MDP – Résolution par programmation dynamique

MDP - Itération sur les valeurs – itération sur les politiques

Stéphane Airiau

Université Paris Dauphine

Blackjack

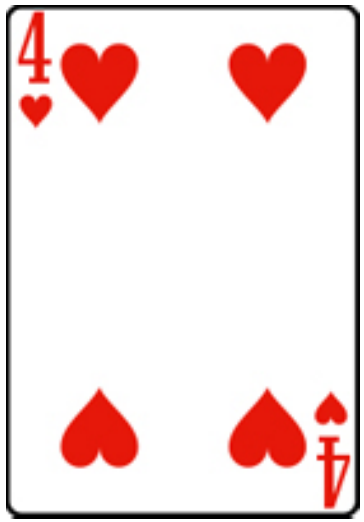
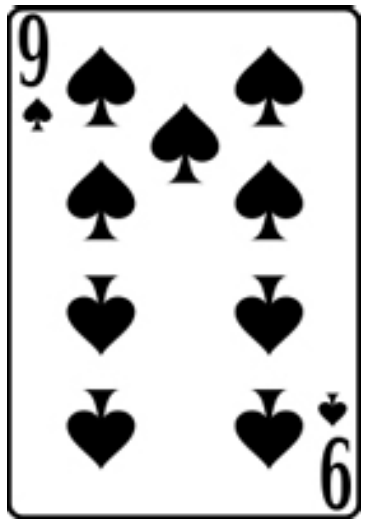
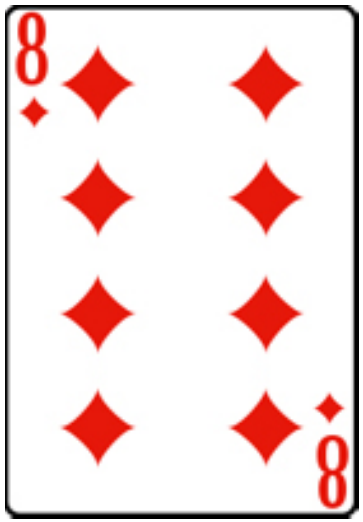
version où un joueur joue seul contre un croupier "automatique".

- chaque carte vaut des points :
  - le nombre de points d'une carte avec un numéro est le numéro
  - les figures (valet, dame, roi) valent 10 points
  - particularité : on peut choisir la valeur de l'as : soit 1 point, soit 11.
- on distribue deux cartes au croupier et deux cartes au joueur
- le joueur observe ses deux cartes, mais ne peut observer qu'une seule des cartes du croupier
- le joueur peut demander, une à une, une nouvelle carte, ou il peut annoncer qu'il ne veut pas de nouvelle carte.
- s'il s'arrête, le croupier peut faire de même : demander, une à une, une nouvelle carte, ou s'arrêter.
- si en recevant une nouvelle carte la somme des points dépasse 21 points, le joueur qui a reçu la carte perd.
- si les deux joueurs se sont arrêtés, le plus proche de 21 points gagne.
- le croupier joue avec une stratégie fixe : il demande une carte tant que la somme de ses cartes ne dépasse pas 17 points.

Blackjack

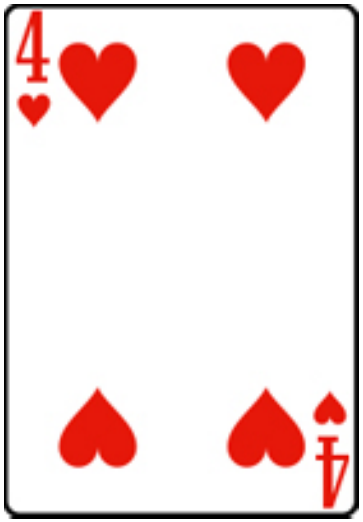
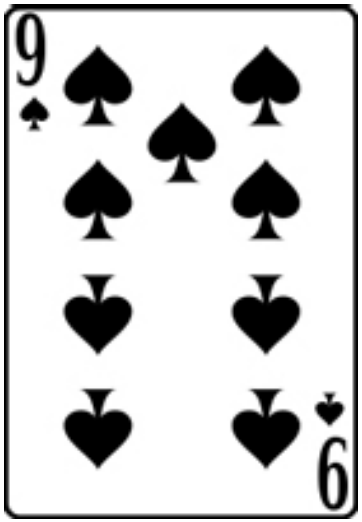
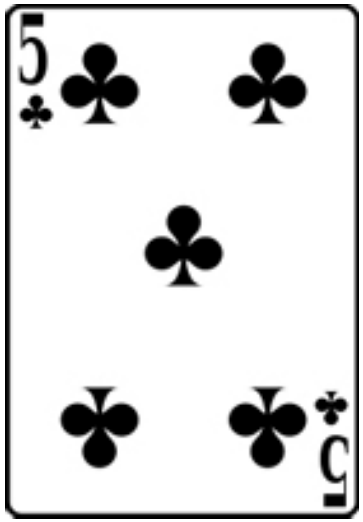
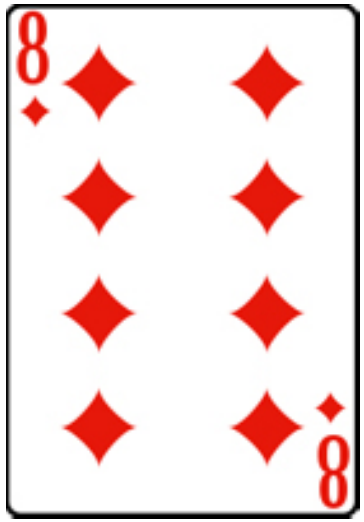
Que faites-vous ?

joueur	croupier

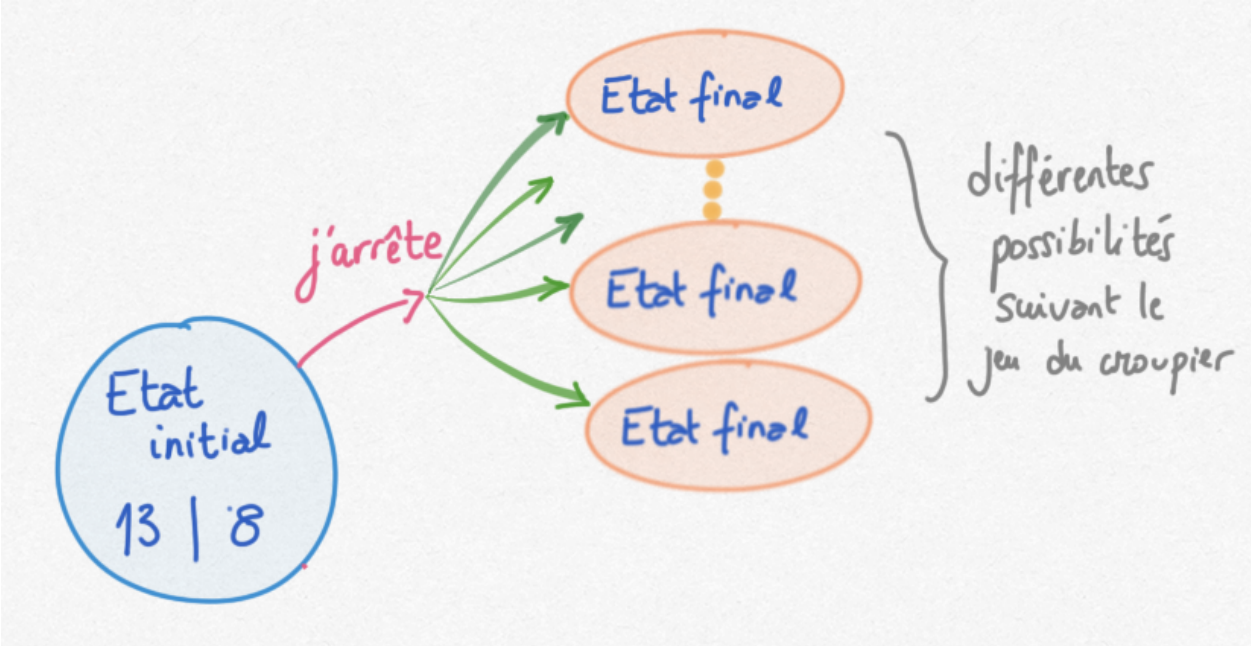
Blackjack

Supposons que vous avez demandé une nouvelle carte. Que faites-vous ?

joueur			croupier
			

Blackjack vs Bandits

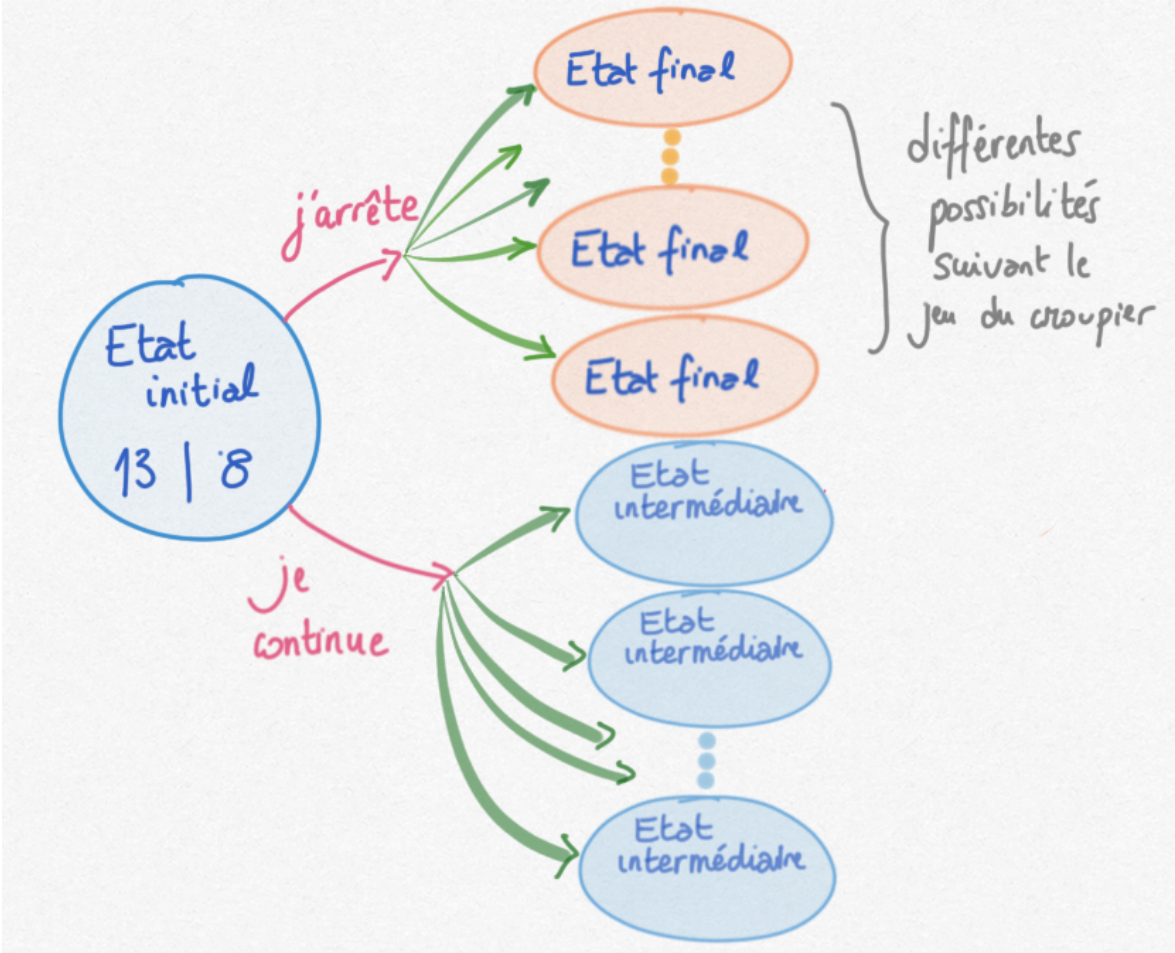
Avec cette version de blackjack, on doit maintenant raisonner sur la possibilité d'avoir de nouveau à prendre une décision dans un nouvel état.



