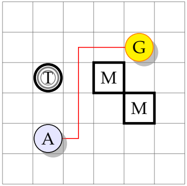
$$R^a_s = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$$

Exemple : contrôle optimal d'un robot



- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

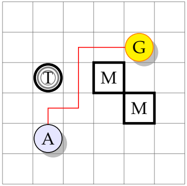
Exemple : contrôle optimal d'un robot



Gains : 95

- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

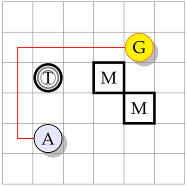
Exemple : contrôle optimal d'un robot



Gains : 95

- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

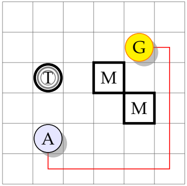
Exemple : contrôle optimal d'un robot



Gains : 93

- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

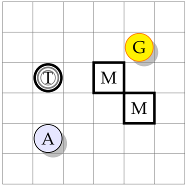
Exemple : contrôle optimal d'un robot



Gains : 91

- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

Exemple : contrôle optimal d'un robot



- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au nord, sud, est et ouest
- Si l'agent arrive dans l'état contenant le pot d'or, il gagne 100, sinon, L'arrivée dans un autre état lui coûte 1 (i.e. la récompense est -1)

Exemple : une politique

→	→	→	→	↓	↓
→	→	→	→	G	←
↑	T	↑	M	↑	↑
↑	→	↑	←	M	↑
↑	→	↑	←	→	↑
↑	→	↑	←	→	↑

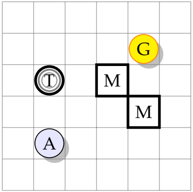
la politique doit bien dire quelle action effectuer dans chaque état !

Exemple : la fonction de valeurs

96	97	98	99	100	99
97	98	99	100	G	100
96	T	98	M	100	99
95	96	97	96	M	98
94	95	96	95	96	97
93	94	95	94	95	96

On peut donc avoir plusieurs politiques qui correspondent à la même fonction de valeurs.

Exemple : contrôle optimal d'un robot



Pour compliquer : environnement n'est pas déterministe
chaque action peut échouer et le robot peut se retrouver dans une case adjaçante
ex : si l'action est $\rightarrow$, le résultat peut amener l'agent

- effectivement dans la case suivante à droite avec une probabilité 0.8
- dans la case suivante au dessus avec une probabilité 0.05
- dans la case suivante au dessous avec une probabilité 0.05
- dans la case de départ avec une probabilité 0.1

⇨ modèle de transition T connu ou non de l'agent.

Comment calculer la fonction de valeurs ou une politique optimale ?

Quelle fonction optimise-t-on ?

- pour des tâches épisodiques
  - il y a des états terminaux et initiaux
  - on repart dans un état initial une fois qu'on atteint un état terminal

  ➡ maximise le cumul des récompenses sur *un épisode*

$$R_T = r_1 + r_2 + \cdots + r_T$$

- pour des tâches en continue

  ➡maximise une récompense "escomptée" $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

  γ est le taux d'escompte
  - $\gamma = 0$ l'agent est "myope" : il n'est intéressé que par la récompense immédiate
  - $0 < \gamma < 1$ quand $\{r_t, t \in \mathbb{N}\}$ est bornée, alors R_T est bien définie.
    ➡l'agent cherche un équilibre entre la récompense immédiate et celle qu'il obtiendra dans le futur

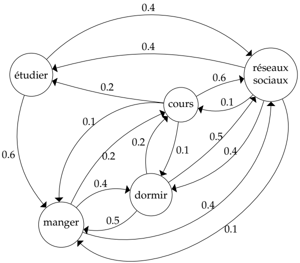
Focus pour le cours

- problèmes itératifs en continue.
- objectif : maximiser la somme "escomptée" $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$
  - Pour éviter une récompense infinie si on tombe dans des cycles
  - Le futur reste uncertain ! Bon compromis entre court et long terme
  - Tendance naturelle vers le court terme
  - Mathématiquement, c'est quand même pratique !
- la fonction de transition est stochastique
- la fonction de récompense est connue

Comment trouver la meilleure politique ? / Comment calculer la fonction de valeur ?

