Apprentissage par renforcement

PESTO 2011

TELECOM ParisTech

Plan

Cadre général

- Objectifs
- Quelques exemples et réalisations
- Mesures de performance
- “Théorie du Wishful Thinking”

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l’algorithme KL-UCRL

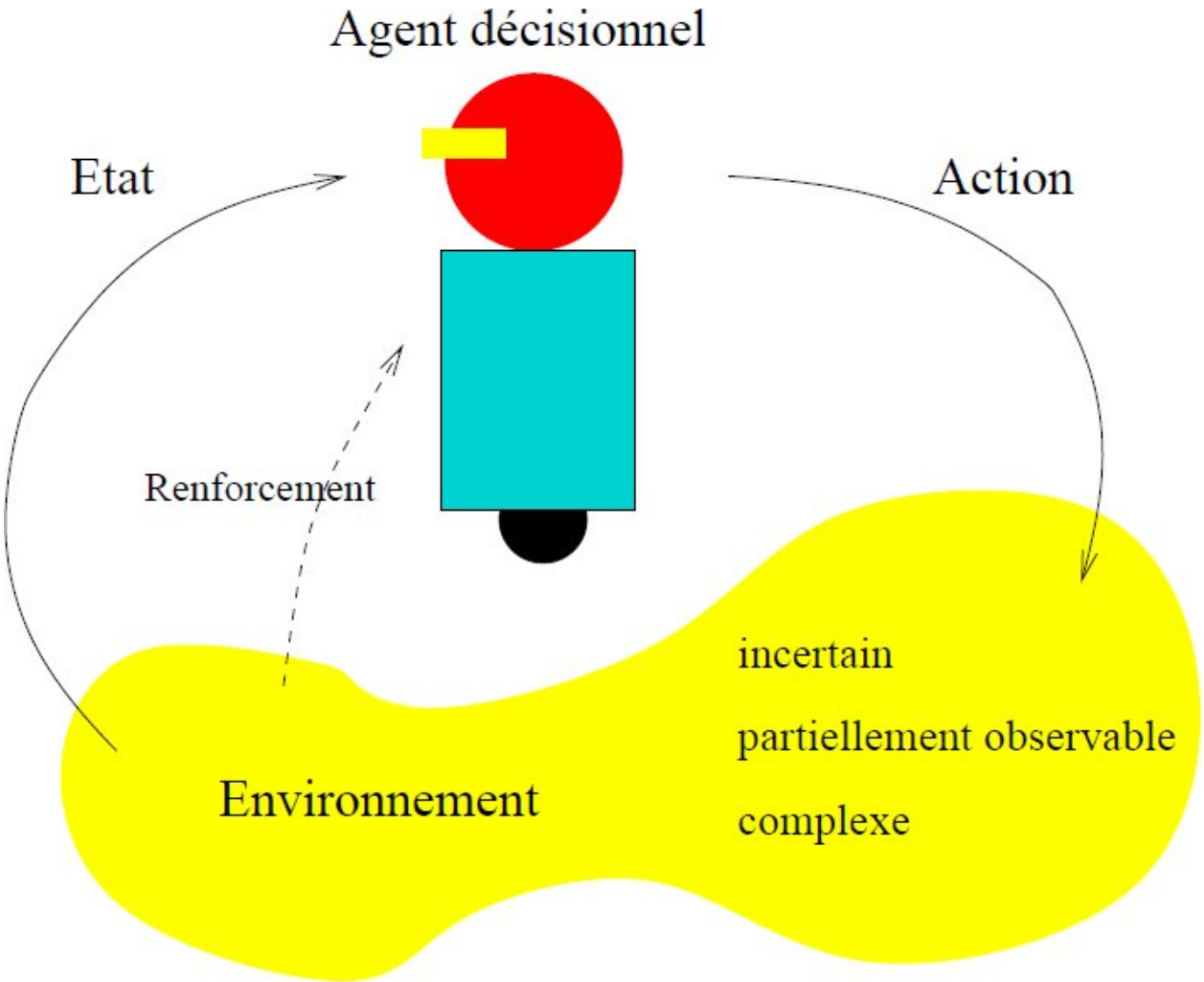
Les différents types d'apprentissage

Apprentissage supervisé : à partir de l'observation de données $(X_t, Y_t)_t$ où $Y_t = f(X_t) + \varepsilon_t$ et f est la fonction cible (inconnue), estimer f afin de faire des prédictions de f(x)

Apprentissage non-supervisé : à partir de données $(X_t)_t$, trouver des structures dans ces données (ex. des classes), estimer des densités, ...

Apprentissage par renforcement : les données arrivent au fur et à mesure des décisions à prendre

Cadre général de l'apprentissage par renforcement



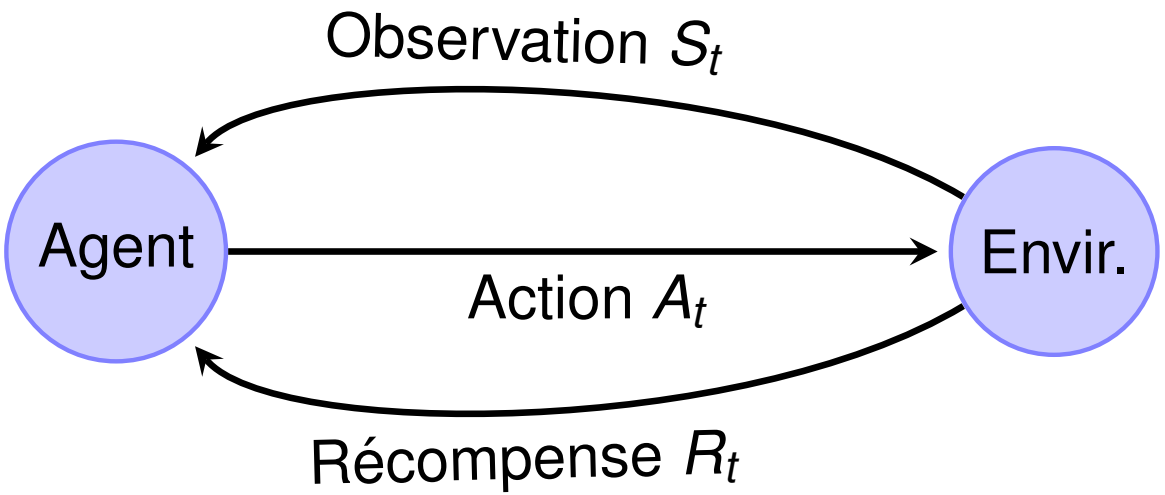
Objectifs de l'Apprentissage par Renforcement

Acquisition automatisée de compétences pour la prise de décisions (actions ou contrôle) en milieu complexe et incertain.

Apprendre par l'expérience une stratégie comportementale (appelée politique) en fonction des échecs ou succès constatés (les renforcements ou récompenses).

Exemples : apprentissage sensori-moteur, jeux (backgammon, échecs, poker, go), robotique mobile autonome, gestion de portefeuille, recherche opérationnelle,. . .

RL : première formalisation



dilemme
exploration
|
exploitation

- L'agent est acteur et pas spectateur [Sutton '92; Bertsekas '95]
- A chaque instant t, il choisit une action $A_t \in A$ en fonction des observations et récompenses passés $(S_s, R_s)_{s<t}$ pour maximiser la récompense cumulée $\sum_{t=1}^{n} R_t$
- Exemples: essais médicaux, robotique, proposition de contenu, finance, publicité, internet mobile, ...

