

Apprentissage par renforcement :

Introduction

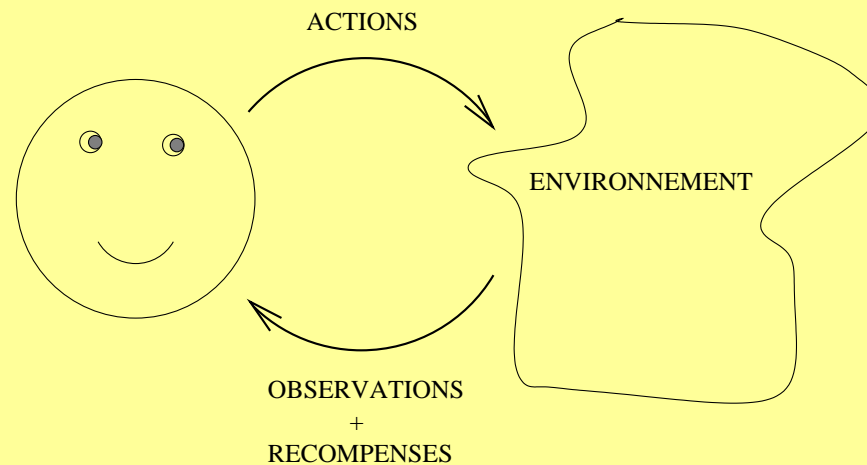
Frédéric Garcia



*Unité de Biométrie et Intelligence Artificielle
BP 27, 31326 Castanet-Tolosan*

Le paradigme de l'apprentissage par renforcement

Un agent autonome agissant au sein d'un environnement, qui recherche au travers d'expériences itérées un comportement décisionnel optimal.



Le double objectif de l'apprentissage par renforcement

Conduire de manière optimale un système au cours du temps

Apprendre cette conduite optimale à travers des expériences

Le premier point a été très bien étudié ces 50 dernières années en automatique, sous le nom de contrôle optimal.

L'apprentissage par renforcement étend la portée de ces travaux à des problèmes auparavant non traitables.

Conduite optimale d'un système et AR

Quelques notions clés

- l'agent et l'environnement
- le temps
- les états
- les actions
- la dynamique de l'état
- les revenus instantanés
- les politiques
- le critère d'optimalité

l'agent et l'environnement

Un agent, au cours du temps,

- observe son environnement,
- prend des décisions qui influencent l'évolution de l'environnement,
- modifie son comportement

Le temps

Les interactions agent / environnement sont instantanées.

$T = \{t_1, t_2, \dots, t_n, \dots\}$ est l'horizon temporel

T peut être fini ou infini.

Espace d'états

- L'état résume la situation de l'agent et de l'environnement à chaque instant. Il peut décrire :
 - la situation de l'agent vis à vis de l'environnement (position, etc.)
 - l'état propre à l'environnement
 - l'état interne de l'agent (ses ressources, sa mémoire, ses plans, etc.)

Espaces d'actions

À chaque instant l'agent applique une action (décision, action physique, etc.)

À chaque état est associé un ensemble d'actions possibles.

La dynamique de l'état

- La dynamique de l'état résulte
 - des actions de l'agent sur l'environnement
 - de la dynamique propre de l'environnement
 - de la dynamique interne à l'agent

Cette dynamique est non-déterministe

$$s_t, a_t \rightsquigarrow s_{t+1}$$

Les revenus instantanés

Après chaque interaction avec l'environnement, l'agent reçoit un revenu instantané r_t

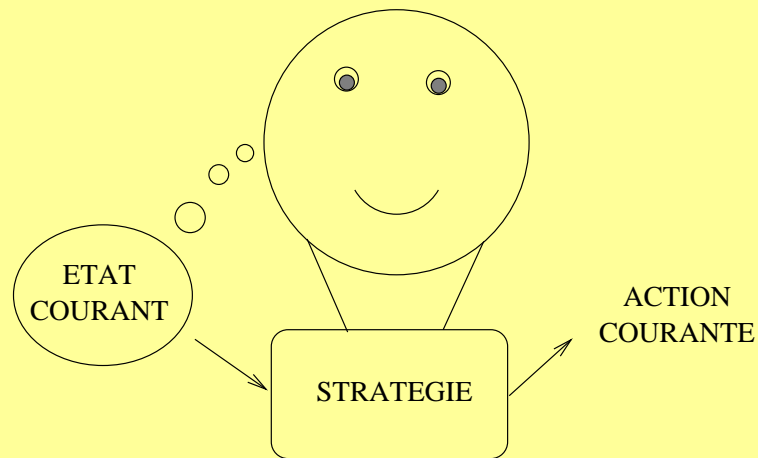
$$\begin{array}{ccc} s_t, a_t & \rightsquigarrow & s_{t+1} \\ & \downarrow & \\ & r_t & \end{array}$$

Les revenus sont réels ≥ 0 ou ≤ 0 .