Un pas en arrière : chaînes de Markov

- pas d'actions, juste des transitions.
- on connait la probabilité d'aller d'un état à un état voisin



Matrice de transition T

	étudier	manger	dormir	cours	réseaux sociaux
étudier		0.6			0.4
manger			0.4	0.2	0.4
dormir		0.5		0.2	0.3
cours	0.2	0.1	0.1		0.6
réseaux sociaux	0.4	0.1	0.4	0.1	

Un pas en arrière : chaînes de Markov

$$T = \begin{pmatrix} 0 & 0.6 & 0 & 0 & 0.4 \\ 0 & 0 & 0.4 & 0.2 & 0.4 \\ 0 & 0.5 & 0 & 0.2 & 0.3 \\ 0.2 & 0.1 & 0.1 & 0 & 0.6 \\ 0.4 & 0.1 & 0.4 & 0.1 & 0 \end{pmatrix}$$

- Evidemment, la somme des éléments d'une ligne est 1.
- on peut donner la probabilité d'être un état de départ $q(i)$: probabilité que i soit état de départ.
- Que représente qT ? qT^2, … qT^n, $\lim_{n \to +\infty} qT^n$ [a] ?

---

a. si elle existe !

Un pas en arrière : chaînes de Markov

$$T = \begin{pmatrix} 0 & 0.6 & 0 & 0 & 0.4 \\ 0 & 0 & 0.4 & 0.2 & 0.4 \\ 0 & 0.5 & 0 & 0.2 & 0.3 \\ 0.2 & 0.1 & 0.1 & 0 & 0.6 \\ 0.4 & 0.1 & 0.4 & 0.1 & 0 \end{pmatrix}$$

- Evidemment, la somme des éléments d'une ligne est 1.
- on peut donner la probabilité d'être un état de départ
  $q(i)$: probabilité que i soit état de départ.
- Que représente qT ? qT^2, … qT^n, $\lim_{n\to+\infty} qT^n$ [a] ?
- L'entrée (i,j) de T^n nous donne la probabilité d'arriver dans l'état j en partant de l'état i après n étapes

a. si elle existe !

Un pas en arrière : chaînes de Markov

$$T = \begin{pmatrix} 0 & 0.6 & 0 & 0 & 0.4 \\ 0 & 0 & 0.4 & 0.2 & 0.4 \\ 0 & 0.5 & 0 & 0.2 & 0.3 \\ 0.2 & 0.1 & 0.1 & 0 & 0.6 \\ 0.4 & 0.1 & 0.4 & 0.1 & 0 \end{pmatrix}$$

- Evidemment, la somme des éléments d'une ligne est 1.
- on peut donner la probabilité d'être un état de départ $q(i)$: probabilité que i soit état de départ.
- Que représente qT ? $qT^2, \ldots \, qT^n, \lim_{n \to +\infty} qT^n$ [a] ?
- L'entrée (i,j) de T^n nous donne la probabilité d'arriver dans l'état j en partant de l'état i après n étapes
- L'entrée i de $q \cdot T^n$ nous donne donc la probabilité de se trouver dans l'état i après n étapes en partant de q

a. si elle existe !

Un pas en arrière : chaînes de Markov

$$T = \begin{pmatrix} 0 & 0.6 & 0 & 0 & 0.4 \\ 0 & 0 & 0.4 & 0.2 & 0.4 \\ 0 & 0.5 & 0 & 0.2 & 0.3 \\ 0.2 & 0.1 & 0.1 & 0 & 0.6 \\ 0.4 & 0.1 & 0.4 & 0.1 & 0 \end{pmatrix}$$

- Evidemment, la somme des éléments d'une ligne est 1.
- on peut donner la probabilité d'être un état de départ
  $q(i)$: probabilité que i soit état de départ.
- Que représente qT ? qT^2, … qT^n, $\lim_{n\to+\infty} qT^n$ [a] ?
- L'entrée (i,j) de T^n nous donne la probabilité d'arriver dans l'état j en partant de l'état i après n étapes
- L'entrée i de $q \cdot T^n$ nous donne donc la probabilité de se trouver dans l'état i après n étapes en partant de q
- $T^n(i,j)$ agrège toutes les possibilités pour rejoindre l'état j depuis l'état i.

---

a. si elle existe !