Historique

Naissance du domaine : Rencontre fin années 1970 entre

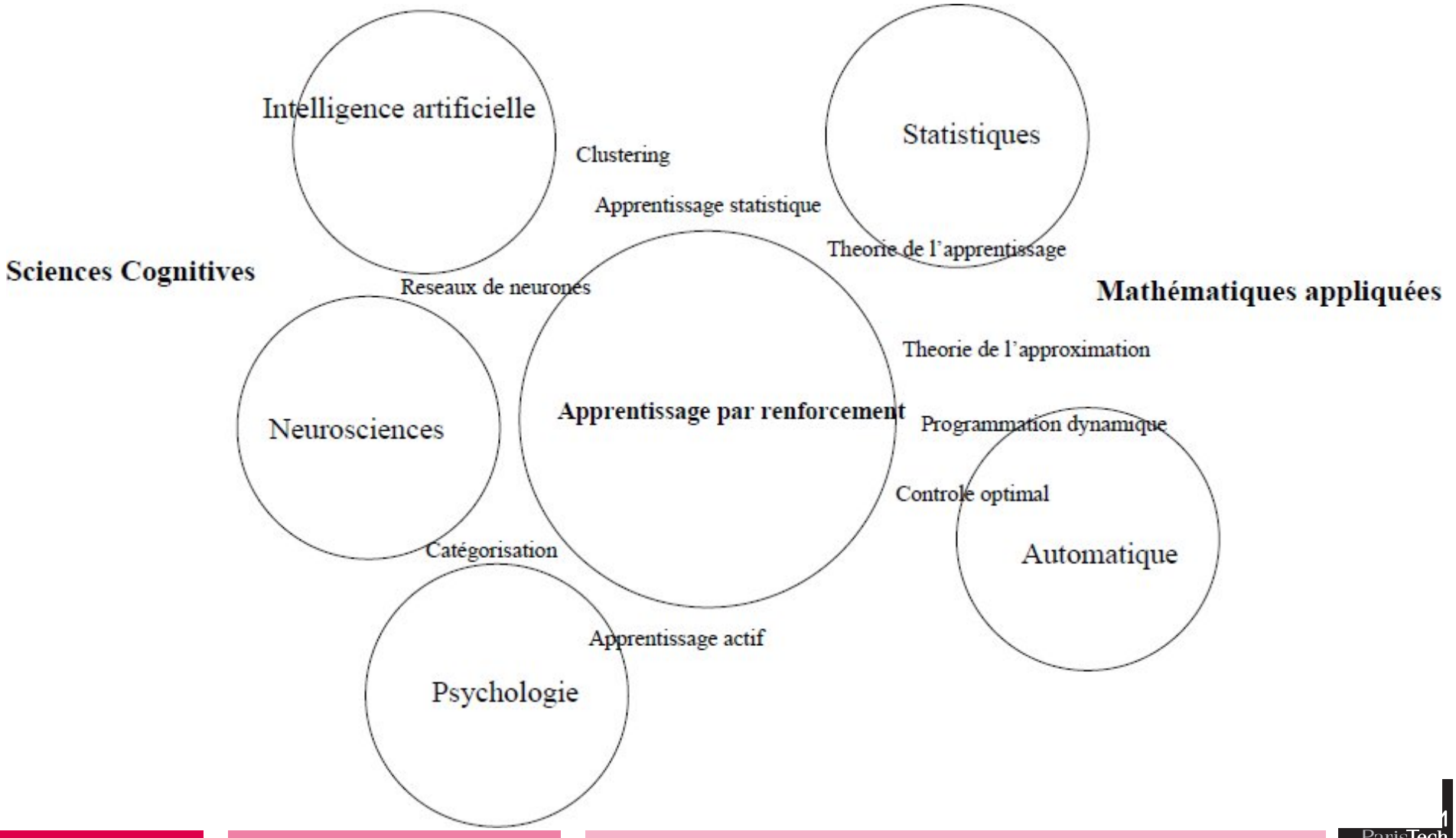
- Neurosciences computationnelles. Renforcement des poids synaptiques des transmissions neuronales (règle de Hebbs, modèles de Rescorla et Wagner dans les années 60, 70). Renforcement = corrélations activités neuronales.
- Psychologie expérimentale. Modèles de conditionnement animal: renforcement de comportement menant à une satisfaction (recherches initiées vers 1900 par Pavlov, Skinner et le courant béhavioriste). Renforcement = satisfaction, plaisir ou inconfort, douleur.
- Cadre mathématique adéquat: Programmation dynamique de Bellman (années 50, 60), en théorie du contrôle optimal. Renforcement = critère à maximiser.

Psychologie expérimentale

Loi des effets (Thorndike, 1911)

> *Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond"*

Domaine mutlidisciplinaire



L'environnement

Déterministe ou stochastique (ex: backgammon)

Hostile (ex: jeu d'échecs) ou non (ex: jeu Tétris)

Partiellement observable (ex: robotique mobile)

Connu ou inconnu (ex: vélo) de l'agent décisionnel

Le renforcement

Peut récompenser une séquence d'actions

$\Longrightarrow$ problème du "credit-assignment" : quelles actions doivent être accréditées pour un renforcement obtenu au terme d'une séquence de décisions?

Comment sacrifier petit gain à court terme pour privilégier meilleur gain à long terme?

$\Longrightarrow$ Dilemme exploration / exploitation

Plan

Cadre général

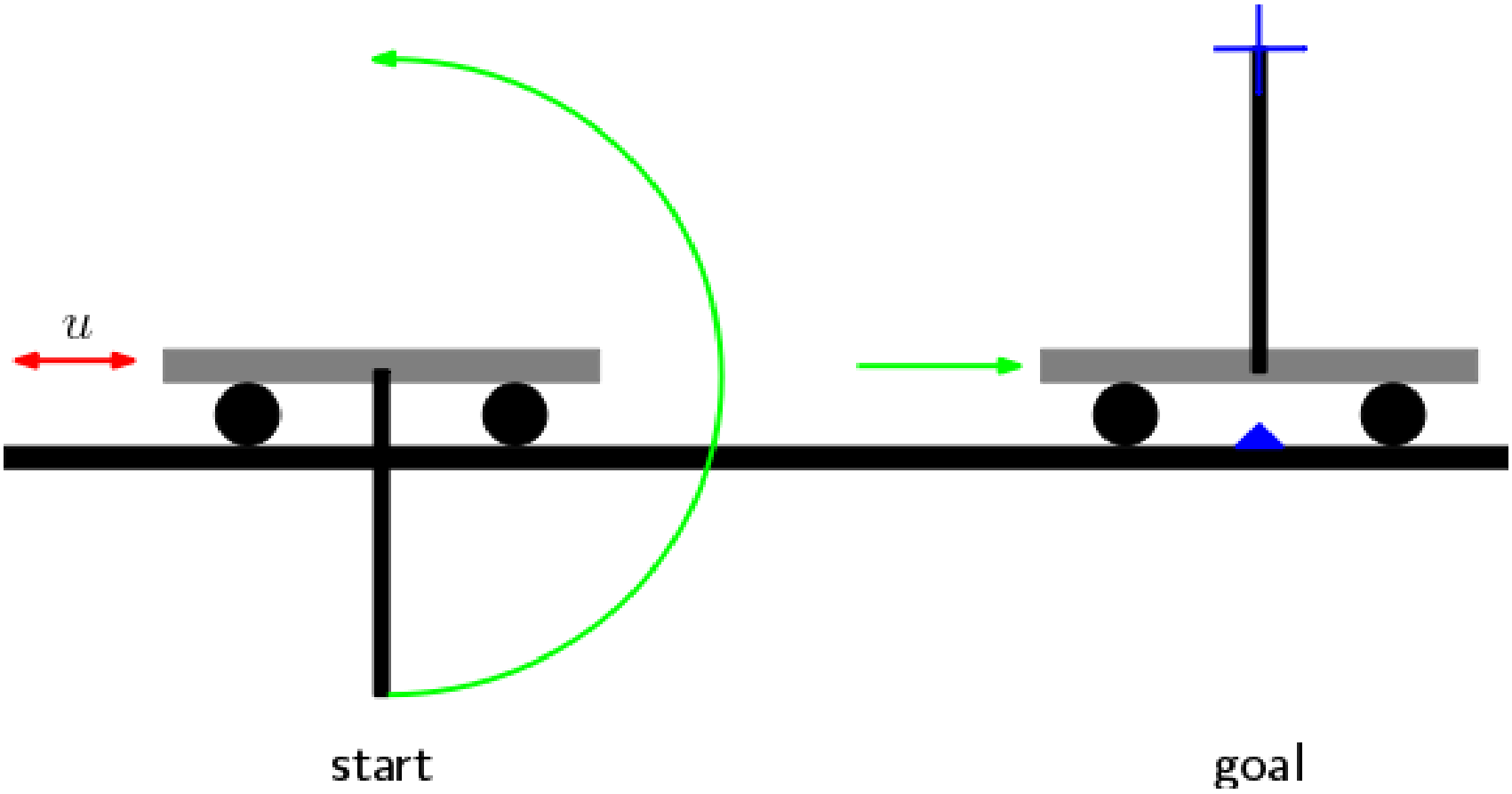
- Objectifs
- Quelques exemples et réalisations
- Mesures de performance
- “Théorie du Wishful Thinking”

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l’algorithme KL-UCRL

Exemple : pendule inversé



L'algorithme d'apprentissage utilisé par Martin est Neural Fitted Q iteration, une version de fitted Q-iteration où des réseaux de neurones sont utilisés comme approximateur de fonction.