action "j'arrête" : similaire aux bandits : ⇨ ensuite, selon le jeu du croupier et ses décisions, je gagne ou je perd la partie.

Blackjack vs Bandits

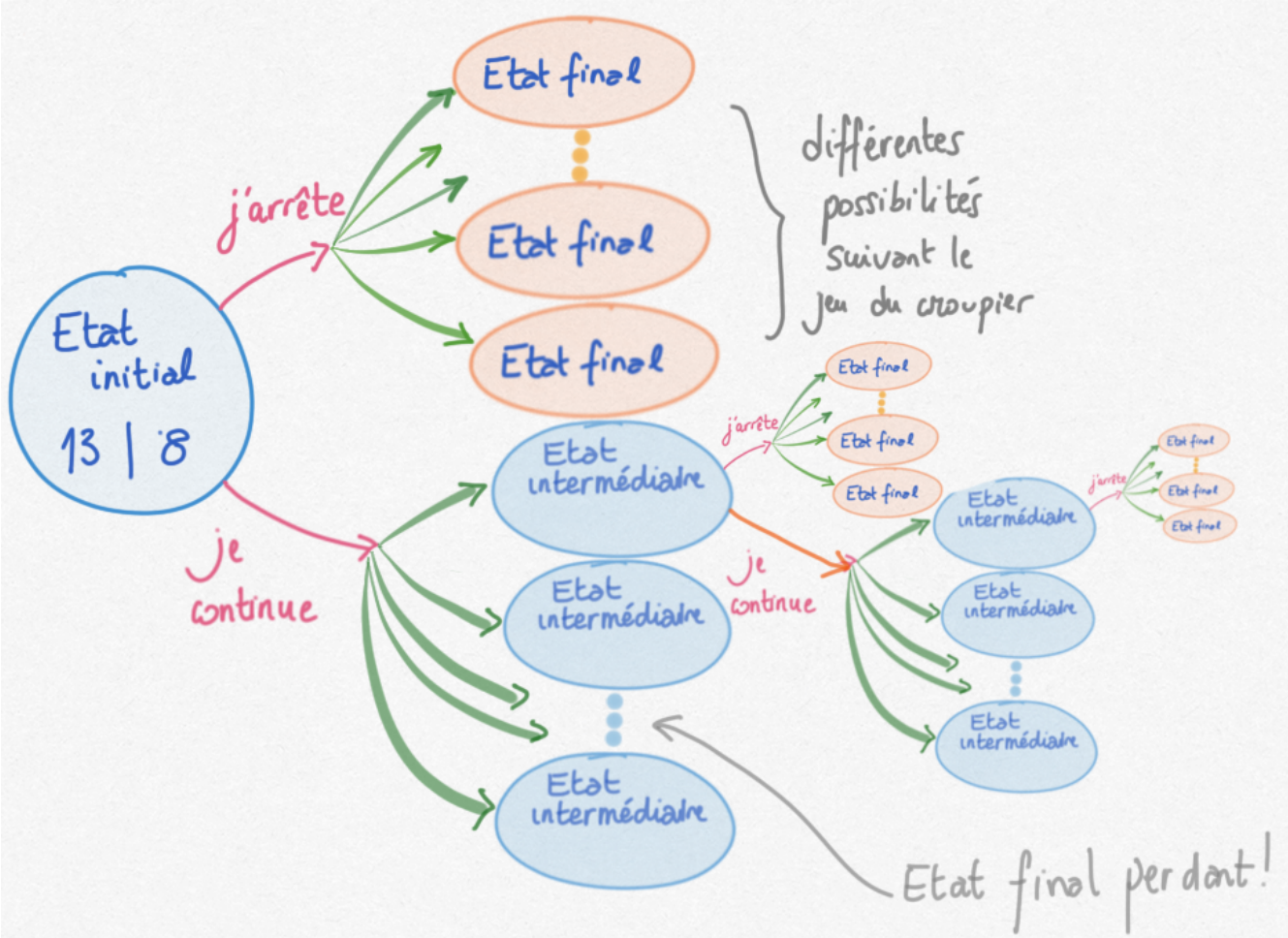
Avec cette version de blackjack, on doit maintenant raisonner sur la possibilité d'avoir de nouveau à prendre une décision dans un nouvel état.



action "je continue" : maintenant, je peux me retrouver dans plusieurs états suivants (dont des perdants !)

Blackjack vs Bandits

Avec cette version de blackjack, on doit maintenant raisonner sur la possibilité d'avoir de nouveau à prendre une décision dans un nouvel état.



et je peux avoir de nouvelles décisions à prendre...

Modèles

C'est avec ce type de modèle que l'on va travailler dans le cours :

- on se trouve dans un état courant (que l'on connait)
- on a plusieurs actions possibles (que l'on connait)
- lorsqu'on exécute une action, on peut aller dans un état suivant
  - on ne peut pas prédire *a priori* quel sera l'état suivant après avoir pris une action dans l'état courant
  - le résultat d'avoir exécuté une action dans un état n'est pas forcemment deterministe (l'état suivant peut être stochastique, ici dépend par exemple de la carte tirée)
- on reçoit une récompense immédiate après chaque exécution d'action.
  dans l'exemple du blackjack, on reçoit 0 pour les actions intermédiaire, et -1, 0 ou 1 lorsqu'on atteint un état final.

⇨ **but** choisir de manière optimale une action pour chaque état.

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

Notations

On note

- S l'ensemble des états.
- A l'ensemble des actions
- S_t l'état courant à l'instant t
- A_t l'action exécutée à l'instant t
- on passe de l'instant t à l'instant $t+1$ lorsqu'on exécute une action dans un état

⇨ système dynamique à temps discret.

Hypothèse de Markov

Si on exécute une action a dans un état s :

- On obtient une récompense r
- On arrive dans l'état suivant 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 !

Transitions

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

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

  - définit comment on passe d'un état à un autre en fonction de l'action exécutée
  - satisfait l'hypothèse de Markov
- **modèle de récompense**

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

Récompense "immédiate"

  - définit la valeur de la récompense en fonction de l'état courant et de l'action exécutée
  - satisfait l'hypothèse de Markov

Processus décisionnel de Markov

Definition (Processus décisionnel de Markov)

Un *Processus décisionnel de Markov* est un tuple $\langle S, A, T, R \rangle$ 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
  $R^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$

- dans certains cas, on peut aussi avoir un vecteur de probabilité qui représente la probabilité de se trouver dans un état initiallement.
- dans certains cas, on ajoutera à la définition un paramètre de la fonction objective que l'on cherchera à optimiser

Définitions : politique d'un agent

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

⇨ c'est ce qu'on cherche !

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 $\sum_{i \in N} p(i) = 1$

Quel est l'objectif ?

pour des tâches épisodiques (blackjack, exemple du robot)

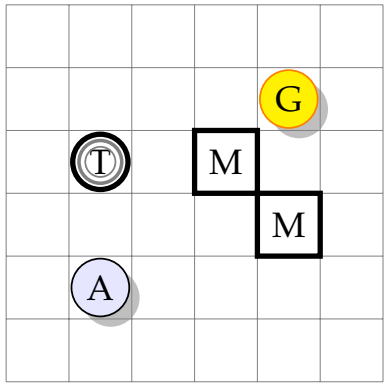
- il y a des états terminaux et initiaux
- on repart dans un état initial une fois qu'on atteint un état terminal

⇨ maximise le cumul des récompenses sur *un épisode*

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

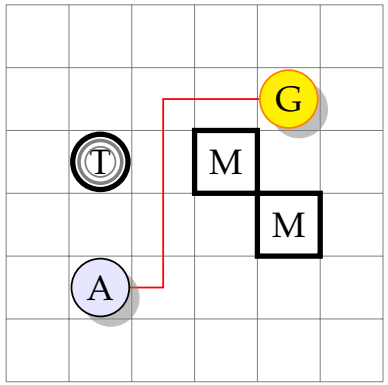
N.B. La longueur d'un épisode n'est pas nécessairement fixe
(ex pour la version de blackjack, la longueur dépend de combien de cartes on prend)

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)

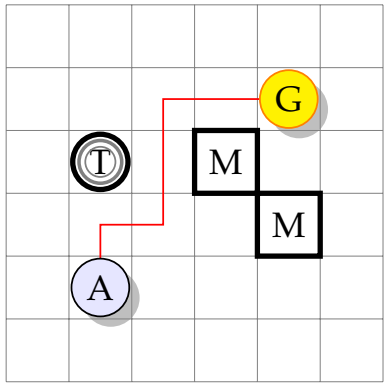
Exemple : contrôle optimal d'un robot



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)

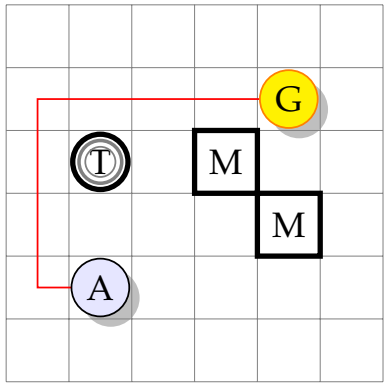
Exemple : contrôle optimal d'un robot



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)