Politiques d'actions

La politique, ou stratégie, modélise le comportement décisionnel de l'agent.

Les politiques markoviennes π associent à tout état l'action à exécuter en cet état : $s_t \mapsto a_t$



Différence entre politiques et plans

Quand la dynamique est déterministe, et si l'état initial est connu, alors l'exécution d'une politique conduit toujours à la même séquence d'actions :

$$(\pi(s_0), \pi(s_1), \dots, \pi(s_n), \dots)$$

Mais si l'état initial n'est pas fixé, ou si la dynamique est aléatoire, alors la même politique conduit à des séquences différentes.

Une politique permet de couvrir toutes les situations possibles : c'est un **plan universel**.

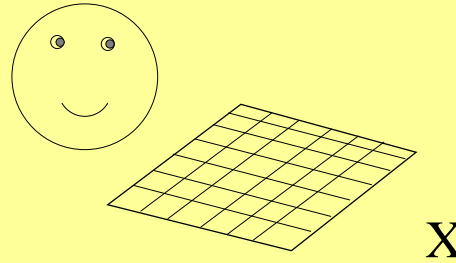
Critères d'optimalité

On cherche une politique qui maximise le cumul des revenus au cours du temps :

- $r_0 + r_1 + \dots + r_N$
- $r_0 + \gamma r_1 + \gamma^2 r_2 + \dots + \gamma^n r_n + \dots$ ($\gamma \in [0, 1]$)
- $\frac{r_0 + r_1 + \dots + r_{N-1}}{N}$ avec $N \rightarrow \infty$

en moyenne (car la dynamique est non-déterministe)

Applications : les jeux



- les dames américaines
- Backgammon
- Tétris
- Go, échecs

Tétris

- des pièces de forme différentes tombent du haut de l'écran
- on peut les bouger de gauche à droite et les faire tourner
- l'objectif est d'éliminer des lignes en les complétant
- le jeu s'arrête lorsque l'écran est plein, et le score est le nombre de lignes éliminées.

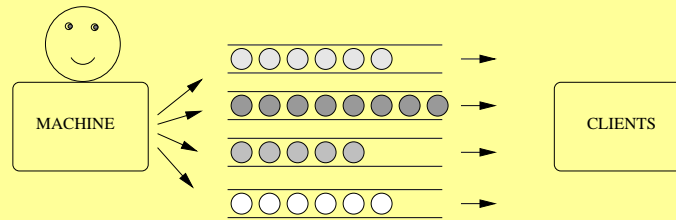
La robotique

- robot jongleur, balancier. . .
- apprentissage de tâches d'assemblage
- robotique mobile, navigation

Applications industrielles

- routage de paquets
- contrôle et maintenance de machines
- contrôle d'une flotte d'ascenseurs
- maintenance d'une constellation de satellites
- conduite de systèmes agricoles