Réalisations 1/2

- TD-Gammon. [Tesauro 1992-1995]: jeu de backgammon. Produit le meilleur joueur mondial!
- KnightCap [Baxter et al. 1998]: jeu d'échec ('2500 ELO)
- Computer poker (calcul d'un équilibre de Nash avec bandits adversarials), [Alberta, 2008]
- Computer go (algorithmes de bandits hiérarchiques), [Mogo, 2006]
- Robotique: jongleurs, balanciers, acrobots, ... [Schaal et Atkeson, 1994]
- Robotique mobile, navigation: robot guide au musée Smithonian [Thrun et al., 1999]

Réalisations 2/2

- Commande d'une batterie d'ascenseurs [Crites et Barto, 1996]
- Routage de paquets [Boyan et Littman, 1993]
- Ordonnancement de tâches [Zhang et Dietterich, 1995]
- Maintenance de machines [Mahadevan et al., 1997]
- Réseaux sociaux [Acemoglu et Ozdaglar, 2010]
- Yield Management, pricing des places d'avion [Gosavi 2010]
- Prévision de charge et gestion électrique [S. Meynn, 2010]

Références

[Puterman ’94] Markov Decision Processes, Discrete Stochastic Dynamic Programming

[Bertsekas ’95] Dynamic Programming and Optimal Control

[Sutton & Barto ’98] Reinforcement Learning

[Sigaud & Buffet ’08] Processus Décisionnels de Markov en Intelligence Artificielle

[Cesa-Bianchi & Lugosi ’06] Prediction, Learning, and Games

Plan

Cadre général

- Objectifs
- Quelques exemples et réalisations
- Mesures de performance
- “Théorie du Wishful Thinking”

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l’algorithme KL-UCRL

Consistance

Une stratégie est dite *consistante* si elle permet de trouver, en un temps fini, la politique optimale quelles que soit le problème.

Force : exige de trouver *exactement* la solution (et pas une solution approchée)

Faiblesse : on ne contrôle pas du tout ce qu'on a perdu pendant la phase d'apprentissage

Bornes PAC

PAC = “Probably Approximately Correct”

La *complexité* d’une stratégie est, pour un ε donné, le temps qu’il lui faut pour identifier une stratégie ε-optimale

Une stratégie est dite PAC-MDP (Probably Approximately Correct in Markov Decision Processes) si, pour tous ε et δ, sa complexité est bornée par un polynôme en $1/\varepsilon$ et en les paramètres du problème avec probabilité au moins $1-\delta$.

Regret

Le regret est, de manière générale, défini comme la différence entre la somme des récompenses reçues avec une stratégie et la récompense *oracle* qu'accumulerait, dans le même temps, un agent connaissant la politique optimale

Dans les recherches, différentes variantes sont étudiées par commoditié (regret moyen, moyennes conditionnelles, etc.)

Cette mesure est plus exigeante : elle prend en compte la performance d'une stratégie *dès les premiers instants* (pas de burn-in)

Plan

Cadre général

- Objectifs
- Quelques exemples et réalisations
- Mesures de performance
- “Théorie du Wishful Thinking”

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l’algorithme KL-UCRL

Le paradigme optimiste

Algorithmes **optimistes** : [Lai&Robins ’85; Agrawal ’95]

Fais comme si tu te trouvais dans l’environnement qui t’est le plus favorable parmi tous ceux qui rendent les observations suffisamment vraisemblables

D’abord présenté dans un contexte bandit, puis largement généralisé ces dernières années

Popriétés

De façon plutôt inattendue, les méthodes optimistes se révèlent :

- pertinentes dans des cadres très différents
- efficaces
- robustes
- simples à mettre en oeuvre

cf. les exemples ci-après.

Plan

Cadre général

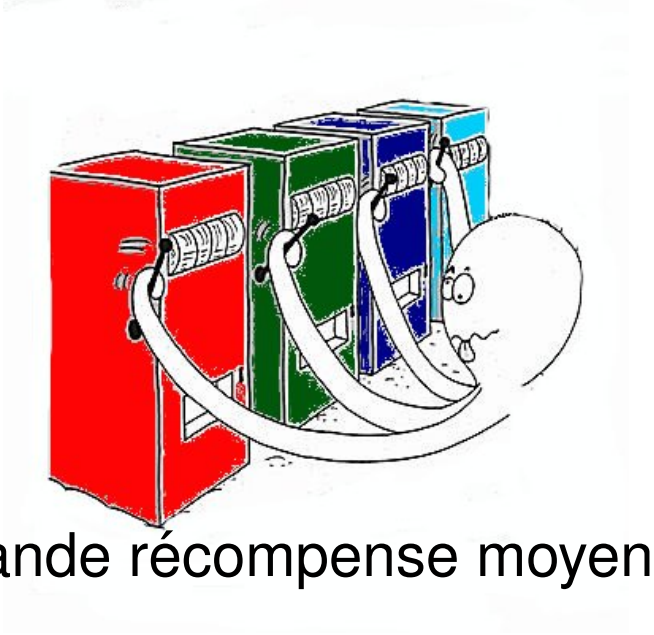
Problèmes de bandits

- Le modèle
- Différentes solutions
  - Premières approches
  - Soft-Max : EXP3
  - Paradigme optimiste : UCB
  - La solution des martingales
- Optimalité
- Application originale : Good-UCB

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

Formulation en Apprentissage par renforcement



- Environnement constant
- Conditionnellement aux actions $(A_t)_{1\leq t\leq n}$, les récompenses $(R_t)_{1\leq t\leq n}$ sont i.i.d. de moyenne μ_{A_t}
- But : jouer l'action a^* qui a la plus grande récompense moyenne :

$$\mu_{a^*} = \max_{a\in A} \mu_a$$

- Mesure de performance : *regret cumulé* (en moyenne conditionnelle)

$$\text{Regret}(n) = \sum_{t=1}^{n} \mu_{a^*} - \mu_{A_t}$$

Plan

Cadre général

Problèmes de bandits

- Le modèle
- Différentes solutions
  - Premières approches
  - Soft-Max : EXP3
  - Paradigme optimiste : UCB
  - La solution des martingales
- Optimalité
- Application originale : Good-UCB

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

Algorithmes ε-Greedy

Idée : on joue le meilleur bras avec probabilité $1-\varepsilon$ (greedy = glouton), et un bras au hasard avec probabilité ε

Pour une valeur (ou une suite de valeur) bien choisie de ε, l'algorithme est consistant et on peut prouver des bornes de regret

Cependant, c'est très souvent sous-optimal (c'est en général la première approche d'un problème)

La solution de Gittins

[Gittins '79] Bandit Processes and Dynamic Allocation Indices

Politique d'indice : on associe à chaque bras un indice de performance et on choisit celui qui a le plus grand indice

Préfigure les méthodes optimistes pour les MDP, cf. plus loin

Algorithme EXP3.P

Cf prédiction de séquence individuelles avec perte $M - R_t^a$. Estimation des pertes (non observées) :

$$\hat{\ell}_t(a) = \frac{M - R_t^a}{p_t(a)} \mathbb{1}_{\{A_t = a\}}$$

estimateur *sans biais* de $M - R_t^a$

On estime ainsi les pertes cumulées

$$\hat{L}_t(j, \mathbf{y}_t)) = \sum_{s=1}^{t} \hat{\ell}_t(a))$$

EXP3.P = stratégie randomisée avec comme choix de pondération :

$$\hat{p}_t(j) = \frac{\exp(-\beta \hat{L}_{t-1}(j, \mathbf{y}_{t-1}))}{\sum_k \exp(-\beta \hat{L}_{t-1}(k, \mathbf{y}_{t-1}))}$$

Borne de regret pour EXP3.P

Théorème: En choisissant

$$\beta = 1/M\sqrt{\frac{2\log(N)}{nM}}$$

le regret de l'algorithme EXP3.P face à la meilleure stratégie constante vérifie :

$$\mathbb{E}\left[R_n(\hat{p})\right] \leq M\sqrt{2nN\log(N)}$$

En pratique, très robuste mais peu véloce à se concentrer sur le bon bras quand il y en a un.