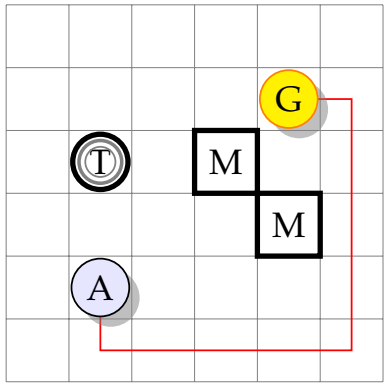
Exemple : contrôle optimal d'un robot



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)

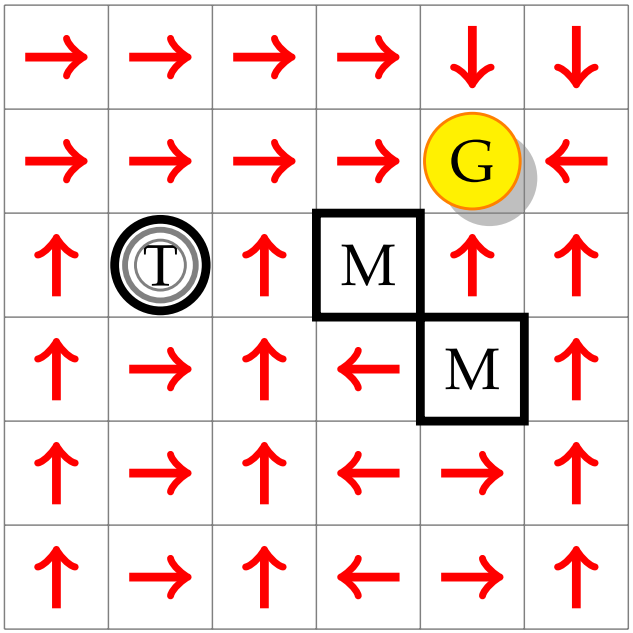
Exemple : contrôle optimal d'un robot



Gains : 91

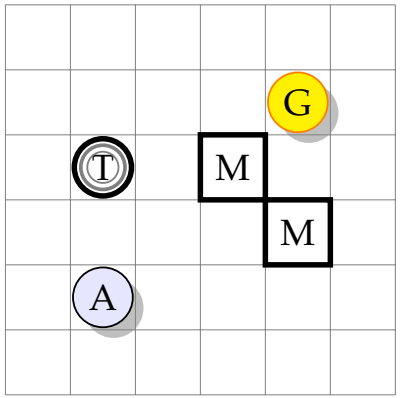
- ensemble des états : une grille $n \times n$
  - certains états sont des murs (M) : l'agent reste à sa place s'il essaye de traverser le mur !
  - certains états sont des trous (T) : si l'agent tombe dans un trou, l'épisode se termine
  - un état contient un pot d'or !
- les actions sont d'aller au 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 : une politique



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

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

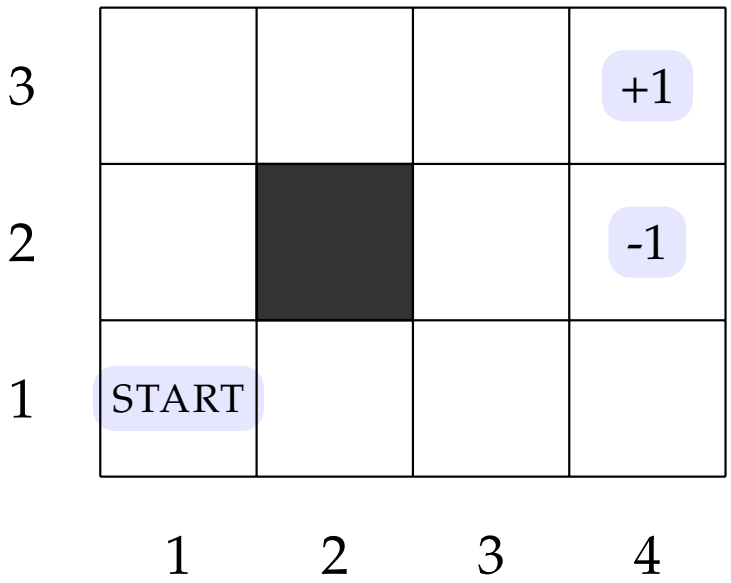
⇨ modèle de transition T connu ou non de l'agent.

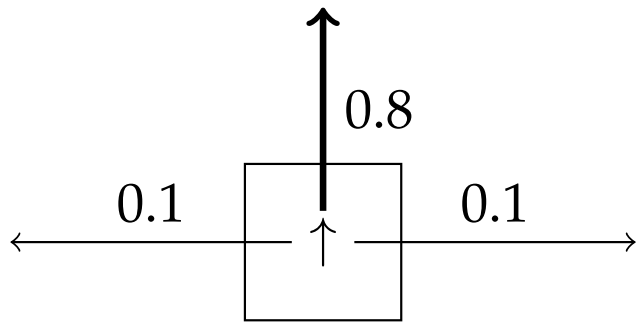
Comment calculer la fonction de valeur ou une politique optimale ?

Exemple : gridworld

- l'agent paie une pénalité pour chaque déplacement (il dépense de l'energie)
- certaines cases peuvent contenir une grosse récompense

cf Opportunity et Curiosity sur Mars...





exemple action vers le haut

Quel est l'objectif ? – tâches en continue

pour des tâches en continue (optimiser la marche d'un robot), il n'existe pas nécessairement d'état final.

- ⇨ maximise une récompense avec escompte $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

  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 G_t est bien définie.
    ⇨ l'agent cherche un équilibre entre la récompense immédiate et celle qu'il obtiendra dans le futur

- maximiser une récompense "en moyenne" $G_t = \lim_{T \to +\infty} \frac{1}{T} \cdot \sum_{k=0}^{T-1} r_{t+k+1}$

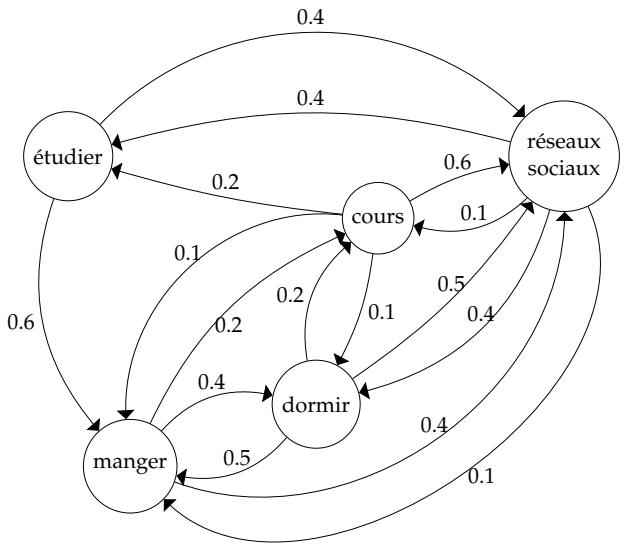
Focus pour le cours

- problèmes itératifs en continue.
- objectif : maximiser la somme "avec escompte" $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$
  - 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, … 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
- $R(s) = \mathbb{E}[r_{t+1}|S_t = s]$ 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 $G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

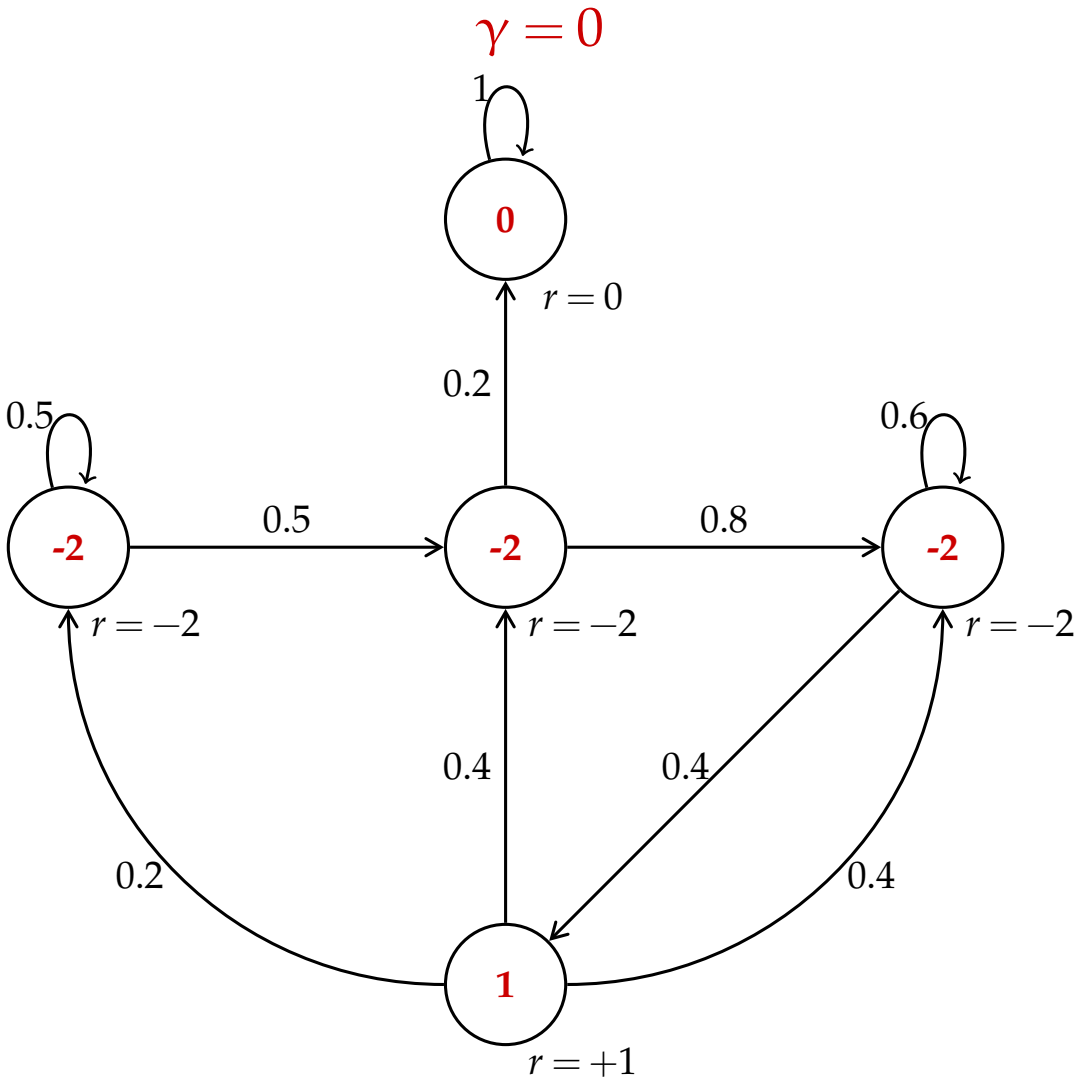
La fonction de valeur donne la valeur à long terme de l'état s.