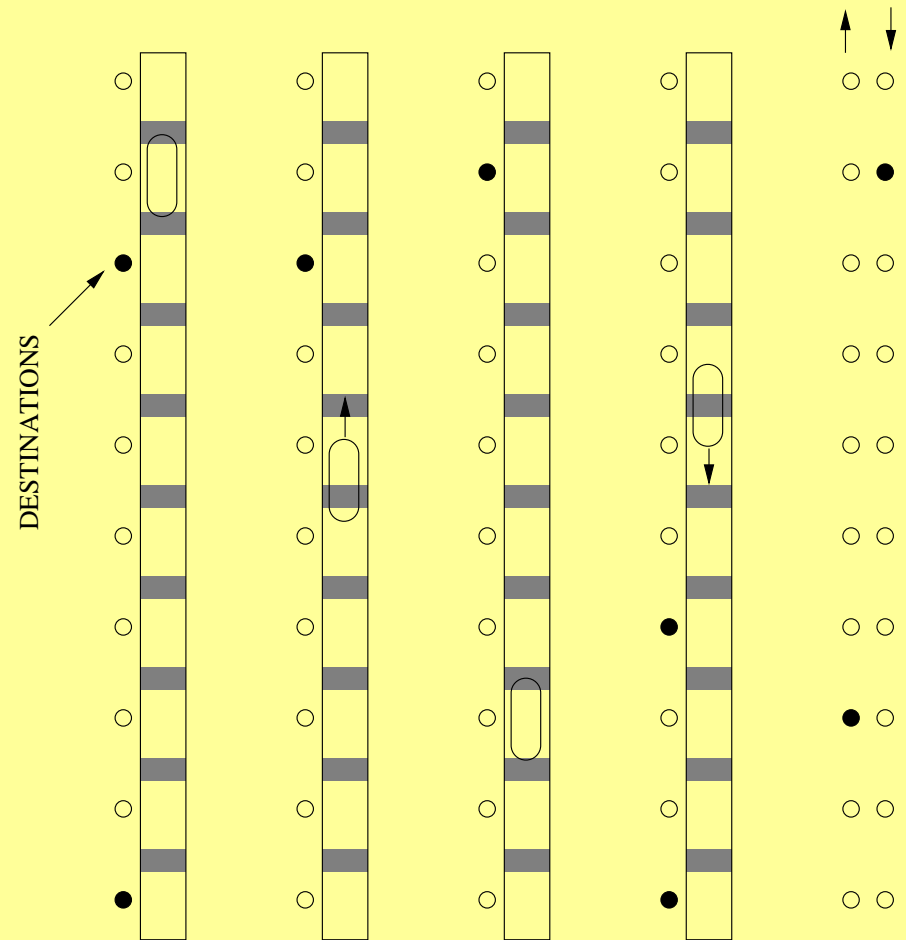
contrôle et maintenance



- la machine produit plusieurs types de produits
- les clients consomment ces produits
- la machine peut tomber en panne, être changée ou réparée
- l'objectif est de maximiser le gain moyen par unité de temps

contrôle d'une flotte d'ascenseurs

- 4 ascenseurs dans un immeuble de 10 étages
- les utilisateurs arrivent selon des lois de Poisson avec des taux fonction du temps.
- l'objectif est de minimiser la somme des temps d'attente au carré



À la recherche d'une politique optimale

Il s'agit là d'un problème d'optimisation complexe

Exemple du Tétris : pour m différentes pièces, une hauteur h et une largeur l , on compte environ $m2^{hl}$ états.

Pour $m = 7$, $h = 20$ et $l = 10$, on a plus de 10^{61} états, et 5 actions, soit $5^{10^{61}} \geq 10^{10^{60}}$ politiques markoviennes possibles !

Principes forts de l'apprentissage par renforcement

Assimilation de l'agent acteur et de l'agent optimisateur

Exploitation de l'information apportée par l'expérience pour guider la recherche de politiques optimales

L'expérience est localisée

Apprentissage ou optimisation ?

Considérons un algorithme génétique où les individus codent des politiques, et où la *fitness* de ces individus est estimée par simulation.

Peut-on parler dans ce cas d'apprentissage par renforcement ?

Pas vraiment, car on parle d'apprentissage par renforcement

- quand il y a évolution de la politique courante à la suite d'expériences, ou d'interactions avec l'environnement,
- c'est à dire quand la politique est considérée comme une fonction des états vers les actions, et non comme une boîte noire

Apprentissage ou optimisation ?

Les méthodes évolutionnistes ne sont pas spécialement bien adaptées au problème de l'apprentissage par renforcement,

sauf lorsque les politiques peuvent être représentées par un petit nombre de paramètres.

Il s'agit d'une tendance récente de l'apprentissage par renforcement

Apprentissage par essais-erreurs

Les différentes méthodes d'apprentissage automatique peuvent être classées selon le type de **professeur** qu'elles considèrent.

Deux principales classes :

- l'apprentissage supervisé
- l'apprentissage non-supervisé

Nous avons affaire ici à un troisième type d'apprentissage, intermédiaire :

l'apprentissage par essais-erreurs.

L'apprentissage supervisé

Les exemples proposés par le professeur sont du type (x, y) où x est la donnée proprement dite, et y est le label que le professeur associe à x .

Le rôle de l'apprentissage est alors de déterminer la règle suivie par le professeur pour associer les labels aux données.

Exemple: apprentissage de fonctions par réseaux de neurones.

L'apprentissage non-supervisé

Le professeur ne fournit que des exemples x , et le problème consiste à apprendre des "régularités" dans l'espace des données, afin d'en effectuer une classification. Le professeur ne supervise pas l'apprentissage.

Exemple: apprentissage par cartes de Kohonen

L'apprentissage par essais-erreurs

le professeur fournit pour chaque association proposée (x, y) une note locale r (la récompense) qui indique approximativement la valeur de ce choix. L'agent est actif à travers le choix des couples états-actions qu'il va tester.