Upper Confidence Bound (UCB)

- Algorithmes optimistes : [Lai&Robins '85; Agrawal '95]

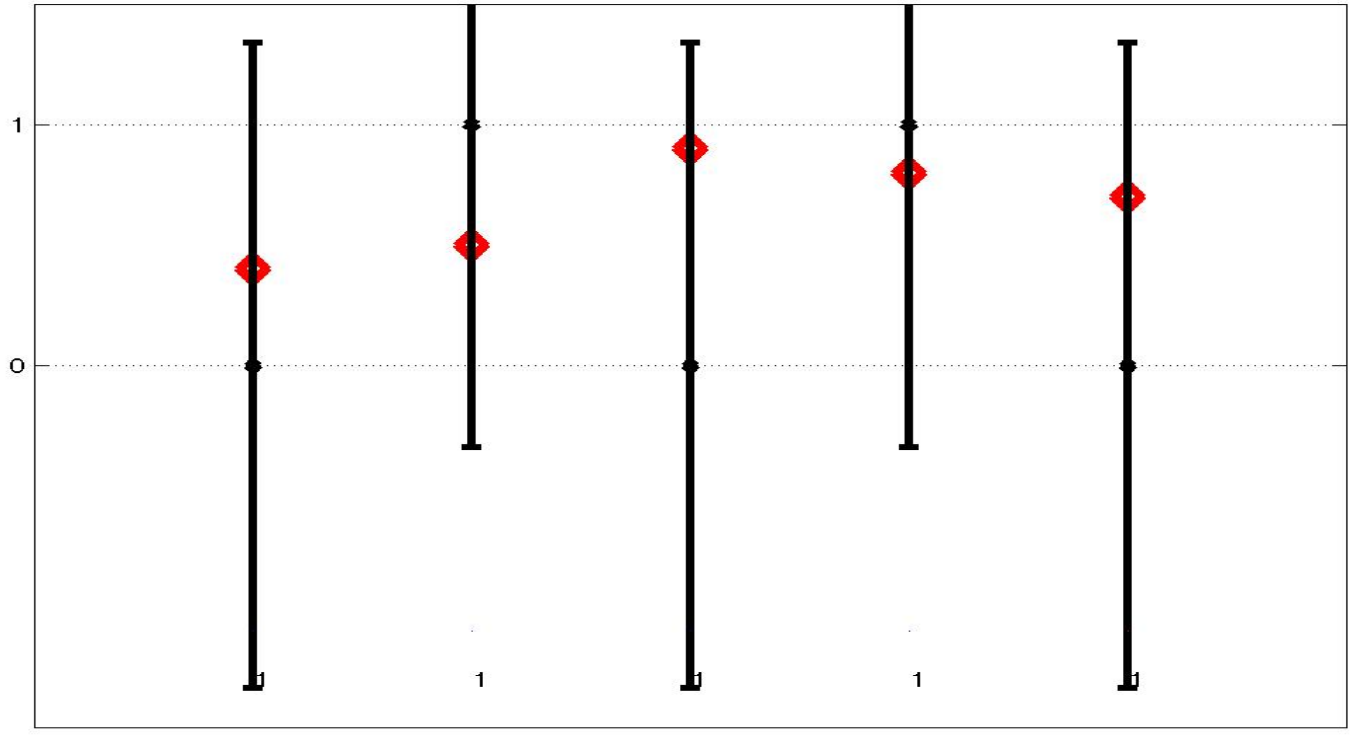
  Fais comme si tu te trouvais dans l'environnement qui t'est le plus favorable parmi tous ceux qui rendent les observations assez vraisemblables

- Ici : UCB (Upper Confidence Bound) = établir une borne supérieure de l'intérêt de chaque action, et choisir celle qui est la plus prometteuse [Auer&al '02; Audibert&al '07]
- Avantage : comportement facilement interprétable et "acceptable"

  $\Rightarrow$ le regret grandit comme $C\log(n)$, où C dépend de

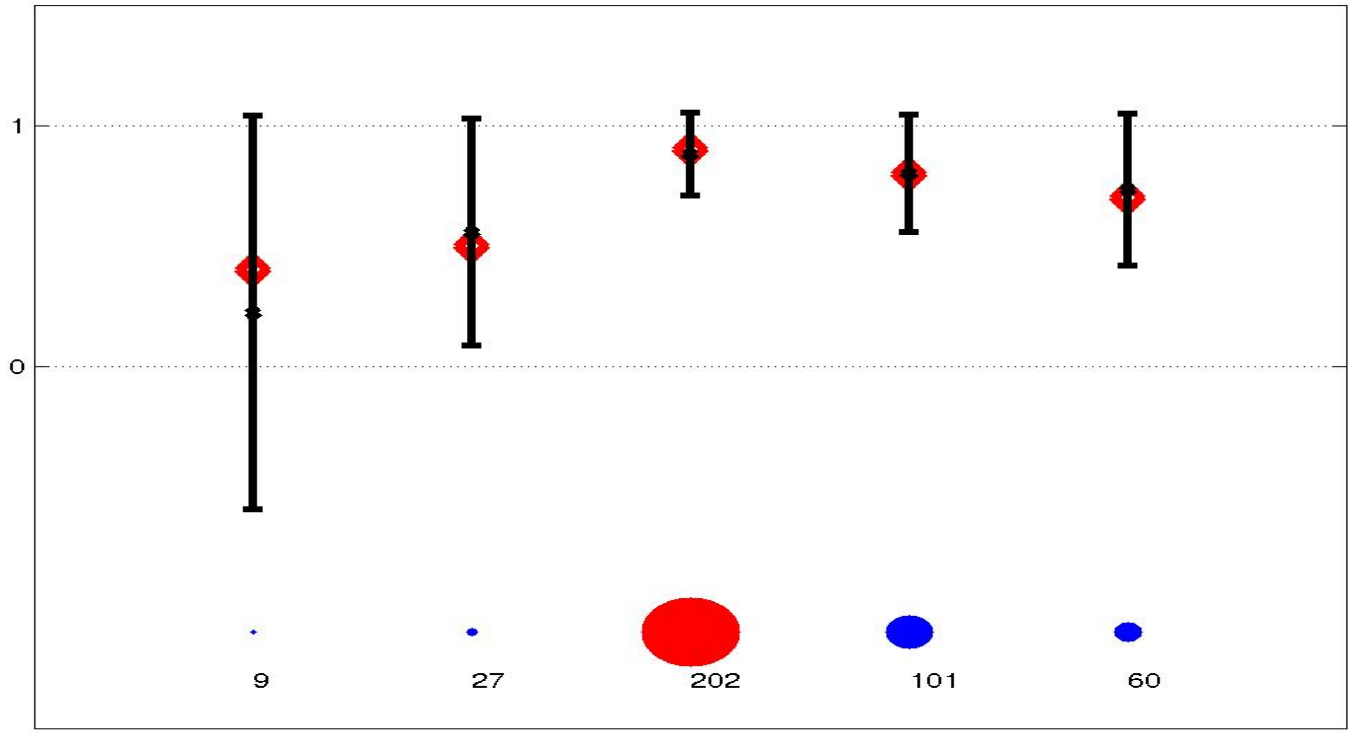
$$\Delta = \min_{\mu_a < \mu_{a^*}} \mu_{a^*} - \mu_a$$

UCB en action



Début

UCB en action



Début

Preuve

UCB tend à aligner les bornes supérieures des intervalles de confiance

On borne le nombre de fois que le bras sous-optimal j est joué

On travaille conditionnellement aux intervalles de confiance

Un bras peu performant ne peut pas avoir une borne haute de confiance durablement au dessus de celle du meilleur bras

Un problème à deux bandits

[Lamberton & Pagès & Tarrès '04] When can the two-armed Bandit algorithm be trusted ?

Modèle : un fond est géré par deux traders A et B, qui ont chaque jour la responsabilité d'un pourcentage du fond variable de jour en jour.

On veut allouer le plus vite possible la gestion au meilleur des deux

Chaque jour, un des deux traders est évalué (avec une probabilité proportionnelle à la part qu'il gère) :

- s'il a bien performé, on augmente sa part du fond
- sinon, on ne le punit pas (on laisse les parts inchangées)

Question : va-t-on finir par confier tout le fond au meilleur trader ?

Formalisation mathématique

On note X_n la portion du fond géré par le trader A. On suppose $X_0 = x \in]0,1[$.

On note E_n (resp. F_n) l'événement "le trader A (resp. B) sur-performe le jour n", et on suppose que les $\{E_n, F_n\}$ sont indépendants.

On note γ_{n+1} la portion de ce que gère B que gagnera le trader A s'il est évalué et qu'il sur-performe le jour n :on suppose $\gamma_n \in]0,1[$ et $\sum_n \gamma_n = \infty$.

$$X_{n+1} = X_n + \gamma_{n+1}\left((1-X_n)\mathbb{1}_{\{U_{n+1}\leq X_n\}\cap E_{n+1}} - X_n \mathbb{1}_{\{\{U_{n+1}\leq X_n\}\}\cap E_n}\right)$$

où U_n désigne sur suite de v.a. i.i.d. uniformes sur $[0,1]$.