Un pas en arrière : chaînes de Markov avec récompenses

- On ajoute aux chaînes de Markov une fonction de récompenses R
- $R(s) = \mathbb{E}\left[R_{t+1} | S_t = t\right]$ est l'espérance[a] de récompense lorsqu'on quitte l'état s
- γ représente la valeur présente d'obtenir une récompense dans le futur
- La valeur d'obtenir la récompense R après $k+1$ étape est $\gamma^k R$

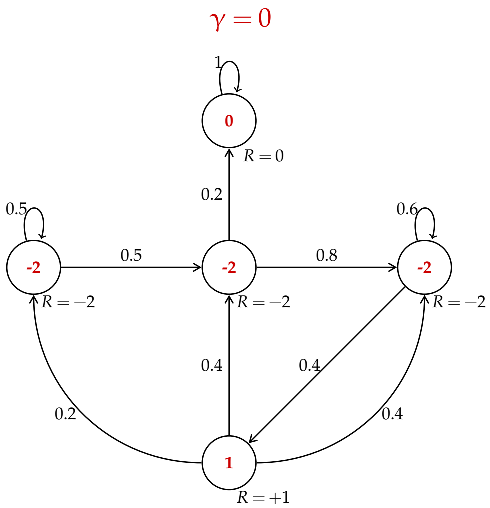
➩ on veut optimiser $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

La fonction de valeurs donne la valeur à long terme de l'état s.

$$v(s) = \mathbb{E}\left[G_t \mid S_t = s\right]$$

a. on peut aussi avoir une version deterministe

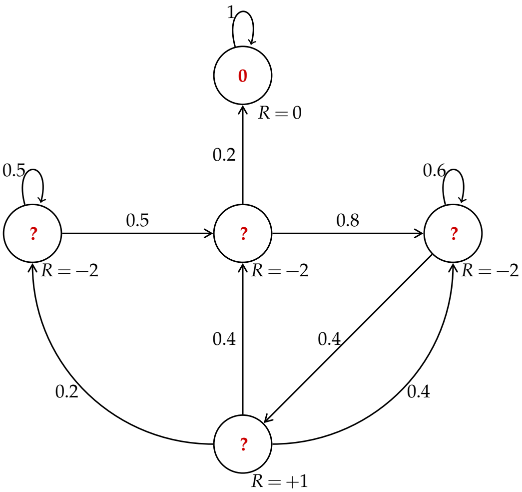
Un pas en arrière : chaînes de Markov avec récompenses



La récompense indiquée est celle quand on quitte l'état.

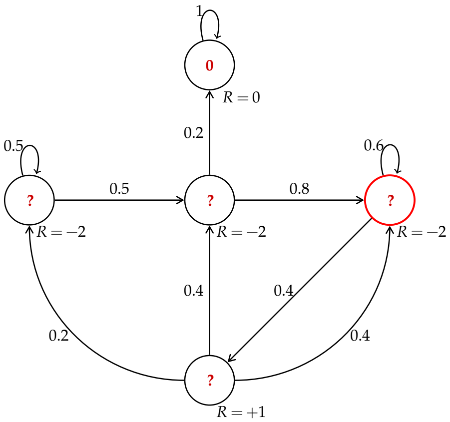
Un pas en arrière : chaînes de Markov avec récompenses

$\gamma = 0.9$??



Un pas en arrière : chaînes de Markov avec récompenses

$\gamma = 0.9$??



$1^{ière}$ étape :

$-2 + \gamma(0.4 \cdot 1 + 0.6 \cdot -2)$

$= -2.72$

2^{nde} étape :

il faut prendre en compte la valeur obtenue par les autres noeuds en une étape !

Equation de Bellman

On peut décomposer la fonction de valeurs

- la récompense immédiate
- La somme escomptée de l'état suivant $\gamma \cdot v(s_{t+1})$

$$
\begin{aligned}
v(s) &= \mathbb{E}\left[G_t \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + \ldots \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma\left(R_{t+2} + \gamma R_{t+2} + \ldots\right) \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right] \\
&= R_s + \gamma \sum_{s' \in S} T_{ss'} v(s')
\end{aligned}
$$

Equation de Bellman : représentation matricielle

- R est le vecteur des récompenses
- T est la matrice de transition
- v est le vecteur de la fonction de valeurs

On obtient donc l'équation

$$v = R + \gamma \cdot Tv$$

Dans notre exemple, cela donne

$$\begin{pmatrix} \mathrm{v}(1) \\ \mathrm{v}(2) \\ \mathrm{v}(3) \\ \mathrm{v}(4) \\ \mathrm{v}(5) \end{pmatrix} = \begin{pmatrix} -2 \\ -2 \\ -2 \\ 1 \\ 0 \end{pmatrix} + \gamma \begin{pmatrix} 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.8 & 0 & 0.2 \\ 0 & 0 & 0.6 & 0.4 & 0 \\ 0.2 & 0.4 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \mathrm{v}(1) \\ \mathrm{v}(2) \\ \mathrm{v}(3) \\ \mathrm{v}(4) \\ \mathrm{v}(5) \end{pmatrix}$$

On doit résoudre un système linéaire !