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

a. on peut aussi avoir une version deterministe

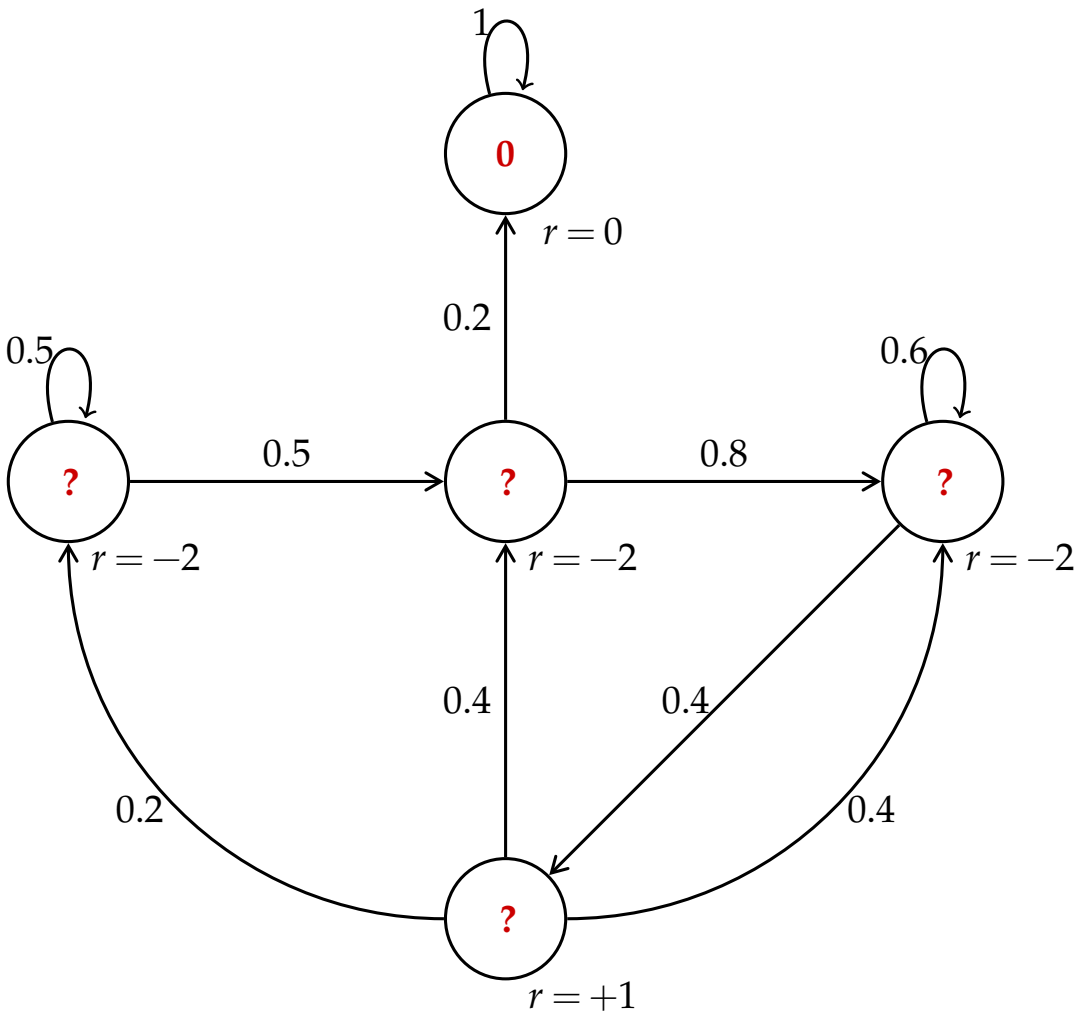
Un pas en arrière : chaînes de Markov avec récompenses

Ici, la notation $r = k$ représente la valeur quand on quitte l'état.



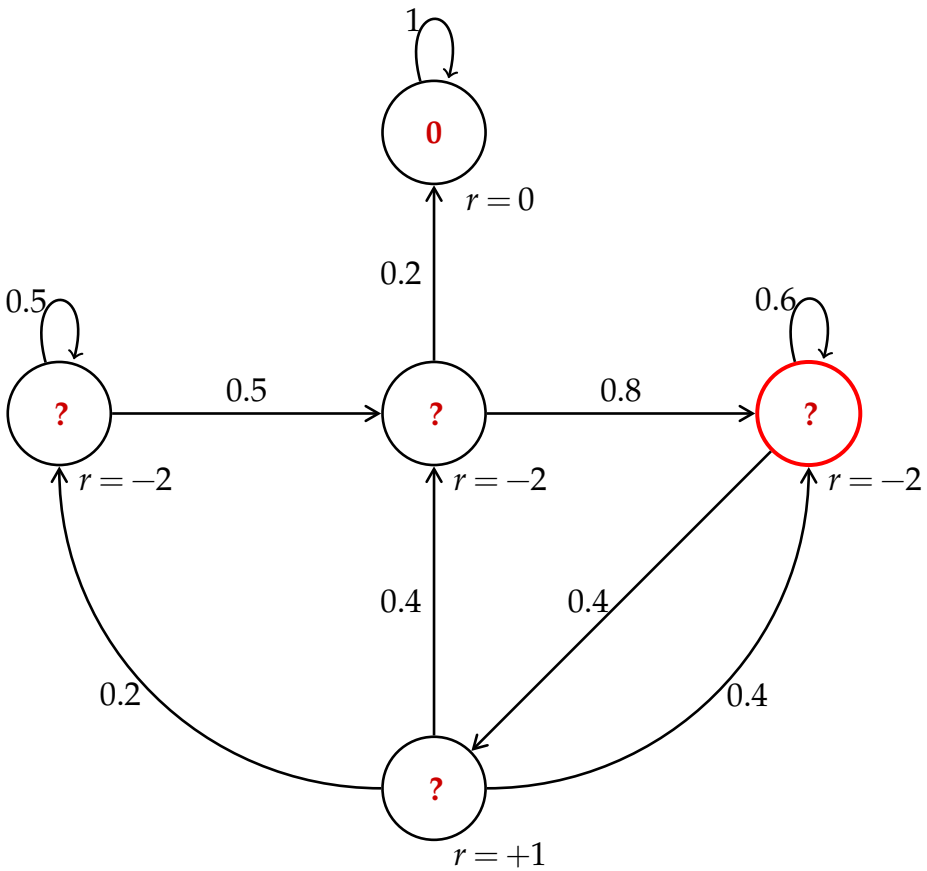
Un pas en arrière : chaînes de Markov avec récompenses

$\gamma = 0.9$??



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 valeur

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

$$
\begin{aligned}
v(s) &= \mathbb{E}\left[G_t \mid S_t = s\right] \\
&= \mathbb{E}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \mid S_t = s\right] \\
&= \mathbb{E}\left[r_{t+1} + \gamma\left(r_{t+2} + \gamma r_{t+3} + \ldots\right) \mid S_t = s\right] \\
&= \mathbb{E}\left[r_{t+1} + \gamma G_{t+1} \mid S_t = s\right] \\
&= \mathbb{E}\left[r_{t+1} \mid S_t = s\right] + \gamma \mathbb{E}\left[G_{t+1} \mid S_t = s\right] \\
&= R_s + \gamma \sum_{s' \in S} T_{ss'} \mathbb{E}\left[G_{t+1} \mid S_{t+1} = s', S_t = s\right] \\
&= R_s + \gamma \sum_{s' \in S} T_{ss'} \mathbb{E}\left[G_{t+1} \mid S_{t+1} = s'\right] \text{ Hypothèse de Markov} \\
&= R_s + \gamma \sum_{s' \in S} T_{ss'} v(s')
\end{aligned}
$$

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 valeur

On obtient donc l'équation

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

Dans notre exemple, cela donne

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

Equation de Bellman : résolution

On doit résoudre un système linéaire !

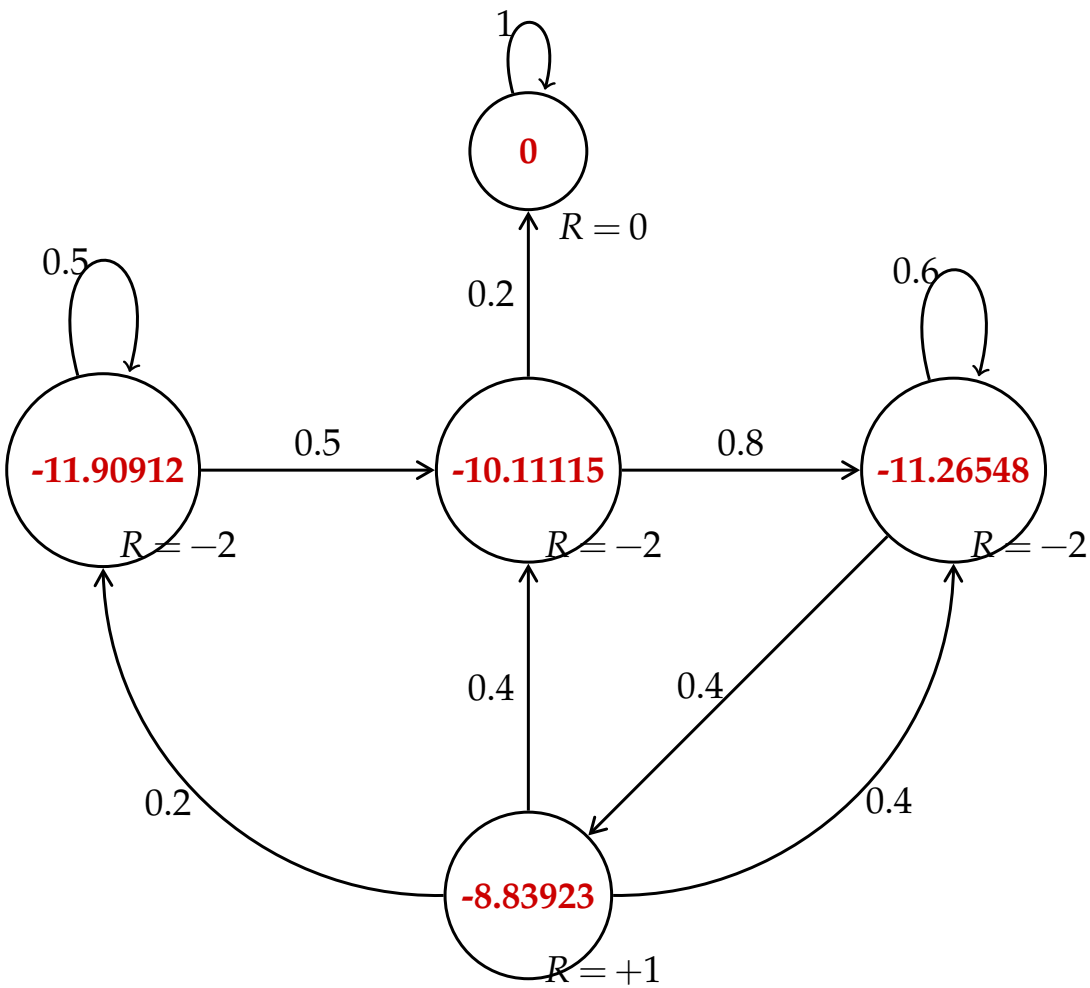
$$\begin{aligned} v &= R + \gamma \cdot Tv \\ (I - \gamma T)v &= R \\ v &= (I - \gamma T)^{-1} R \end{aligned}$$