Résultats

Théorème :

- X_n converge p.s. vers $X_\infty \in \{0, 1\}$.
- si $0 < P(F_n) < P(E_n) \leq 1$, $P(X_\infty = 0)$ peut être non nulle si γ_n ne décroit pas assez vite vers 0. En particulier, si $\gamma_n = \left(\frac{C}{n+C}\right)^\alpha$:
  - si $0 < \alpha < 1$ ou si ($\alpha = 1$ et $C > 1/P(F_n)$), alors $P(X_\infty = 0) > 0$
  - si $\gamma_n = \gamma$, $P(X_\infty = 0) \geq (1 - x)^{1/\gamma P(F_n)}$
  - si $\alpha = 1$ et $C < 1/P(F_n)$, alors $P(X_\infty = 0) = 0$
- si $0 < P(F_n) = P(E_n) \leq 1$, $P(X_\infty = 1) = x$

Résultats fins découlant de la théorie des *martingales*

Plan

Cadre général

Problèmes de bandits

- Le modèle
- Différentes solutions
  - Premières approches
  - Soft-Max : EXP3
  - Paradigme optimiste : UCB
  - La solution des martingales
- Optimalité
- Application originale : Good-UCB

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

La borne inférieure de Lai&Robbins

Notons $KL(p_j|p^*)$ la divergence de Kullback-Leibler entre la loi du j-ième bras et le bras optimal

Theorème : pour toute stratégie jouant toujours "suffisamment" le bras optimal, et pour tout bras sous-optimal j, le nombre de fois qu'est joué le bras j est borné inférieurement en espérance :

$$\mathbb{E}[T^j(n)] \geq \frac{\log(n)}{KL(p_j|p^*)}$$

Corollaire : toute stratégie a un regret au moins en $C\log(n)$, où C dépend de la distribution des bras;

Borne inférieure minimax

Théorème : Pour n et N assez grands, on peut construire un problème de bandit pour lequel le regret de n'importe quelle stratégie est au moins

$$\frac{1}{20}\sqrt{Nn}$$

Remarque : l'analyse d'UCB permet de borner le regret par

$$C\sqrt{n\log(n)}$$

pour une certaine constante C ne *dépendant pas* du problème.

Plan

Cadre général

Problèmes de bandits
- Le modèle
- Différentes solutions
  - Premières approches
  - Soft-Max : EXP3
  - Paradigme optimiste : UCB
  - La solution des martingales
- Optimalité
- Application originale : Good-UCB

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

Identification de pannes

Détection de panne dans les réseaux électriques

Nombre de circuits possibles $\approx 10^{50}$

N générateurs aléatoires de réseaux plus ou moins focalisés sur les configurations en panne

On dispose d'un simulateur capable de détecter une panne, mais chaque appel est assez lourd

Problème : comment utiliser efficacement nos N simulateurs pour trouver le plus vite possible un grand nombre de configurations de pannes

Modélisation simplifiée

Pour $1 \leq j \leq N$, $(X_t^j)_t$ i.i.d. uniformes sur $\{1, \ldots, m\}$

Les configurations de panne sont $\{1, \ldots, M_j\}$

A chaque instant t, on choisit la distribution J_t et on tire $X_t^{J_t}$

But : pour un budget n donné, maximiser

$$\sum_{j=1}^{n} \# \{1, \ldots, M_j\} \cap \left\{ X_t^j : J_t = j \right\}$$

Problème dual : minimiser le temps qu'il faut pour retrouver tout (ou telle partie) des pannes

Un problème de bandits ?

Analogies :

- problème de décisions séquentielles
- N bras
- récompenses $R_t = 1$ si $X_t^{J_t} \leq M_{J_t}$ *et s'il n'a encore jamais été vu*

MAIS une grosse différence : une "récompense" n'encourage pas forcément à recommencer la même action ! Au contraire : une fois un simulateur j épuisé, il faut s'en détourner !

$\Longrightarrow$ Paramètre d'un simulateur $r_t^j = R_t^j/m$, où

$$R_t^j = \#\{1, \dots, M_j\} \setminus \left\{X_t^j : J_t = j\right\}$$

évolue au cours du temps (*bandit non stationnaire*)

Une solution optimiste

Estimation de la masse manquante : Good-Turing ['53]

$$\widehat{R}_t^j = \#\{i \in \{1,\dots,m\} : \sum_{t:J_t=j} \mathbb{1}_{X_j^j=1} = 1\}$$

Borne de [McAllester-Schapire '97]

$$P\left(r_t^j > \frac{\widehat{R}_t^j}{m} + \left(2\sqrt{2}+\sqrt{3}\right)\sqrt{\frac{\log(3/\delta)}{m}} \right) \leq \delta$$

Très comparables aux inégalités de concentration habituelles

Good-UCB

Algorithme optimiste : borne de confiance supérieure pour la masse manquante de chaque simulateur
Si

$$N_t(j) = \sum_{s=1}^{t} \mathbb{1}_{J_s=j}$$

désigne le nombre de tirage avec le simulateurs j jusqu'à l'instant t,
Good-UCB choisit :

$$J_{t+1} = \operatorname*{argmax}_j \frac{\widehat{R}_t^j}{m} + c\sqrt{\frac{\log(3t)}{N_t(j)}}$$

Performance, bornes de regret, améliorations, généralisations :

work in progress. . .

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens
- Définitions
- Exemple: maintenance d'un stock
- Le problème de planning
- Et si on ne connaît pas l'environnement

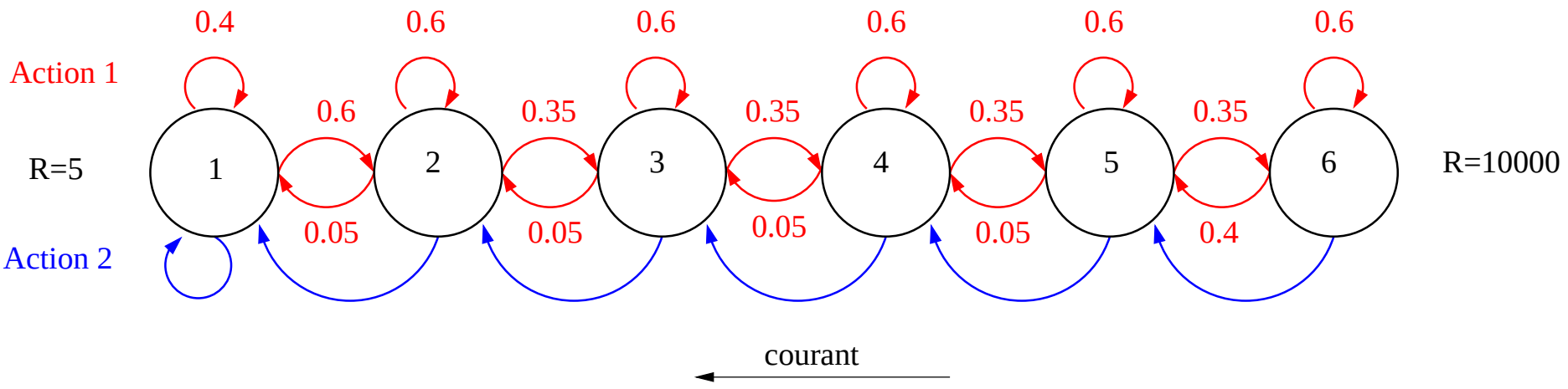
Une méthode indirecte : l'algorithme KL-UCRL

Le modèle 1/2

Le système est dans un état S_t qui évolue de façon markovienne :

$$S_{t+1} \sim P(\cdot; S_t, A_t) \text{ et } R_t = r(S_t, A_t) + \varepsilon_t$$

Exemple / Benchmark : RiverSwim [StrehlLittman'08]



Le modèle 2/2

[Bellman 1957, Howard 1960, Dubins et Savage 1965, Fleming et Rishel 1975, Bertsekas 1987, Puterman 1994]

Défini par $(\mathcal{S}, \mathcal{A}, p, r)$, où:

- $\mathcal{S}$ espace d'états (supposé fini ici mais peut être dénombrable, continu)
- $\mathcal{A}$ espace d'actions (supposé fini aussi)
- $p(y|x,a)$: probabilités de transition d'un état $x \in \mathcal{S}$ à $y \in \mathcal{S}$ lorsque l'action a est choisie:

$$p(y|x,a) = P(X_{t+1} = y | X_t = x, A_t = a)$$

- $r(x,a,y)$: récompense obtenue lors de la transition de l'état x à y en ayant choisi l'action a.

Politique

Politique π = une règle de décision, une stratégie comportementale, qui détermine, à un instant donné quelle action doit être choisie.