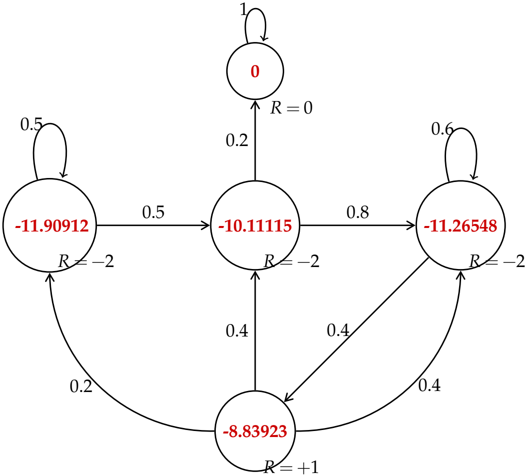
$$\begin{aligned} v &= R + \gamma \cdot T v \\ (I - \gamma T) v &= R \\ v &= (I - \gamma T)^{-1} R \end{aligned}$$

$$v = \begin{pmatrix} 1-0.5\gamma & -0.5\gamma & 0 & 0 & 0 \\ 0 & 1 & -0.8\gamma & 0 & -0.2\gamma \\ 0 & 0 & 1-0.6\gamma & -0.4\gamma & 0 \\ -0.2\gamma & -0.4\gamma & -0.4\gamma & 1 & 0 \\ 0 & 0 & 0 & 0 & 1-\gamma \end{pmatrix}^{-1} \cdot \begin{pmatrix} \text{-2} \\ \text{-2} \\ \text{-2} \\ 1 \\ 0 \end{pmatrix}$$

- Résolution exacte d'un systemen $\mathcal{O}(n^3)$ pour n états
- Résolution directe si n n'est pas très grand (pivot de Gauss)
- méthodes itératives si n est grand

Fonction de valeurs : Solution

$\gamma = 0.9$



Processus décisionnel de Markov

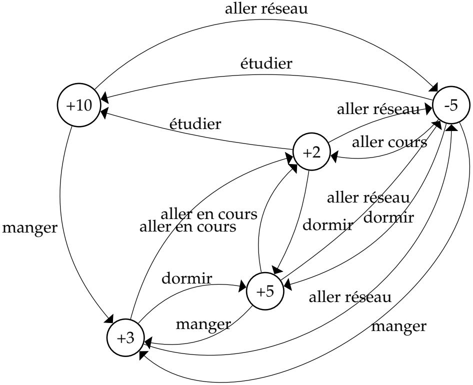
On ajoute des actions au modèle précédent.

Definition (Processus décisionnel de Markov)

Un *Processus décisionnel de Markov* est un tuple $\langle S, A, T, R, \gamma \rangle$ où

- S est un ensemble fini d'états
- A est un ensemble fini d'actions
- T est une matrice de transition
  $T^a_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$
- R est le vecteur de récompenses
  $R^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$
- γ est le taux d'escompte

Exemple



- la fonction de transition reste probabiliste :
  (non représenté)
  parfois vous voulez aller étudier, mais vous finissez sur facebook !
- la valeur des noeuds est ici la récompense (immédiate) R

Rappel : politique

La politique d'un agent définit son comportement

Definition (Politique)

Une *politique* π est une distribution de probabilité sur les actions étant donné un état, i.e.

$$\pi(a \mid s) = \mathbb{P}[A_t = a \mid S_t = s]$$

- la politique ne dépend pas du temps !
- la politique ne dépend que de l'état actuel !

Rappel : politique

Notez que si on a

- un PDM $\langle S, A, T, R, \gamma \rangle$
- une politique π

➪ on peut maintenant se ramener au cas précédents !

on obtient une chaîne de Markov avec récompenses $\langle S, T^\pi, R^\pi, \gamma \rangle$

- $T^\pi_{ss'} = \sum_{a \in A} \pi(a|s) \cdot T^a_{ss'}$
- $R^\pi_s = \sum_{a \in A} \pi(a|s) R^a_s$

On peut définir deux fonctions de valeurs

Definition (fonction de valeurs pour les états)

La *fonction de valeurs pour les états* $v_\pi(s)$ d'un PDM est la valeur espérée de gains en partant dans l'état s et en poursuivant la politique π.