Exemple: l'apprentissage par renforcement

Un modèle formel de la décision séquentielle dans l'incertain :

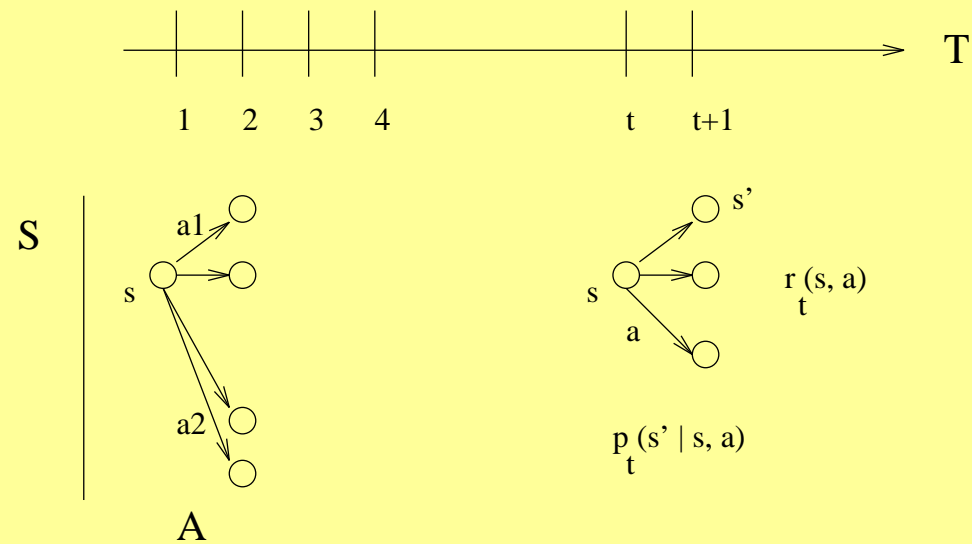
les Processus Décisionnels de Markov

- des espaces d'états et d'actions finis
- des revenus $r_t = r_t(s_t, a_t)$
- une dynamique probabiliste et markovienne

$$s_t, a_t \rightsquigarrow s_{t+1} \text{ selon } p(s_{t+1} \mid s_t, a_t)$$

Définition d'un PDM

$\langle S, A, T, p, r \rangle +$ un critère d'optimalité



$$\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid \pi\right]$$

Le problème du parking

Chaque place est libre ou non avec une probabilité p .

Le conducteur ne peut voir si la place est libre que lorsqu'il est devant.

Il décide alors de se garer ou non.

Chaque place i de N à 1 coûte i , le parking souterrain coûte $C > 0$

Comment minimiser le coût en moyenne ?