On distingue deux types de politiques :

- *déterministe* : $\pi : \mathcal{S} \to A$
  $\pi(x)$ = action choisie en x.
- *stochastique* : $\pi : \mathcal{S} \to \mathfrak{M}_1(A)$
  $\pi(a|x)$ = probabilité de choisir a en x.

Valeur d'une politique

Horizon temporel fini :

$$V^{\pi}(x,t) = E_{\pi}\left[\sum_{s=t}^{T} r(X_s, \pi(X_s))|X_t = x;\right]$$

Horizon temporel infini avec critère actualisé

$$V^{\pi}(x) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(X_s, \pi(X_s))|X_0 = x;\right]$$

où $\gamma \in]0,1[$ est un coefficient d'actualisation.

Horizon temporel infini avec critère moyen

$$V^{\pi}(x) = \lim_{T\to\infty} \frac{1}{T} E_{\pi}\left[\sum_{t=0}^{T-1} r(X_s, \pi(X_s))|X_0 = x;\right]$$

Politique optimale

- But : trouver la politique $\pi : \mathcal{S} \to$ A qui a la plus grande *récompense moyenne* :

$$\rho^{\pi} = \lim_{n \to \infty} \frac{1}{n} \mathbb{E}^{\pi} \left[\sum_{t=0}^{n} R_t \right]$$

- Même en connaissant les paramètres, trouver la politique optimale n'est pas évident : c'est le problème dit de *planification*
$\Rightarrow$ Programmation Dynamique

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens
- Définitions
- Exemple: maintenance d'un stock
- Le problème de planning
- Et si on ne connaît pas l'environnement

Une méthode indirecte : l'algorithme KL-UCRL

Le problème

Le responsable d'un entrepot dispose d'un stock X_t d'une marchandise. Il doit satisfaire la demande D_t des clients. Pour cela, il peut, tous les mois, décider de commander une quantité at supplémentaire à son fournisseur.

Il paye un coût de maintenance du stock $h(x)$, un coût de commande du produit $C(a)$

Il reçoit un revenu $f(q)$, où q est la quantité vendue

Si la demande est supérieure au stock actuel, le client va s'approvisionner ailleurs

Le stock restant à la fin procure un revenu $g(x)$

Contrainte: l'entrepot à une capacité limitée M

Objectif: maximiser le profit sur une durée donnée T

Modélisation simplifiée

Modélisation de la demande D_t par une variable aléatoire i.i.d.

Etat: $X_t \in \mathcal{S} = 0, 1, ..., M$: quantité (discrète) de produit en stock

Décisions: $a \in \mathcal{A} = \{0, 1, ...M\}$: commande supplémentaire du produit (Rq: ici l'ensemble des actions disponibles à chaque instant dépend de l'état)

Dynamique: $X_{t+1} = [X_t + A_t - D_t]_+$ (ce qui définit les probabilités de transition $p(y|x, a)$).

Récompense: $R_t = -C(A_t) - h(X_t + A_t) + f([X_t + a_t - x_{t+1}]_+)$

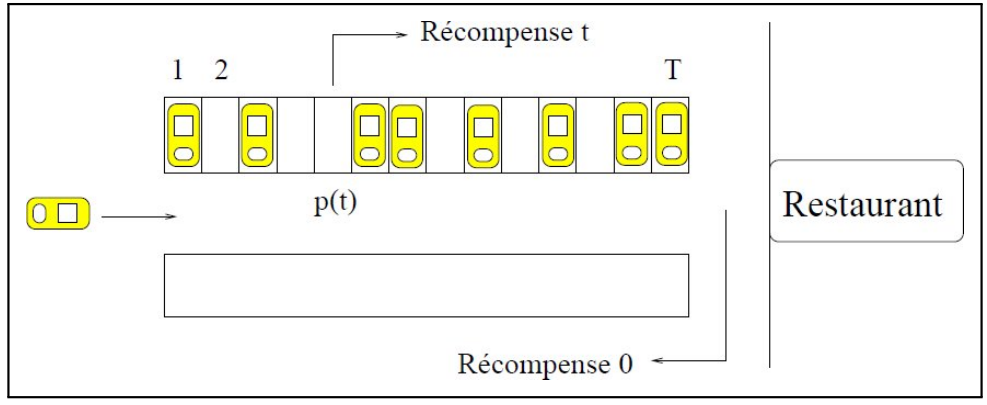
Critère à maximiser:

$$E\left[\sum_{t=1}^{T-1} R_t + g(X_T)\right]$$

Problème du parking

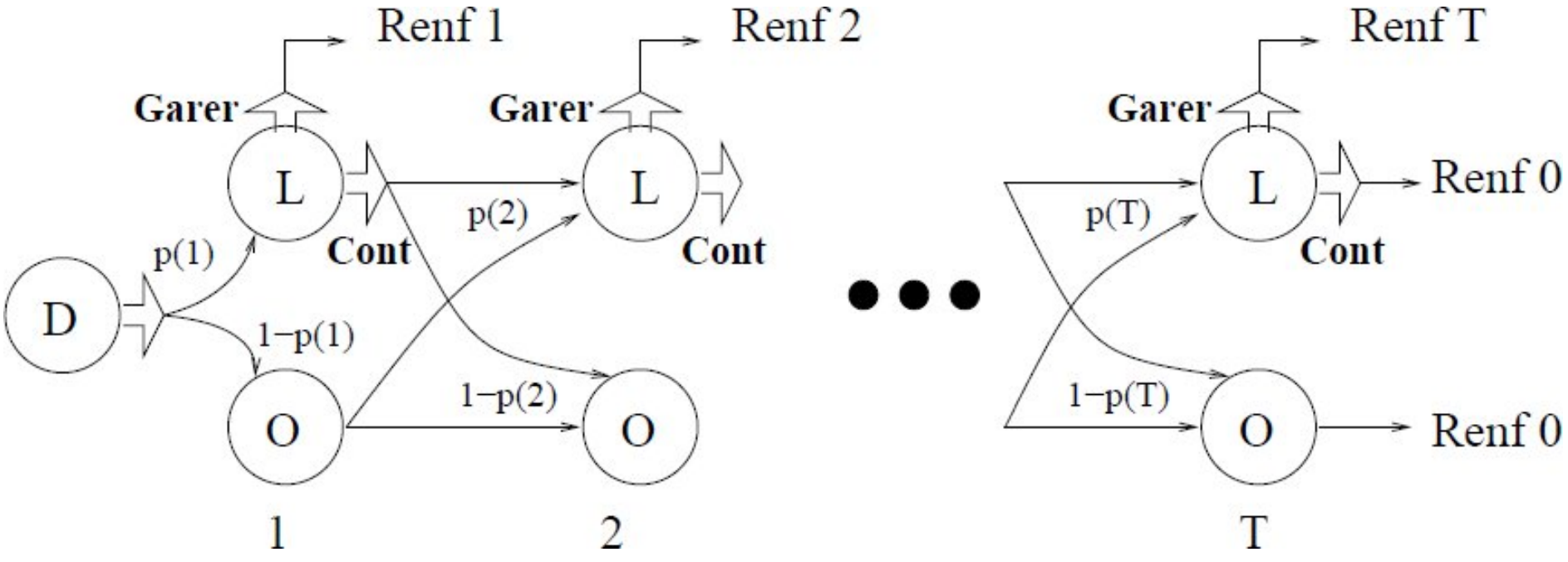
Un conducteur souhaite se garer le plus près possible du restaurant. A chaque instant, l'agent possède 2 actions: continuer ou arrêter.

- Chaque place i est libre avec une probabilité $p(i)$.
- Le conducteur ne peut voir si la place est libre que lorsqu'il est devant. Il décide alors de se garer ou de continuer.
- La place t procure une récompense t. Si pas garé, récompense nulle.



Quelle stratégie maximise le gain espéré ?

MDP pour le problème du parking



Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens
- Définitions
- Exemple: maintenance d'un stock
- Le problème de planning
- Et si on ne connaît pas l'environnement

Une méthode indirecte : l'algorithme KL-UCRL

La fonction valeur

Attribue une valeur à chaque état = ce que l'agent peut espérer de mieux en moyenne s'il est dans cet état.

La valeur $V(x)$ en un état dépend de la récompense immédiate et de la valeur des états résultants $V(y)$

V doit être telle que sa valeur en un état doit être la récompense immédiate plus la valeur moyenne de l'état suivant si je choisis l'action optimale:

Equation de Bellman (critère escompté)

$$V_t(x) = \max_a r(x,a) + \mathbb{E}[V_{t+1}(Y)|x,a]$$

Du local au global

Equation de Bellman

$$V(x) = \max_a r(x,a) + \gamma \mathbb{E}[V(Y)|x,a]$$

Si je connais V , “en moyenne, pas de surprise” !