Definition (fonction de valeurs pour les paires (état, action))

La *fonction de valeurs pour les paires état-actions* $q_\pi(s,a)$ d'un PDM est la valeur espérée de gains en partant dans l'état s, en effectuant l'action a *puis* et en poursuivant la politique π.

Equation de Bellman

On peut appliquer la même idée que précédemment : la fonction de valeurs pour les états peut être décomposée en deux parties

- la récompense immédiate
- la valeur escomptée de l'état suivant

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s\right]$$

De même pour la fonction de valeurs pour les actions :

$$q_\pi(s,a) = \mathbb{E}_\pi \left[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a\right]$$

Equation de Bellman : pour les états

Dans l'état s

on tire notre action avec la politique $\pi(s)$

pour chaque action, on choisit l'action a avec la probabilité $\pi(a|s)$,

on va effectuer l'action a puis continuer avec π dans l'état suivant

➡ on peut utiliser q_π !

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s\right] \\
&= \sum_{a \in A} \pi(a|s) q_\pi(s,a)
\end{aligned}
$$

Equation de Bellman : pour les actions

Similairement pour la fonction de valeurs pour les actions

- le modèle de récompense nous donne la récompense pour avoir effectué l'action a dans l'état s.
- le modèle de transition nous donne l'état suivant s'

➯ dans ce nouvel état s', on peut utiliser v_π !

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a\right] \\
&= R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_\pi(s')
\end{aligned}
$$

Equation de Bellman

On a établi :

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s,a)$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_\pi(s')$$

Ensemble on obtient :

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_\pi(s') \right)$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a \sum_{a' \in A} \pi(a'|s') q_\pi(s',a')$$

Equations de Bellman : formes concises

$$v_\pi = R^\pi + \gamma T^\pi v_\pi$$

avec la solution

$$v_\pi = (I - \gamma T^\pi)^{-1} R^\pi$$

Etant donnée une politique, on peut trouver les fonctions de valeurs.

Notez bien que tout cela est possible car on a fixé une politique π, la fonction de valeurs trouvée est seulement pour cette politique et donne la valeur à long terme si on suit π.

Optimal Value Functions

Definition (Fonction de valeurs optimale)

La fonction optimale de valeurs $v^\star$est la fonction qui a la valeur maximale pour toutes les politiques

$$v^\star(s) = \max_\pi v_\pi(s), \forall s \in S$$

Pour résoudre notre PDM, on aimerait trouver $\pi^\star$ ou $q^\star$

Optimal Value Functions

On peut définir un ordre partiel $\succeq$ sur les différentes politiques :
$\pi \succeq \pi'$ iff $\forall s \in S\ v_\pi(s) \geqslant v_{\pi'}(s)$

Theorem

- Il existe une politique optimale $\pi^\star$ qui est meilleure ou égale à toutes les autres politiques π, i.e. $\forall \pi\ \pi^\star \geqslant \pi$
- toutes les politiques optimales partagent la même fonction d'évaluation des états
  ⇨ fonction optimale d'évaluation des états $v^\star = \max_\pi v_\pi(s)$
- toutes les politiques optimales partagent la même fonction de valeurs des actions $q^\star(s,a) = \max_\pi q_\pi(s,a)\ \forall s,a$

plusieurs politiques optimales peuvent exister. Le premier point suggère qu'il serait possible que deux politiques optimales puissent être incomparables (pour un état, l'une est meilleure que l'autre, mais pour l'autre ce serait l'inverse). Mais le second point nous indique que toutes les politiques optimales partagent la même fonction de valeurs.

Trouver une politique optimale

Tâche facile si on connait $q^\star(s,a)$:
Dans un état s, la politique déterministe qui choisit l'action qui a la plus grande valeur de $q^\star(s,a)$ est optimale !

$$\pi^\star(a|s) = \begin{cases} 1 \text{ si } a = argmax_{a \in A} q^\star(s,a) \\ 0 \text{ sinon} \end{cases}$$

- ➯ il y a toujours une politique deterministe qui est optimale !
- reste à trouver $q^\star(s,a)$

Equation de Bellman pour $v^\star$

$$v^\star(s) = \max_{a \in A} q^\star(s,a)$$

$$q^\star(s,a) = R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v^\star(s')$$

donc

$$v^\star(s) = \max_{a \in A} R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v^\star(s')$$

et

$$q^\star(s,a) = R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a \max_{a' \in A} q^\star(s',a')$$

Résoudre l'équation de Bellman

Pour des PDMs finis, l'équation de Bellman a une **unique solution** qui est indépendente de la politique.