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

- 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

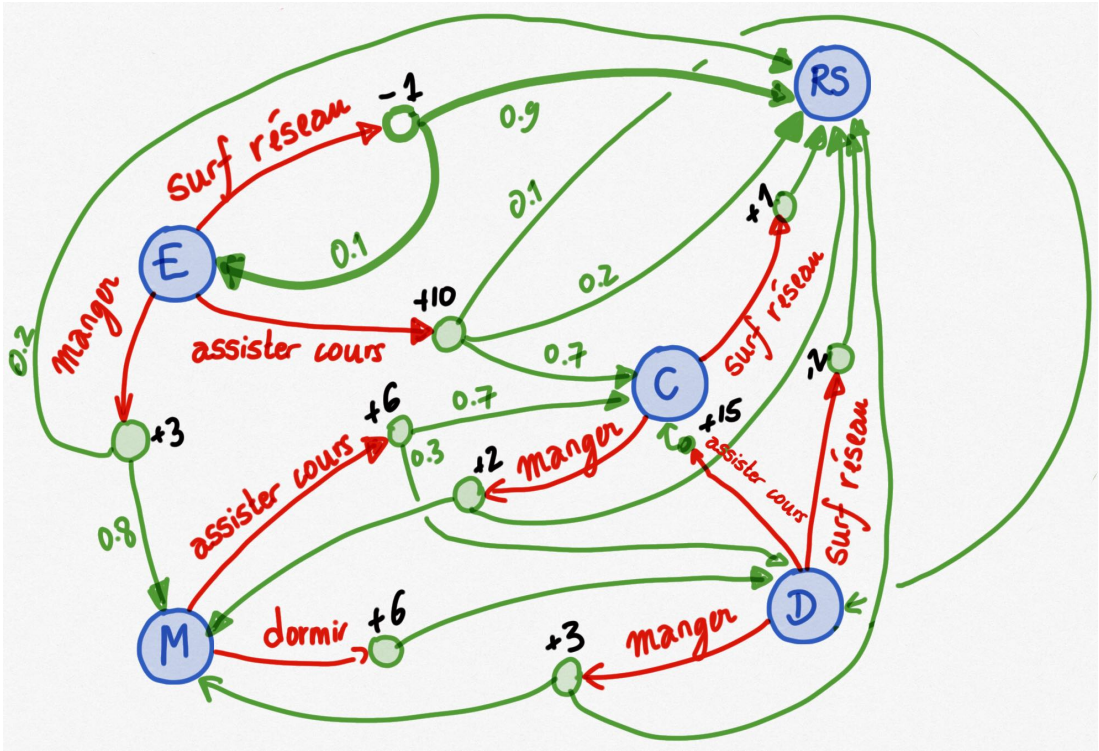
On ajoute des actions au modèle précédent.

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 (parfois vous voulez aller étudier, mais vous finissez sur facebook !)
  - actions sont les flèches rouges
  - nœuds "chance" en vert et plusieurs possibilités, la valeur des probabilités est notée en vert
- la valeur des noeuds est ici la récompense (immédiate) notée en noir (valeur moyenne obtenue en exécutant l'action)

Rappel : politique

La politique d'un agent définit son comportement

Definition (Politique)

Une *politique* π est une distribition 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 $\langle S, T^\pi, R^\pi, \gamma \rangle$

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

Fonctions de valeurs

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 dévaluée 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

1. on tire notre action avec la politique $\pi(s)$
2. pour chaque action, on choisit l'action a avec la probabilité $\pi(a|s)$,
3. 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 :

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

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

Equation de Bellman

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

On peut donc écrire l'expression vectorielle

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

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 valeur 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 $\forall s \in S\ v_\pi(s) \geqslant v_{\pi'}(s)$

Théorème

- 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)

Programmation dynamique – Itération sur les valeurs

On peut montrer (cf notes) que la solution de l'équation de Bellman optimale est l'unique point fixe de l'opérateur de Bellman $\mathcal{T}$,
i.e. suite $v_{k+1} = \mathcal{T} v_k$ converge vers $v^\star$

⇨ pour trouver la solution, on peut donc appliquer itérativement l'opérateur pour converger vers l'optimal !

```
1  for each s ∈ S and k ∈ ℕ
2     V_k(s) ← 0
3
4  repeat for  k = 0 to ...
5
6     for each s ∈ S
7
8        V_{k+1}(s) ← max_{a∈A} [ R_s^a + γ Σ_{s'∈S} T_{ss'}^a V_k(s') ]   /* mise à jour */
9
10 until convergence
```

Value Iteration – plus efficace pour la mémoire

```
1  for each s ∈ S
2     V(s) ← 0
3
4  repeat
5
6     for each s ∈ S
7
8        V(s) ← max_{a∈A} [ R^a_s + γ Σ_{s'∈S} T^a_{ss'} V(s') ]      /* mise à jour */
9
10 until convergence
```

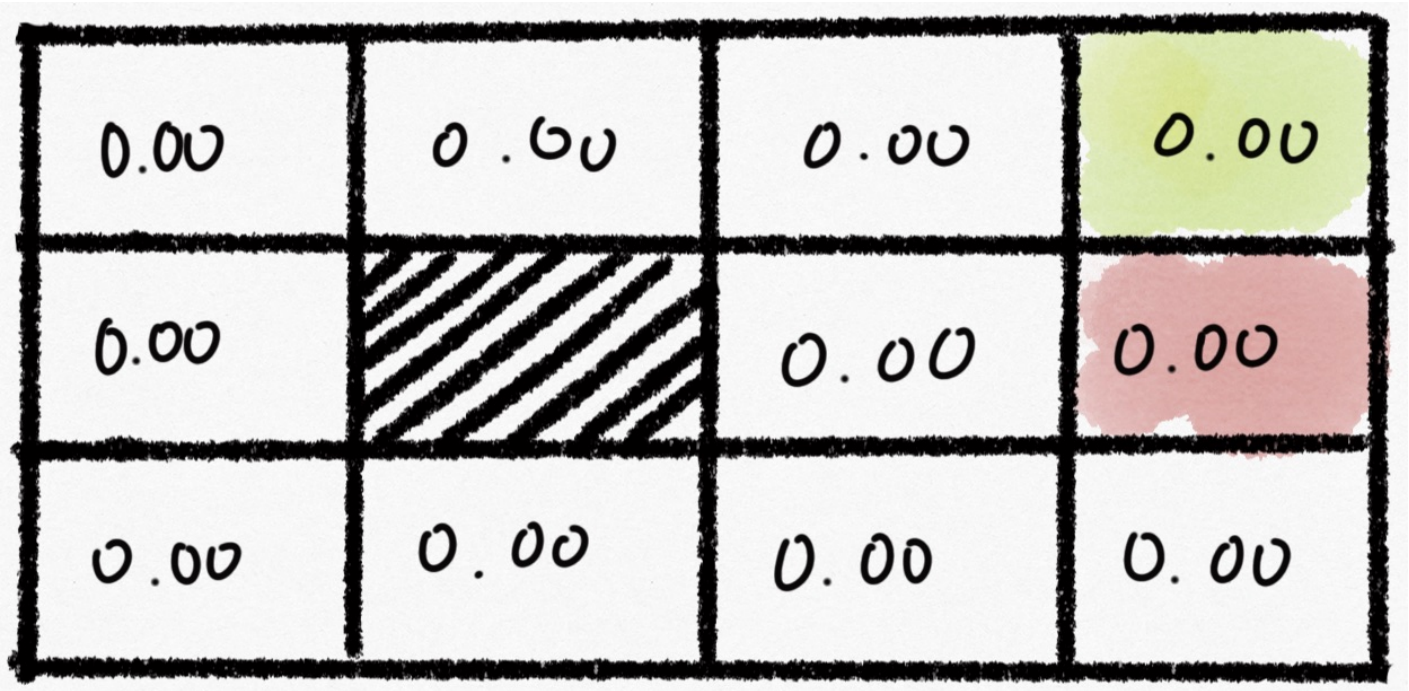
- On peut utiliser seulement un seul vecteur V.
- ⇨ Quand on fera la mise en jour, certaines entrées de V seront plus récentes que d'autres.
- cela ne perturbe pas la convergence.