Le problème du parking

Etats : (i, L) et (i, \bar{L}) pour $i = 0, \dots, N$, D et F

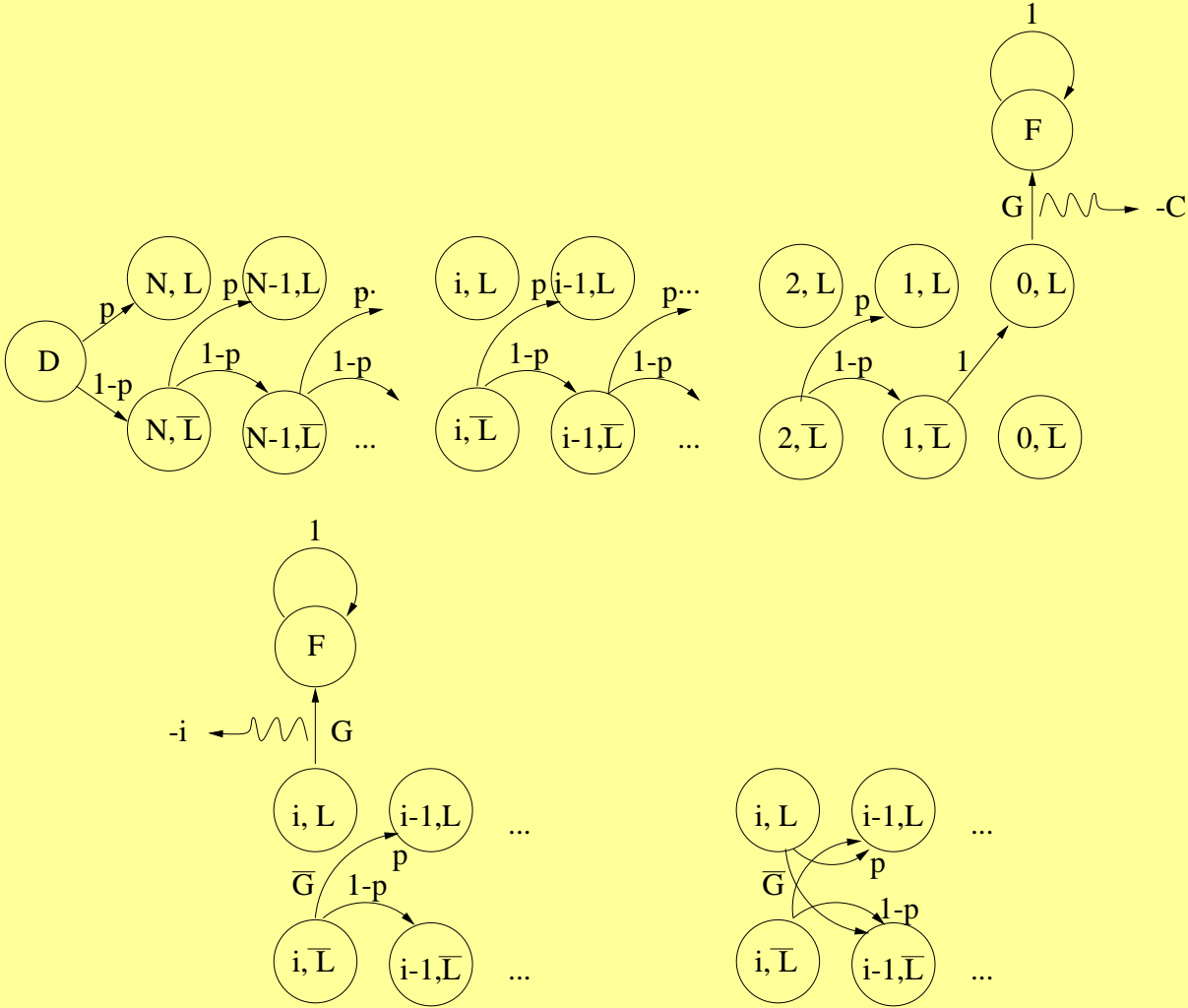
Actions : G et \bar{G}

Probabilités de transitions et récompenses :

$$\begin{aligned}p(F|(i, L) G) &= 1 & i = N, \dots, 0 \\p((i-1, L)|(i, *) \bar{G}) &= p & i = N, \dots, 2 \\p((i-1, \bar{L})|(i, *) \bar{G}) &= 1-p & \text{""} \\p((0, L)|(1, *) \bar{G}) &= 1\end{aligned}$$

$$\begin{aligned}r((i, L), G) &= -i & i = N, \dots, 1 \\r((i, L), \bar{G}) = r((i, \bar{L}), \bar{G}) &= 0 & i = N, \dots, 1 \\r(0, L), G &= -C\end{aligned}$$

Graphe d'états associé



Quelle politique suivre ?

A l'étape N .

- si \bar{L} alors nécessairement \bar{G} , et $\rightarrow N - 1$
- Si L alors
 - si l'on se gare le revenu est $-N$,
 - sinon le meilleur revenu moyen est $V(N - 1)$, et donc
 - si $(-N > V(N - 1))$ on se gare, sinon on continue

Problème : que vaut $V(N - 1)$?