➡ Système de n équations avec n inconnues, mais **non-linéaires** !

➡ différentes méthodes pour trouver $V^\star$

- dynamic programming (policy iteration, value iteration)
- utilisation de méthode de Monte Carlo pour trouver des approximations.
- Apprentissage avec la méthode "temporal difference" $\rightarrow$ combine la programmation dynamique avec une méthode de Monte Carlo (Sarsa, Q-learning)

Evaluation d'une politique

Sous sa forme concise, on a pour une politique π

$$v_\pi = R^\pi + \gamma T^\pi v_\pi$$

- méthode pour résoudre directement le système d'équations
- méthodes itératives

On peut utiliser l'équation de Bellman comme une règle de mise à jour :

$$V_{k+1} = R^\pi + \gamma T^\pi V_k$$

$$v_{k+1}(s) \leftarrow \sum_{a \in A} \pi(a \mid s) \left(R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_k(s') \right)$$

Theorem

La séquence $\{v_k\}_{k \in \mathbb{N}}$ converge vers v^π.

Evaluation d'une politique

Sous sa forme concise, on a pour une politique π

$$v_\pi = R^\pi + \gamma T^\pi v_\pi$$

- méthode pour résoudre directement le système d'équations
- **méthodes itératives**

On peut utiliser l'équation de Bellman comme une règle de mise à jour :

$$V_{k+1} = R^\pi + \gamma T^\pi V_k$$

$$v_{k+1}(s) \leftarrow \sum_{a \in A} \pi(a \mid s) \left(R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_k(s') \right)$$

Theorem

La séquence $\{v_k\}_{k\in\mathbb{N}}$ converge vers v^π.

Convergence plus rapide

Pour réaliser l'algorithme :

- avoir deux vecteur v_{old} et v_{new}
- calculer completement v_{new} à partir de v_{old}

⇨ "full back up"

On peut aussi n'utiliser qu'*un seul* vecteur

- on remplace directement l'ancienne entrée par la nouvelle
- le vecteur v contient à la fois des nouvelles et des anciennes valeurs

⇨ on utilise les nouvelles valeurs au plus vite
convergence toujours garantie et plus rapide
l'ordre de mise à jour joue un rôle sur la vitesse de convergence.

Critère d'arrêt de l'algorithme

- garantie de convergence à la limite
- en pratique, on peut arrêter avant
  par exemple : $\max_{s \in S} |v_{k+1}(s) - v_k(s)| < \epsilon$ pour une valeur de ϵ donnée.

Améliorer une politique

On peut essayer d'améliorer la politique en se comportant de manière "gloutonne"

Une fois v_π évaluée : on peut calculer $q^\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_\pi(s')$

- si $q^\pi(s,a) > v_\pi(s)$: on a trouvé une amélioration ![a]
- ⇨ on peut regarder tous les états $s \in S$ et mettre à jour la politique $\pi'(s) = arg\max_{a \in A} q_\pi(s,a)$

Si aucune amélioration n'est trouvée, on a donc $v_\pi = v_{\pi'}$
⇨ $v_{\pi'} = \max_{a \in A} R_s^a + \gamma \sum_{s' \in S} T_{ss'}^a v_{\pi'}(s')$
On reconnait là l'équation de Bellman pour la fonction de valeurs *optimale*
On a donc trouvé $v^\star = v_{\pi'}$!

a. il faut une petite démonstration sur ce point

Itération de politique

L'idée est donc d'alterner

l'évaluation d'une politique

amélioration de la politique

*jusqu'à ce qu'*on converge vers une politique qui sera la politique optimale.

Pour les politiques deterministes, il y a un nombre fini de politiques, on va converger en un nombre fini d'itérations.

Variantes : quand arrêter l'évaluation ?

- convergence à un ϵ près
- après k itérations (k a une petite valeur)
- pourquoi pas après chaque itération ?

Value Iteration (dynamic programming)

for each $s \in S$
  $V(s) \leftarrow 0$

repeat
  $\Delta \leftarrow 0$
  for each $s \in S$
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T^a_{s \to s'} \left[R^a_{s \to s'} + \gamma V(s') \right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \epsilon$

- On n'a pas de politique explicite
- On a un théorème de convergence

Pas toujours utile en pratique :

- demande de connaître la dynamique de l'environnement (les probabilités $T^a_{s \to s'}$)
- exige de larges ressources de calcul.