Value Iteration – avec test de convergence

```
1  for each s ∈ S
2     V(s) ← 0
3
4  repeat
5     Δ ← 0                                   /* mesure le plus grand changement */
6     for each s ∈ S
7        v ← V(s)          /* sauvegarde l'ancienne valeur pour mesurer le changement */
8        V(s) ← max_{a∈A} [ R^a_s + γ Σ_{s'∈S} T^a_{ss'} V(s') ]        /* mise à jour */
9        Δ ← max(Δ, |v − V(s)|)          /* mise à jour du plus grand changement*/
10 until Δ < ε                                 /* test convergence */
```

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

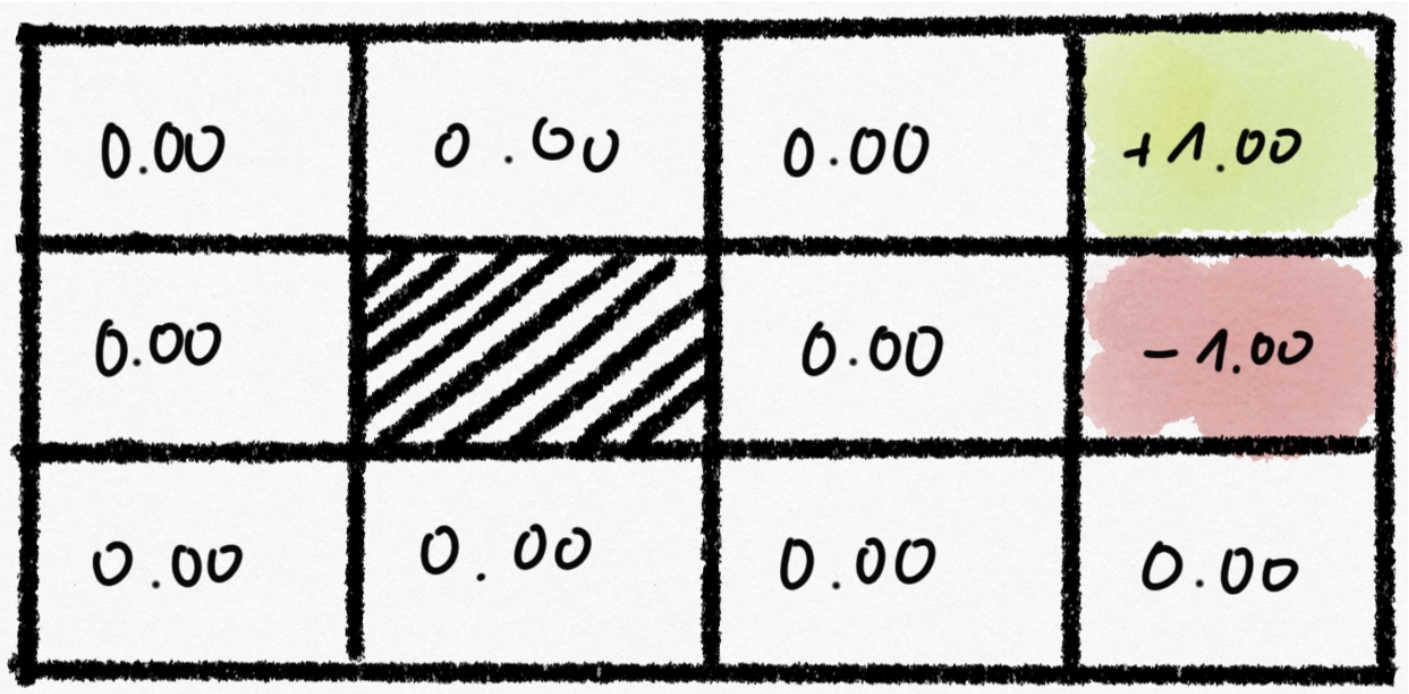
Itération sur les valeurs : $k = 0$



$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

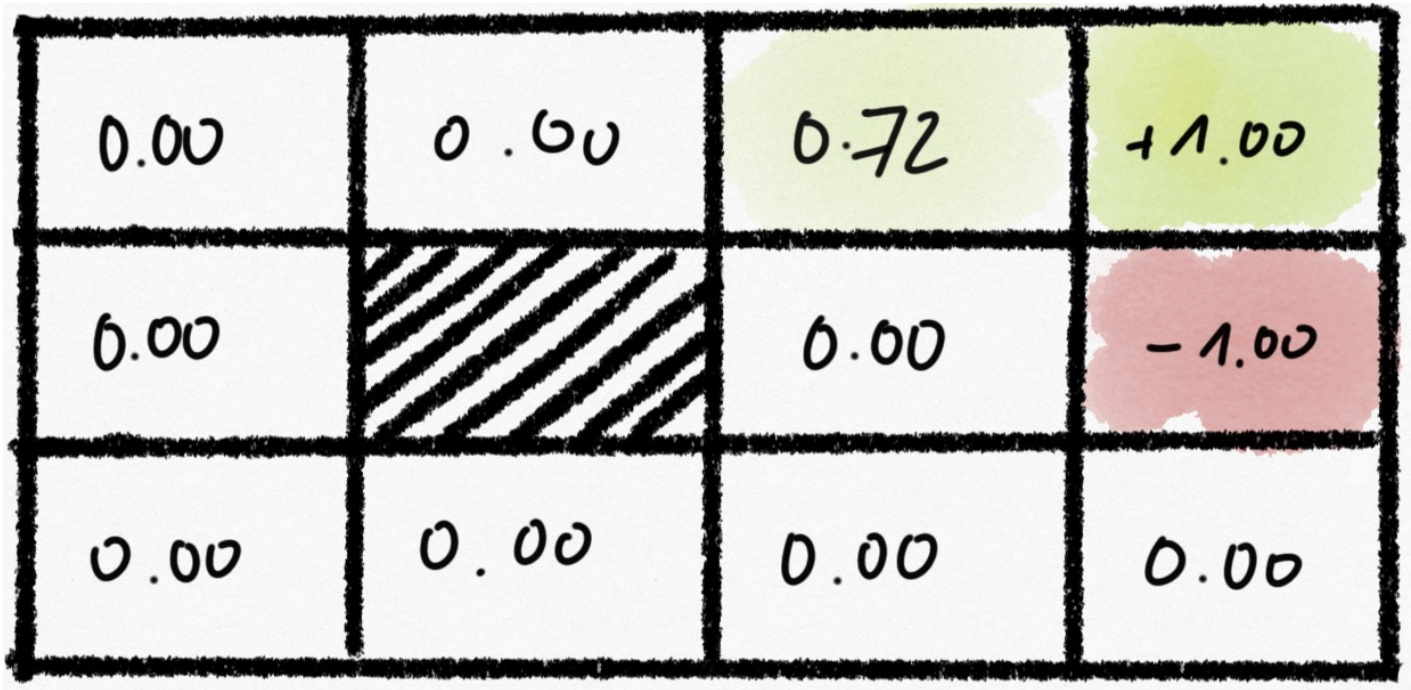
bruit veut dire $1 - bruit$ probabilité qu'une action réussisse, $\frac{1}{2} bruit$ de déraper vers la gauche, $\frac{1}{2} bruit$ de déraper vers la droite ; r est la pénalité pour chaque déplacement sauf pour un déplacement dans un état final.

Itération sur les valeurs : $k = 1$



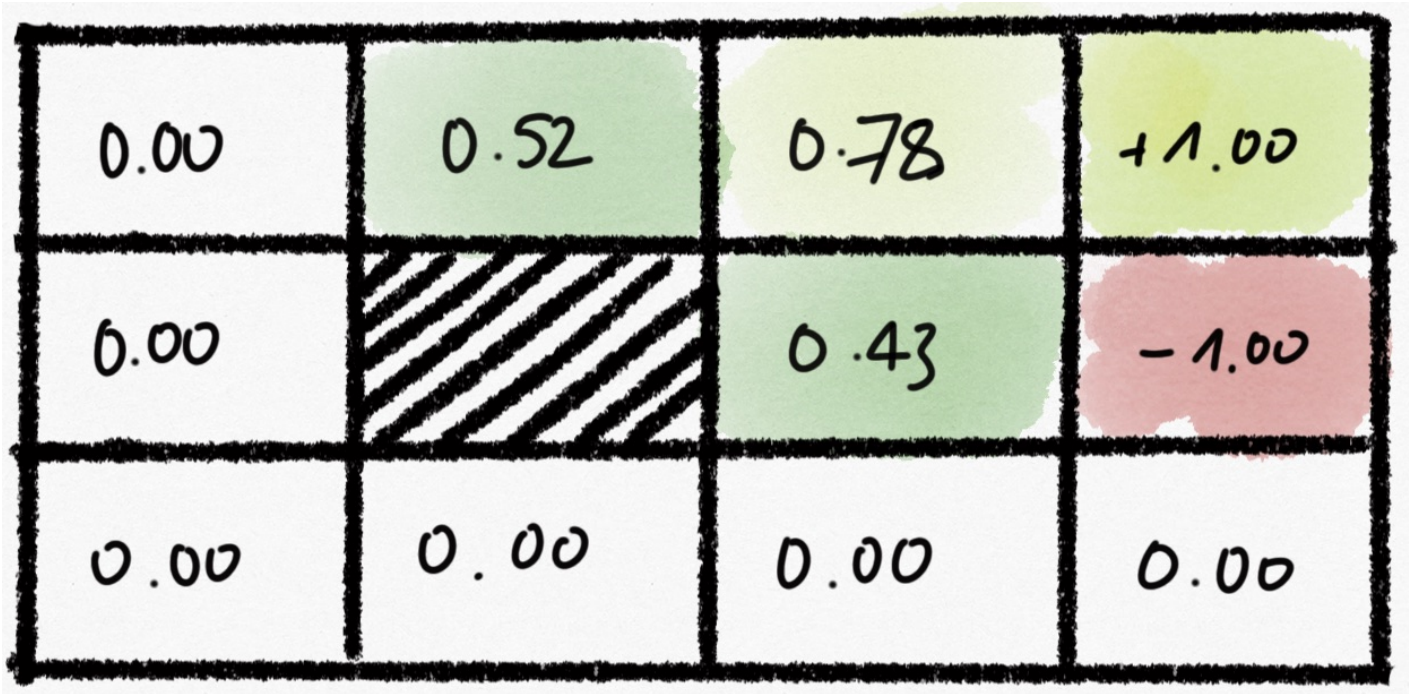
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 2$