On a

$$V(0) = -C$$

$$V(1) = (1 - p)V(0) + p \max\{-1, V(0)\}$$

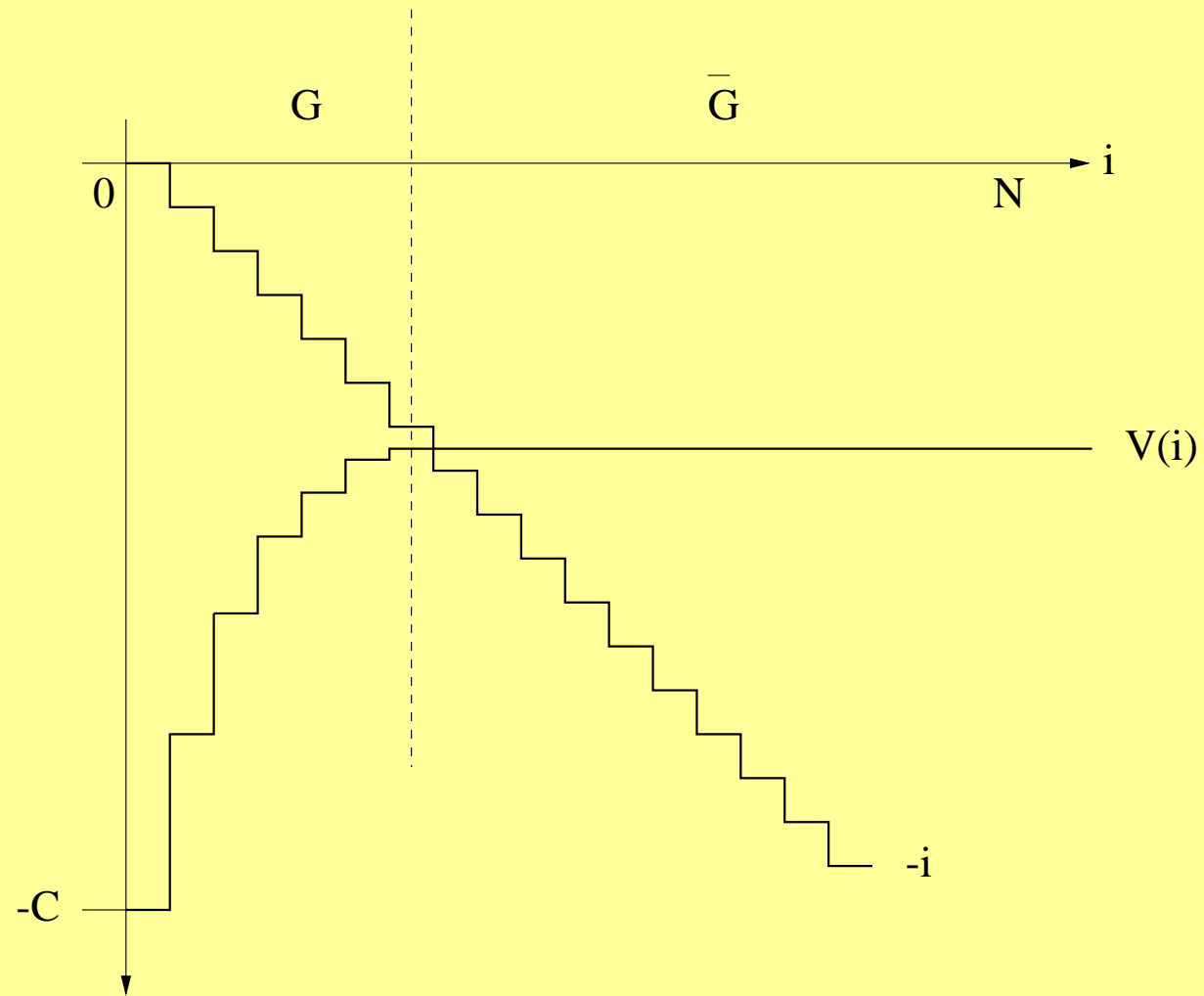
$$V(i) = (1 - p)V(i - 1) + p \max\{-i, V(i - 1)\}$$

et

$$\pi(i, \bar{L}) = \bar{G}$$

$$\pi(i, L) = \begin{cases} G & \text{si } -i > V(i - 1) \\ \bar{G} & \text{si } -i \leq V(i - 1) \end{cases}$$

Fonction de valeur et politique optimale ($p=0.25$)



Fonction de valeur et politique optimale

En résumé :

pour trouver la politique optimale parmi les 2^N politiques possibles, on a calculé en chaque état possible la moyenne des revenus futurs si à partir de cet état on suit jusqu'à la fin la politique optimale.

Une fois cette **fonction de valeur** calculée, il est facile de retrouver la politique optimale.

Fonction de valeur d'une politique

À toute politique π fixée, on associe une fonction de valeur qui vaut en chaque état la moyenne des revenus futurs si à partir de cet état on suit jusqu'à la fin la politique π

$$s \mapsto V_{\pi}^N(s) = E[r_0 + r_1 + \cdots + r_{N-1} \mid s_0 = s]$$

$$V_{\pi}^{\gamma}(s) = E[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots + \gamma^n r_n + \cdots \mid s_0 = s]$$

$$\rho_{\pi}(s) = E\left[\frac{r_0 + r_1 + \cdots + r_{N-1}}{N} \mid s_0 = s\right] \text{ avec } N \rightarrow \infty$$

Autres critères d'optimalité

- (*bias*-optimalité)

$$U_{\pi}(s) = E[(r_0 - \rho^{\pi}(s_0)) + \dots + (r_n - \rho^{\pi}(s_n)) + \dots \mid s_0 = s]$$

- (*n-discount* optimalité, $n = -1, 0, 1, \dots$)

$$\forall \pi, \forall s \quad \lim_{\gamma \rightarrow 1} (1 - \gamma)^{-n} (V_{\pi^*}^{\gamma}(s) - V_{\pi}^{\gamma}(s)) \geq 0$$

$n = -1 \Rightarrow$ gain-optimalité

$n = 0 \Rightarrow$ bias-optimalité

$n = \infty \Rightarrow$ Blackwell-optimalité

- $\forall \pi, \forall s \quad \lim_{N \rightarrow \infty} (V_{\pi^*}^N(s) - V_{\pi}^N(s)) \geq 0$

- Contraintes en fréquences sur S et A , variance minimale, ...