Comment apprendre la fonction valeur? Grâce à la surprise (TD)

Si V est connue, cela permet de choisir, à chaque instant, la meilleure action

$$\operatorname*{argmax}_a r(x,a) + E[V(Y)|x,a]$$

$\Longrightarrow$ maximiser localement la fonction valeur revient à maximiser le renforcement à long terme.

Equation de Bellman pour le critère moyen

- Pour tout MDP $\mathcal{M} = (\mathcal{S}, \mathsf{A}, P, r)$ *faiblement communicant*, la *récompense moyenne* $\rho^*(\mathcal{M})$ est indépendante de l'état initial.
- Il existe un vecteur de biais h^* tel que, pour tout $s \in \mathcal{S}$,

$$h^*(s) + \rho^*(\mathcal{M}) = \max_{a \in \mathsf{A}} \left(\mathbb{E}[R(s,a)] + \sum_{s' \in \mathcal{S}} P(s'; s, a) h^*(s') \right)$$

Algorithme d'itération sur les valeurs

Pour trouver une politique proche de l'optimale, il suffit de résoudre l'équation de Bellman :

- Soit $k = 0$. Fixons $V_k \in \mathbb{R}^{|\mathcal{S}|}$ et $\varepsilon > 0$
- Tant que $\max_s(V_{k+1}(s) - V_k(s)) - \min_s(V_{k+1}(s) - V_k(s)) > \varepsilon$,

$$\forall s\ ,\ V_{k+1}(s) = \max_{a \in \mathsf{A}} \left(\mathbb{E}[R(s,a)](s,a) + \sum_{s' \in \mathcal{S}} P(s';s,a) V_k(s') \right)$$

- Une politique ε-optimale est donnée par

$$\forall s\ ,\ \pi^*(s) \in \operatorname*{argmax}_{a \in \mathsf{A}} \left(\mathbb{E}[R(s,a)](s,a) + \sum_{s' \in \mathcal{S}} P(s';s,a) V_k(s') \right)$$

Itération sur les politiques

Idée : on part d'une politique quelconque, on l'évalue, et on l'améliore !

- Politique initiale π_0 quelconque
- Evaluation de la politique π_k : on calcule $V_k^\pi(x)$ pour tout $x \in \mathcal{S}$.
- Amélioration de la politique : on choisit la politique *gloutonne*

$$\pi_{k+1}(x) = \operatorname*{argmax}_a r(x,a) + \gamma \sum_y p(y|x,a) V^{\pi_k}(y)$$

- Critère d'arrêt : $V^{\pi_{k+1}} = V^{\pi_k}$ (ou $\pi_{k+1} = \pi_k$)

On parle d'algorithme *actor-critique*

Convergence et complexité

Proposition : L'algorithme d'IP génère une séquence de politiques de performances croissantes qui se termine en un nombre fini d'étapes avec une politique optimale π^*
Complexité :

- résolution directe du système

$$V^\pi = r + \gamma P V^\pi$$

  par la méthode de Gauss : en $O(|S|^3)$ (ou un peu moins)
- itération sur les valeurs : $V^\pi_{k+1} = r + \gamma P V^\pi_k$ en $O(|S|^2 \frac{\log \varepsilon}{\log \gamma})$ pour une valeur ε-approchée
- Monte-Carlo : on tire n trajectoires au hasard, erreur en $1/\sqrt{n}$

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens
- Définitions
- Exemple: maintenance d'un stock
- Le problème de planning
- Et si on ne connaît pas l'environnement

Une méthode indirecte : l'algorithme KL-UCRL

Stratégie

Stratégie : séquence de règles de décision $\Pi = (\pi_0, \pi_1, \pi_2, \dots)$ telle que la politique π_t est choisie uniquement à partir des observations $y_1, \dots, y_{t-1}$.

Si Π est constante dans le temps, on parle de politique stationnaire ou Markovienne: $\Pi = (\pi, \pi, \pi, \dots)$

Pour une politique markovienne fixée Π, le processus $(X_t)_t$ est une chaîne de Markov (définie par les probabilités de transition $p(y|x) = p(y|x, \pi(x))$.

Attention : stratégie $\neq$ politique !

Deux grandes familles de méthodes

Méthodes indirectes apprentissage préalable d'un modèle des dynamiques (forme d'apprentissage supervisé), puis utilisation du modèle pour faire de la planification

Méthodes directes apprentissage direct d'une stratégie d'action sans étape préliminaire de modélisation (peut être intéressant quand les dynamiques d'état sont complexes alors que le contrôleur est simple).

Même si les dynamiques sont connues, le problème de planification peut être très complexe! On cherche alors une solution approchée (programmation dynamique avec approximation), ex: le programme TD-gammon.

Méthode directe : SARSA

SARSA = State-Action-Reward-State-Action
Idée : On construit une table action-valeur qu'on met à jour au fur et à mesure des observations suivant la règle :

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right)$$

$\Longrightarrow$ on ajuste l'estimation de $Q(s_t, a_t)$ suivant la "surprise" qu'on reçoit.

Variante : Q-learning

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(R_{t+1} + \gamma \max_a \{Q(s_{t+1}, a)\} - Q(s_t, a_t)\right)$$

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

L'algorithme UCRL-2 [Auer et al, '09]

- Stratégie optimiste : à l'instant t
  1. considère l'ensemble de tous les MDP (transitions + lois des récompenses) qui rendent les observations assez vraisemblables
  2. trouve le MDP (dit *optimiste*) dont la valeur est la plus grande
  3. joue *pendant un certain temps* la politique optimale de ce MDP
- Le MDP *optimiste* maximise les équations d'optimalité :

$$\forall s,\ h^*(s) + \rho^* = \max_{P,r} \max_{a \in \mathsf{A}} \left(r(s,a) + \sum_{s' \in \mathcal{S}} P(s';s,a) h^*(s') \right)$$

$$\text{tel que } \forall s,\ \forall a,\ \left\| \hat{P}_t(.;s,a) - P(.;s,a) \right\|_1 \leq \delta_P$$

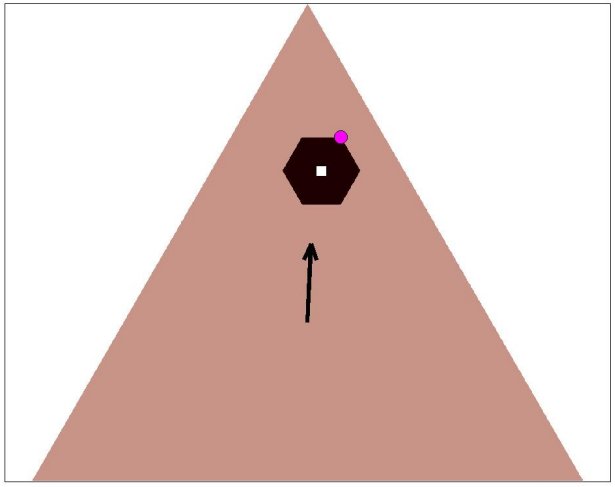
$$\forall s,\ \forall a,\ |\hat{r}_t(s,a) - r(s,a)| \leq \delta_R$$

$\Rightarrow$ *Extended* Value Iteration

Propriétés de l'algorithme UCRL-2

- On doit résoudre à chaque étape des problèmes du type : pour une loi empirique p et pour un vecteur de biais V, trouver

$$q^* = \underset{\|p-q\|_1 \leq \delta}{\text{argmax}}\ q' V$$



Solution :

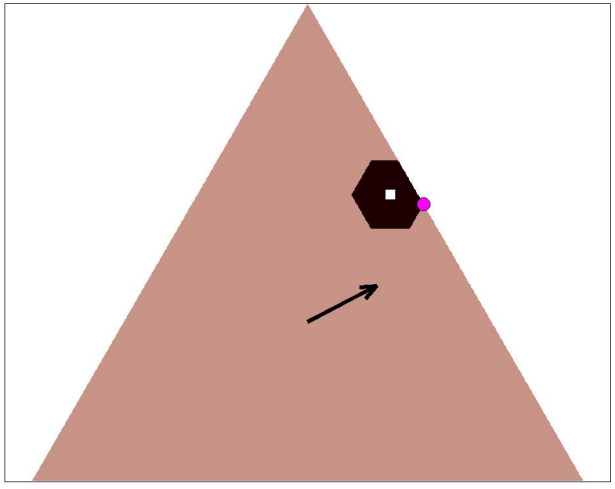
- on gonfle la probabilité de transition vers l'état le plus "prometteur"
- on diminue celle du plus faible, et au besoin du second, etc...