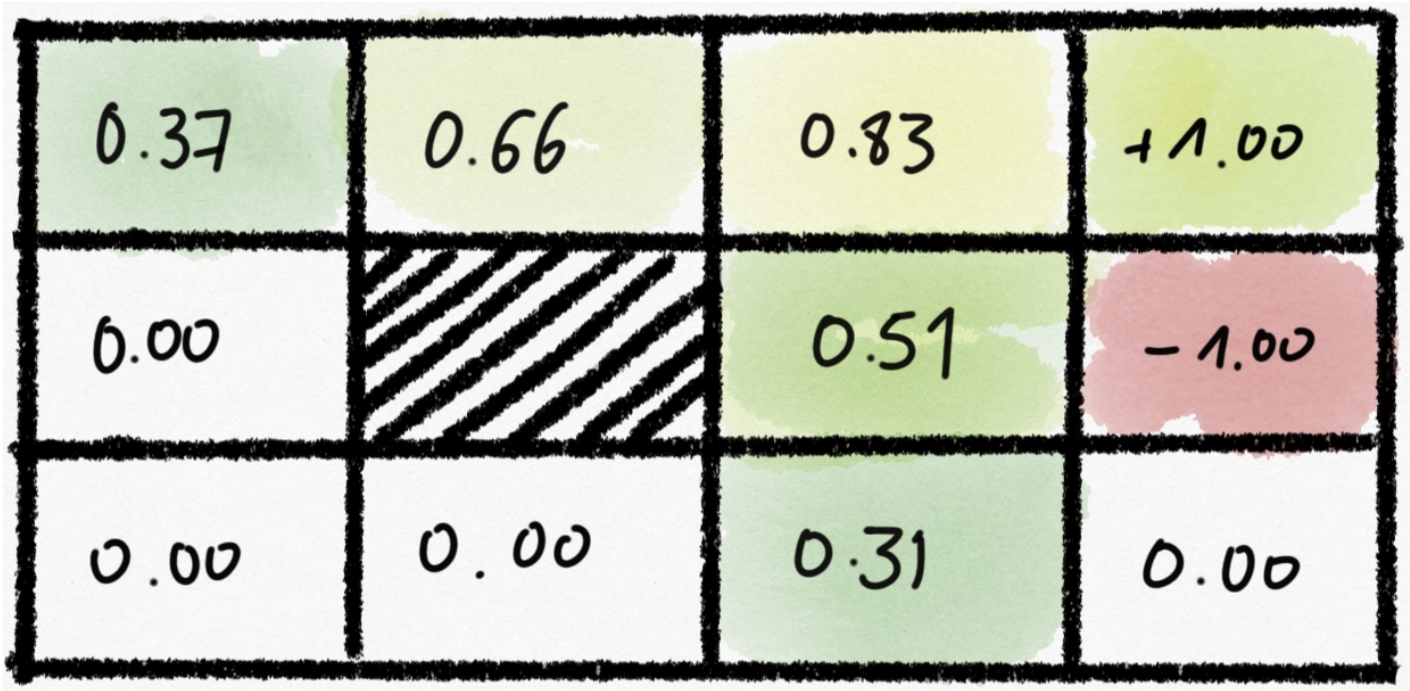
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 3$



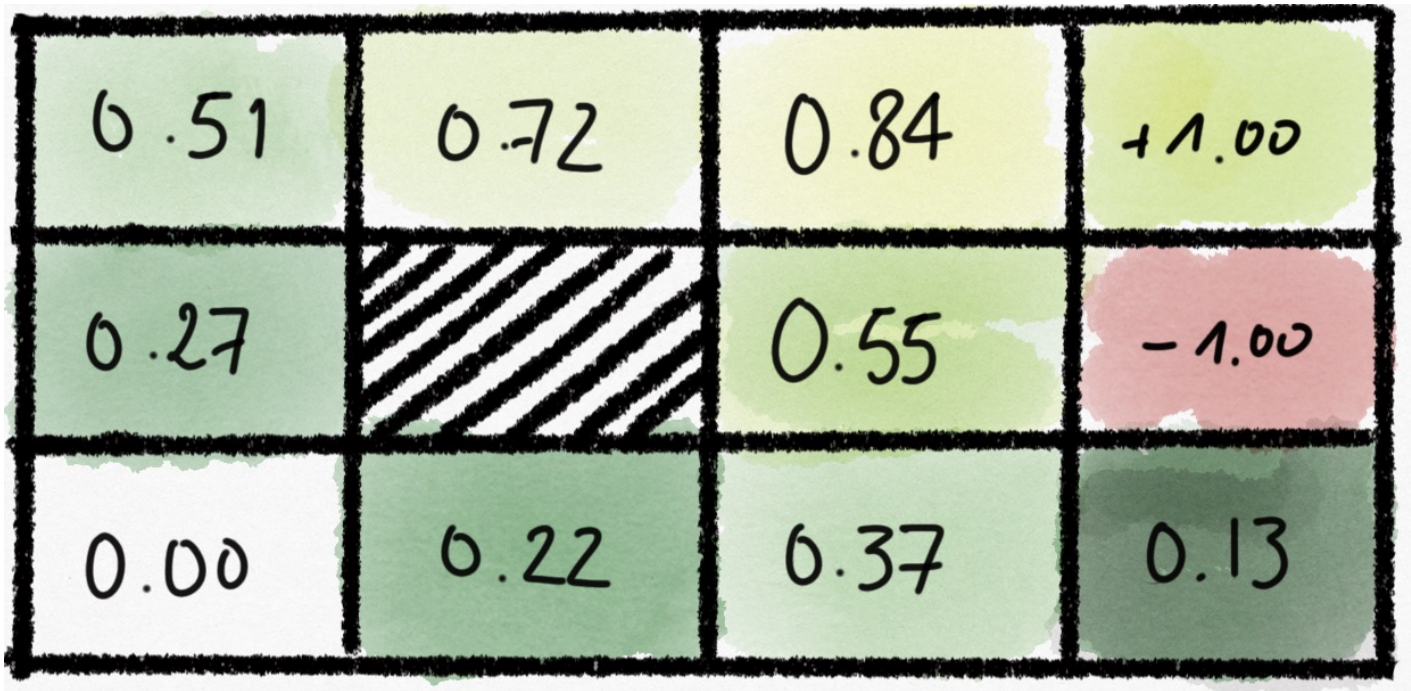
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 4$



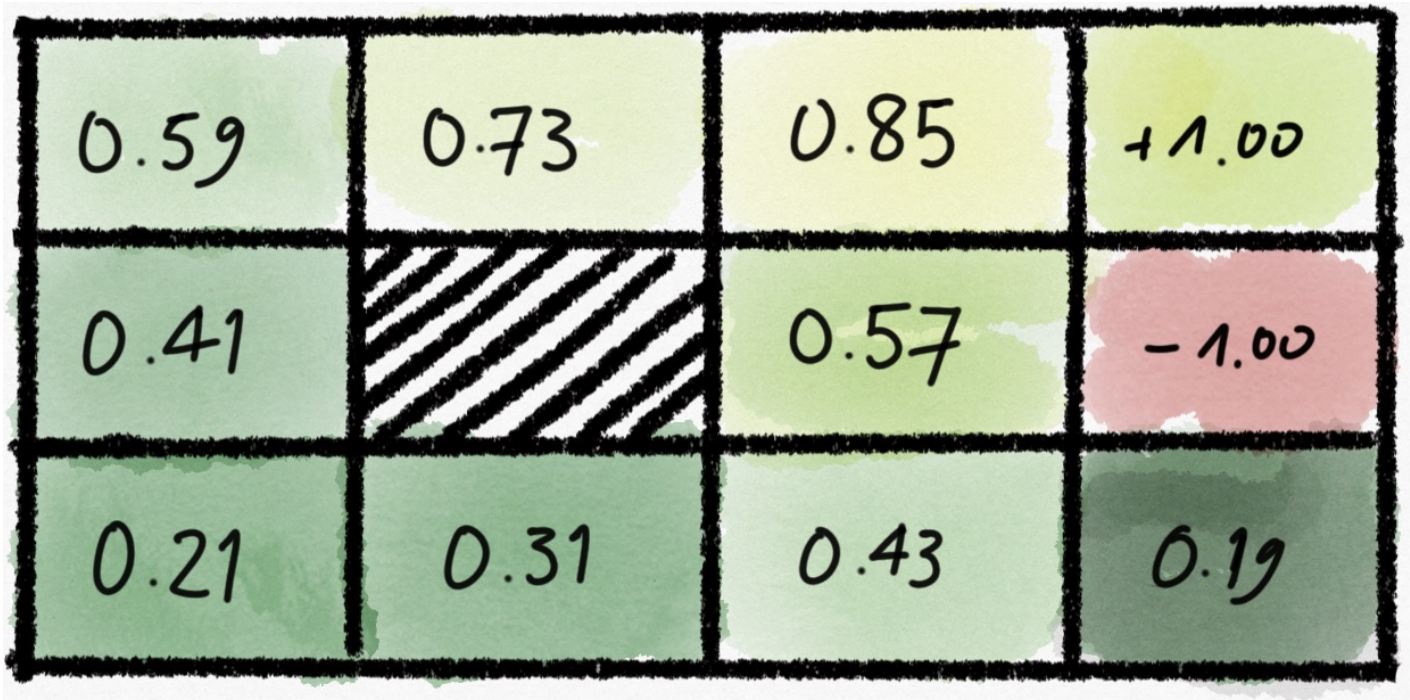
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 5$



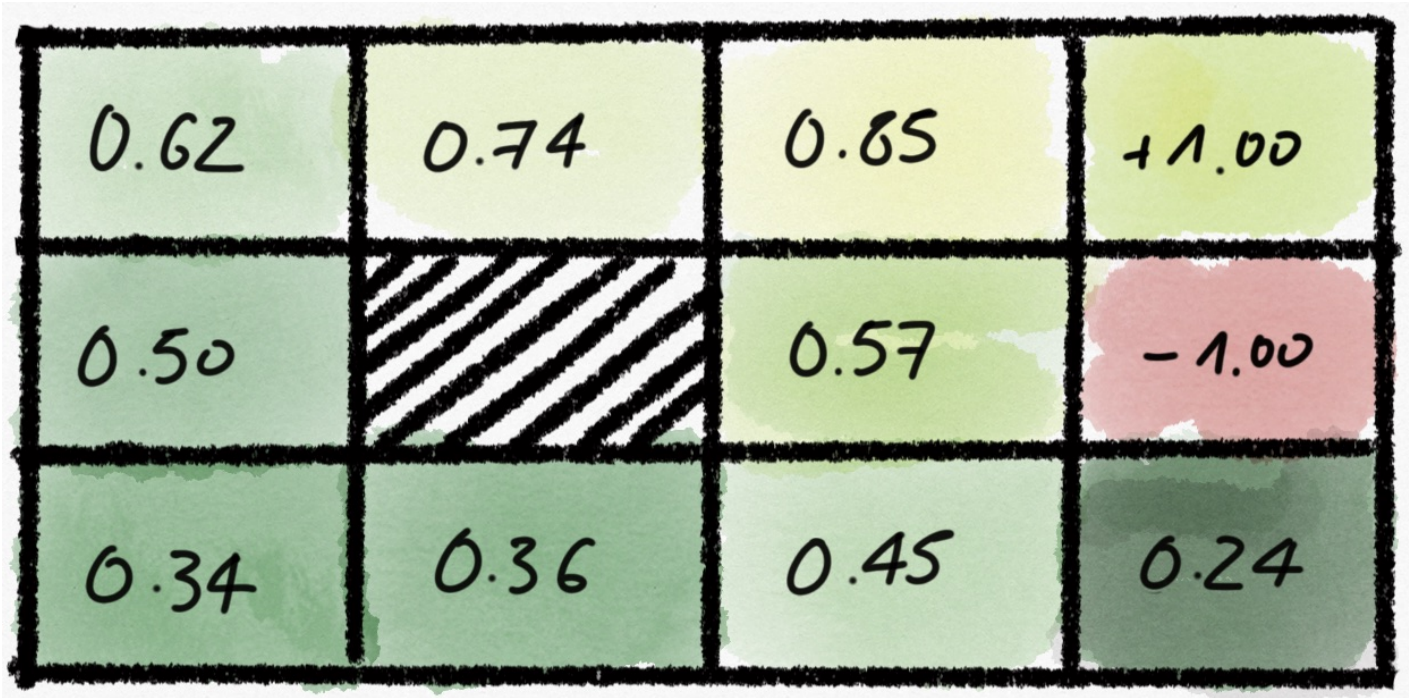
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 6$