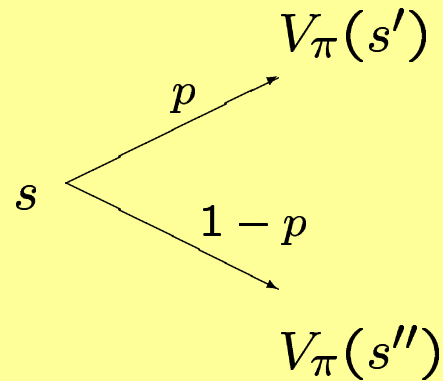
Fonction de valeur et politique optimale

On recherche une politique π^* telle que

$$\forall \pi \quad V_{\pi^*}(s) \geq V_{\pi}(s), \quad \forall s \in S$$

Cette politique existe pour la plupart des critères . . .

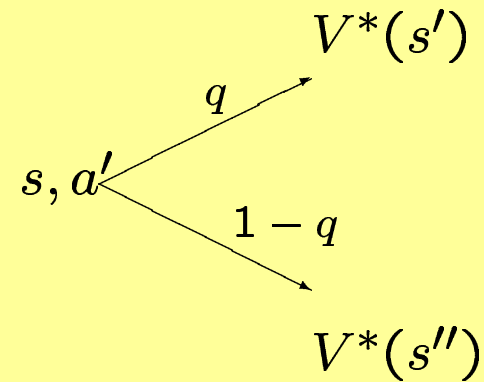
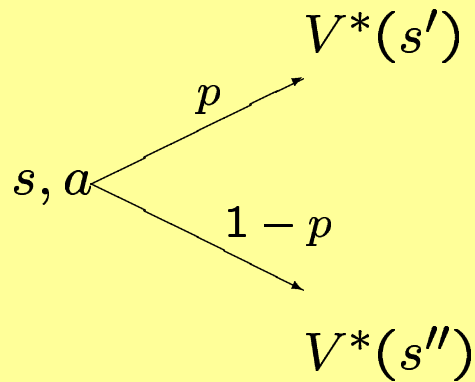
Calcul de la fonction de valeur d'une politique



$$V_\pi(s) = r(s, \pi(s)) + \gamma\{pV_\pi(s') + (1 - p)V_\pi(s'')\}$$

Systeme d'equations lineaires d'inconnues $V_\pi(s)$.

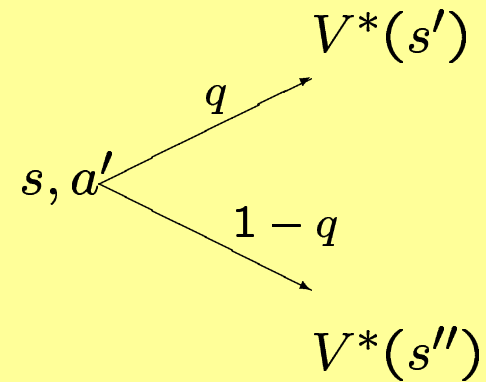
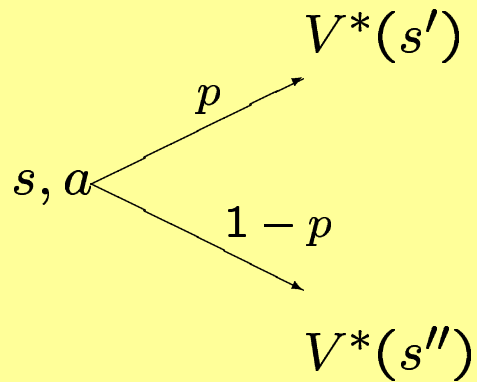
Calcul de la fonction de valeur optimale



$$V^*(s) = \max\left\{ \begin{aligned} &r(s, a) + \gamma\{pV^*(s') + (1 - p)V^*(s'')\}, \\ &r(s, a') + \gamma\{qV^*(s') + (1 - q)V^*(s'')\} \end{aligned} \right\}$$

Systeme d'equations non-lineaires d'inconnues $V^*(s)$.