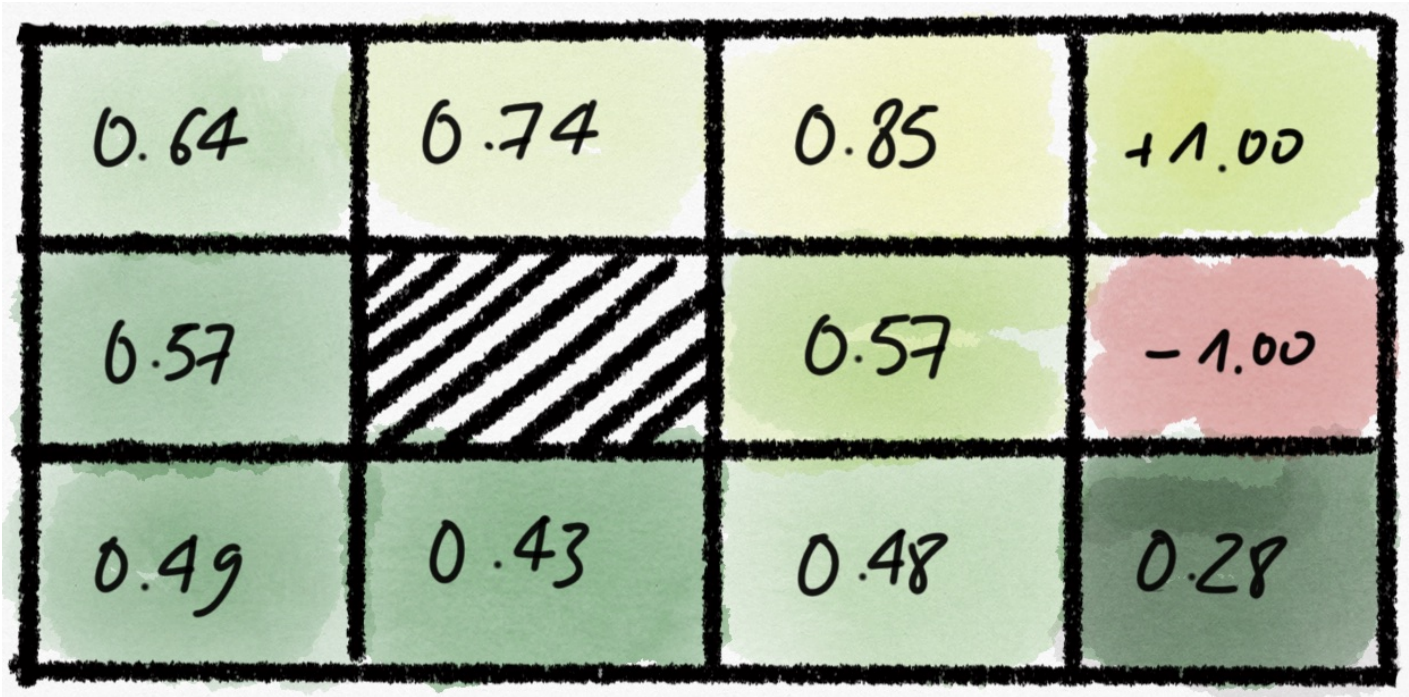
$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 7$



$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Itération sur les valeurs : $k = 100$



$$r = 0$$
$$\gamma = 0.9$$
$$bruit = 0.2$$

Limitations de Value Iteration

- la méthode est lente
- la valeur du max change rarement

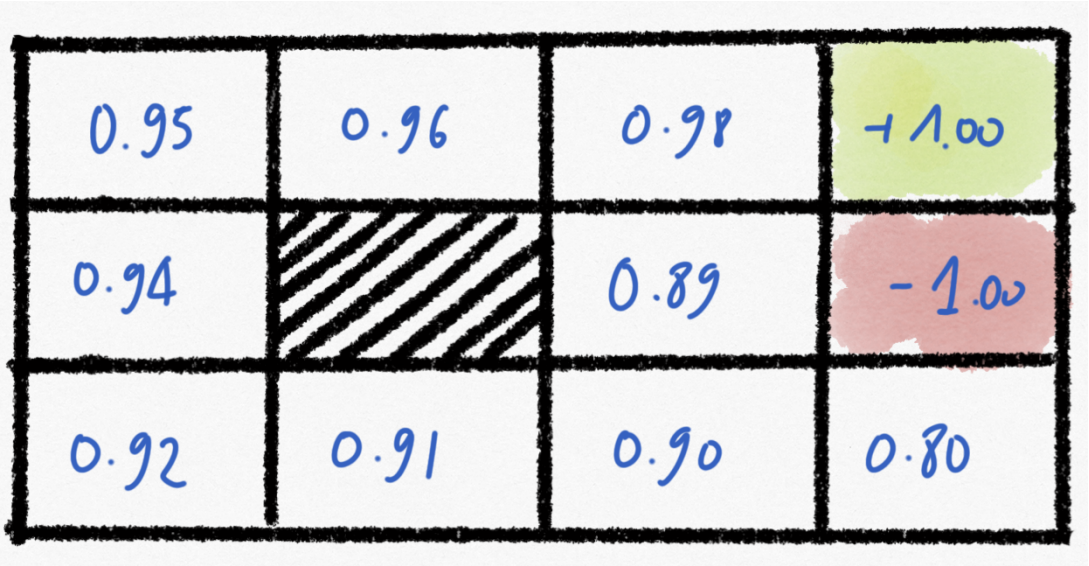
⇨ pourtant c'est cela qui va aider à changer toutes les valeurs !

- les valeurs peuvent mettre longtems à converger exactement alors que la politique, elle, est déjà optimale depuis longtemps

⇨ on va essayer de travailler sur la politique.

Comment déterminer une bonne action à partir de v ?

Supposons qu'on connaisse les valeurs optimales $v^\star(s)$



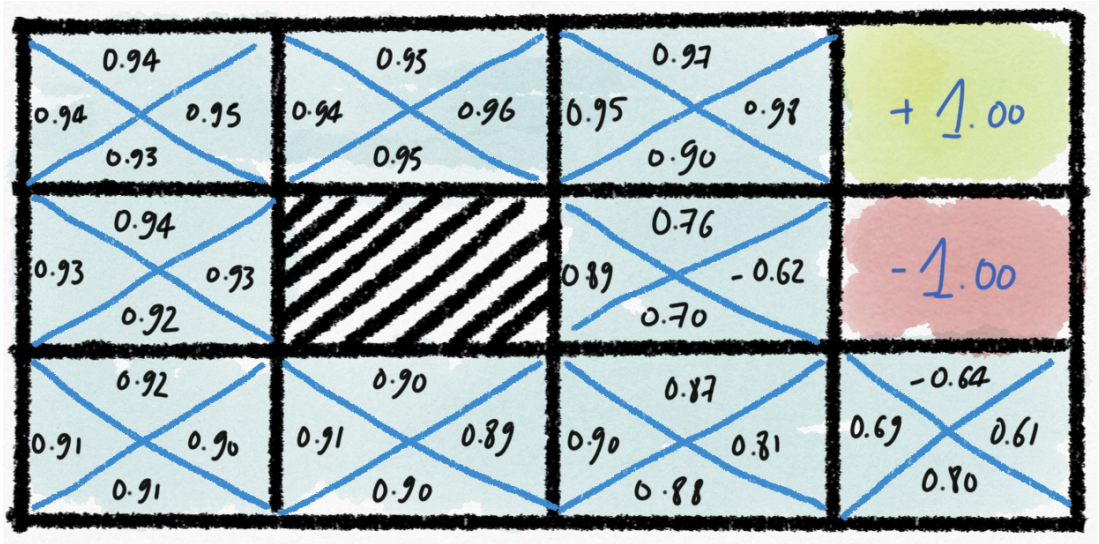
Comment devons-nous agir ?

$$\pi^\star(s) = arg\max_a \left[R_s^a + \gamma \sum_{s'} T_{ss'}^a v^\star(s') \right]$$

⇨ on peut appeler cette étape faire une extraction de politique.

Comment déterminer une bonne action à partir de q ?

Supposons qu'on connaise les valeurs optimales $q^\star(s,a)$



Comment devons-nous agir ?

Trivial, on choisit la meilleure action ! (ou une des meilleurs actions).

$$\pi^\star(s) = arg\max_a q*(s,a)$$

⇨ morale de l'histoire : les actions sont plus faciles à obtenir à partir de q qu'à partir de v ! !

Autre stratégie de résolutions : améliorer une politique petit à petit

On peut partir d'une politique arbitraire π et essayer de la modifier pour améliorer les performances.

On va 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

1- l'évaluation d'une politique

2- l'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 ?

Comparaison

- Les deux méthodes calcule le même résultat : à la fin, on a les mêmes valeurs optimales ($v^\star$ et $q^\star$)
- Itération sur les valeurs
  - la politique est implicite. A chaque itération, on met à jour les valeurs
  - si on travaille avec v, on doit utiliser l'extraction d'une politique pour obtenir la politique optimale.
- Iteration sur les politiques
  - plusieurs itérations pour mettre à jour les valeurs d'une politique fixe (mais pour chaque itération, on ne considère qu'une seule action, ce qui devrait être rapide).
  - on met à jour la politique, on doit comparer toutes les actions (peut être lent)
  - soit on améliore la politique, soit on a terminé !

Conclusions

Un PDM est un tuple $\langle S, A, T, R, \gamma \rangle$

- Le modèle des PDMs permet de représenter des problèmes de décisions séquentielles dans l'incertain
- A chaque instant t, on cherche à optimiser l'objectif *à long terme*

  $$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$
- Deux algorithmes permettent de calculer une politique **optimale**
  - iteration sur les valeurs : travaille uniquement sur la fonction de valeurs.
  - itération sur les politiques : manipule explicitement une politique
- on a des garanties de convergence.
- Nécessite la connaissance des modèles T et R