Détermination de la politique optimale



$$\pi^*(s) = \operatorname{argmax}_{a, a'} \left\{ \begin{array}{l} r(s, a) + \gamma \{ pV^*(s') + (1 - p)V^*(s'') \}, \\ r(s, a') + \gamma \{ qV^*(s') + (1 - q)V^*(s'') \} \end{array} \right\}$$

En résumé

On recherche

$$\pi^* = \operatorname{argmax}_{\pi} V_{\pi}$$

On sait calculer $V^* = \max_{\pi} V_{\pi}$, qui vérifie

$$V^*(s) = \max_a \{r(s, a) + \gamma \sum_{s'} p(s' | s, a) V^*(s')\}$$

On a alors

$$\pi^*(s) = \operatorname{argmax}_a \{r(s, a) + \gamma \sum_{s'} p(s' | s, a) V^*(s')\}$$

Retour sur le parking

Etats : (i, L) et (i, \bar{L}) pour $i = 0, \dots, N$, D et F

Actions : G et \bar{G}

$\gamma = 1$, $V^*(F) = 0$

$$\begin{aligned} V^*(i, \bar{L}) &= p((i-1, L)|(i, \bar{L}) \bar{G})V^*(i-1, L) + p((i-1, \bar{L})|(i, \bar{L}) \bar{G})V^*(i-1, \bar{L}) \\ &= pV^*(i-1, L) + (1-p)V^*(i-1, \bar{L}) \end{aligned}$$

$$\begin{aligned} V^*(i, L) &= \max\{-i + V^*(F), pV^*(i-1, L) + (1-p)V^*(i-1, \bar{L})\} \\ &= \max\{-i, pV^*(i-1, L) + (1-p)V^*(i-1, \bar{L})\} \end{aligned}$$

Retour sur le parking

Notons $V^*(i) = pV^*(i, L) + (1 - p)V^*(i, \bar{L})$

$$V^*(i, \bar{L}) = V^*(i - 1)$$

$$V^*(i, L) = \max\{-i, V^*(i - 1)\}$$

$$V^*(i) = p \max\{-i, V^*(i - 1)\} + (1 - p)V^*(i - 1)$$

$$\pi(i, \bar{L}) = \bar{G}$$

$$\begin{aligned} \pi(i, L) &= \operatorname{argmax}\{-i, V^*(i - 1)\} \\ &= \begin{cases} G & \text{si } -i > V^*(i - 1) \\ \bar{G} & \text{si } -i \leq V^*(i - 1) \end{cases} \end{aligned}$$

Algorithmes de programmation dynamique

- *value iteration*
- *policy iteration*
- programmation linéaire

value iteration

On cherche à résoudre l'équation de Bellman en $V^*(s)$ par une méthode itérative de type point-fixe.

On fixe V_0 , puis

$$\forall s \quad V_{n+1}(s) = \max_a \{r(s, a) + \gamma \sum_{s'} p(s' | s, a) V_n(s')\}$$

Il y a convergence de V_n vers V^* , d'où l'on déduit π^* .

policy iteration

On utilise la propriété suivante :

Soit π une politique. Alors la nouvelle politique π^+ définie par

$$\pi^+(s) = \operatorname{argmax}_a \{r(s, a) + \gamma \sum_{s'} p(s' | s, a) V_\pi(s')\}$$

est une meilleure politique que π : $\forall s \quad V_{\pi^+}(s) \geq V_\pi(s)$.

On fixe donc π_0 , puis

$$\forall s \quad \pi_{n+1}(s) = \operatorname{argmax}_a \{r(s, a) + \gamma \sum_{s'} p(s' | s, a) V_{\pi_n}(s')\}$$

Il y a convergence de π_n vers π^* .

Asynchronous value iteration

Méthode de Gauss Seidel : on met à jour les états un par un, du premier au dernier, en intégrant à chaque étape les corrections précédentes

Asynchronous value iteration : les états sont choisis aléatoirement.

Il y a convergence de V_n vers V^* .

modified policy iteration

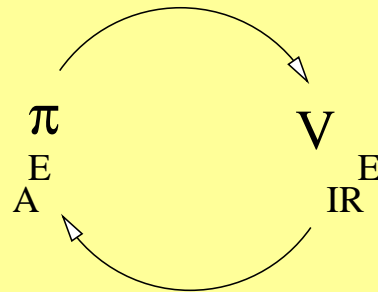
La partie évaluation de la *policy iteration* n'est pas effectuée complètement, mais uniquement sur quelques itérations de l'opérateur linéaire.

On peut aussi coupler cela avec une mise à jour d'un seul état à chaque itération (*asynchronous policy iteration*).

Il y a convergence de V_n vers V^* .

En résumé

évaluation



amélioration

L'équation de Bellman décrit le point fixe de ce cycle dans l'espace des fonctions de valeur.

Les différents algorithmes de programmation dynamique exploitent cette équation pour calculer ce point fixe.