$\Rightarrow$ algorithmiquement trivial, facilement interprétable

Propriétés de l'algorithme UCRL-2

- On doit résoudre à chaque étape des problèmes du type : pour une loi empirique p et pour un vecteur de biais V, trouver

$$q^* = \underset{\|p-q\|_1 \leq \delta}{\text{argmax}}\ q' V$$



Solution :

- on gonfle la probabilité de transition vers l'état le plus "prometteur"
- on diminue celle du plus faible, et au besoin du second, etc...

$\Rightarrow$ algorithmiquement trivial, facilement interprétable

Propriétés de l'algorithme UCRL-2

Mesure de performance : *regret cumulé*

$$\text{Regret}(n) = \sum_{t=1}^{n} \rho^* - R_t$$

De plus, on peut montrer les *bornes de regret* suivantes:

$$\mathbb{E}(\text{Regret}(n)) \leq C|\mathcal{S}|^2|\text{A}|\log(n) \ ,$$

C étant une constante dépendant de

$$\Delta(\mathcal{M}) = \min_{\rho^\pi < \rho^*} \rho^* - \rho^\pi$$

Propriétés du modèle optimiste

Mais le modèle optimiste a quelques propriétés indésirables

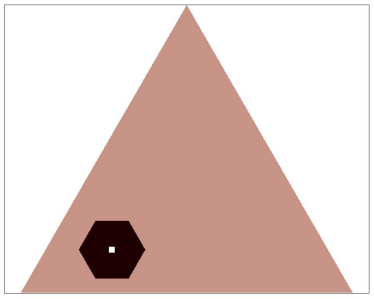
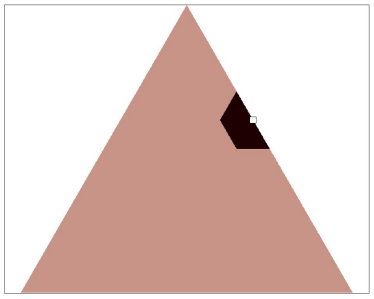
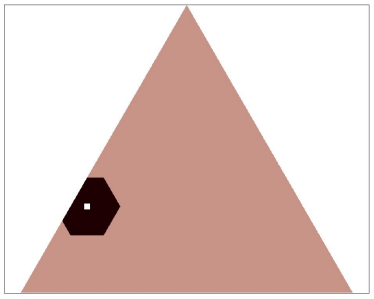
- il ne dépend pas continument des observations
- peut mettre à 0 des transitions observées
- ne peut pas mettre à 0 des transitions vers le ”paradis”
- les voisinages L^1 n’ont pas beaucoup de sens pour des lois de probabilités

$$\Rightarrow \text{comportement difficilement explicable}$$

Les voisinages de UCRL-2

Théorème (Weissman, Ordentlich, Seroussi, Verdu, Weinberger '03) si $X_1, \ldots, X_n$ sont des v.a. iid à valeur dans $\mathcal{S}$ et de loi $p = (p(1), \ldots, p(|\mathcal{S}|))$, l'estimateur $\hat{p}_n = (\hat{p}_n(1), \ldots, \hat{p}_n(|\mathcal{S}|))$ défini par $n\hat{p}_n(i) = \sum_{j=1}^{n} \mathbb{1}_{\{i\}}(X_j)$ vérifie

$$P\left(\|\hat{p}_n - p\|_1 > \delta\right) \leq (2^{|\mathcal{S}|} - 2) \exp\left(-\frac{n\delta^2}{2}\right)$$

Voisinages invariants par translation dans le simplexe.

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

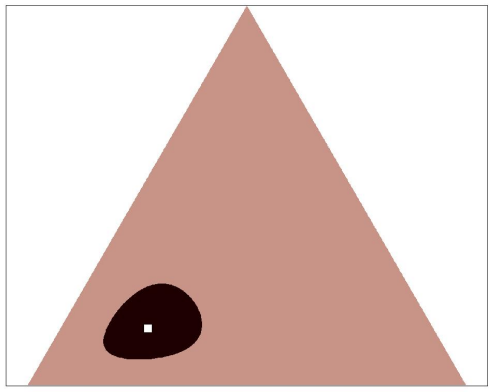
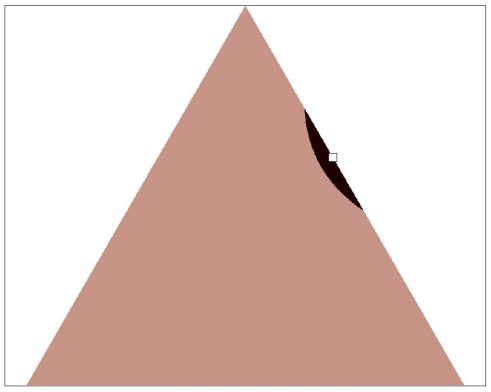
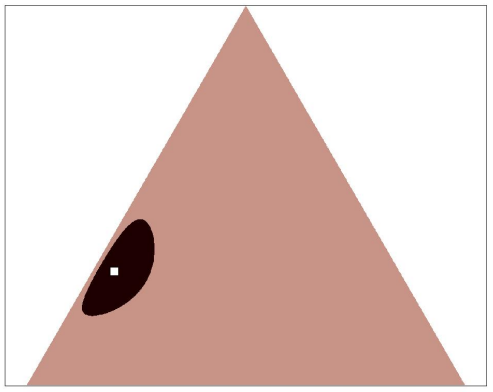
Inégalité de concentration

Théorème: si $X_1, \ldots, X_n$ sont des v.a. iid à valeur dans $\mathcal{S}$ et de loi $p = (p(1), \ldots, p(|\mathcal{S}|))$, l'estimateur $\hat{p}_n = (\hat{p}_n(1), \ldots, \hat{p}_n(|\mathcal{S}|))$ défini par $n\hat{p}_n(i) = \sum_{j=1}^n \mathbb{1}_{\{i\}}(X_j)$ vérifie

$$P\left(\forall t \leq n,\ KL(\hat{p}_t, p) > \frac{\delta}{t}\right) \leq 2e(\delta \log(n) + |\mathcal{S}|)e^{-\delta/|\mathcal{S}|}$$

Contrôle pour tout $1 \leq t \leq n$.
Preuve : à base d'incrément de martingales, cf. Hoeffding-Azuma mais sans majoration uniforme (en particulier, on garde la variance).

Géométrie des voisinages

Le voisinage KL est adapté à la géométrie et aux propriétés probabilistes du simplexe.

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

“Küllback-Leibler UCRL”

- Stratégie optimiste similaire à l’algorithme UCRL-2
- Voisinages du maximum de vraisemblance : utilisation de l’*information de Küllback-Leibler*.
  Le modèle optimiste maximise les équations d’optimalité :

$$\forall s,\ h^*(s) + \rho^* = \max_{P,r} \max_{a \in \mathsf{A}} \left(r(s,a) + \sum_{s' \in \mathcal{S}} P(s';s,a) h^*(s') \right)$$

$$\text{tel que } \forall s,\ \forall a,\ KL(\hat{P}_t(.;s,a); P(.;s,a)) \leq \delta_P$$

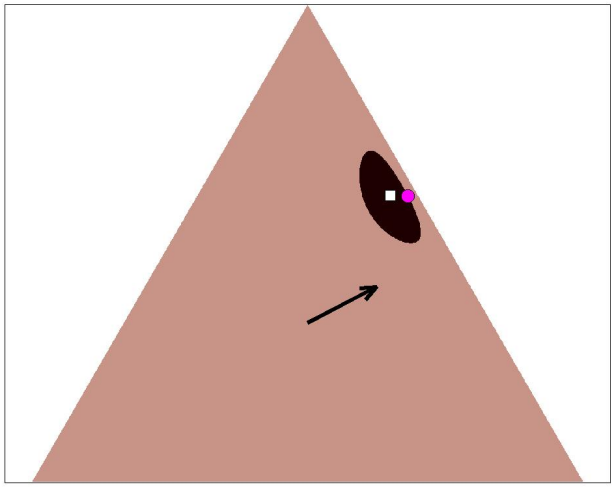
$$\forall s,\ \forall a,\ |\hat{r}_t(s,a) - r(s,a)| \leq \delta_R$$

$\Rightarrow$ *Extended* Value Iteration

La maximisation

- On doit résoudre à chaque étape des problèmes du type : pour une loi empirique p et pour un vecteur de biais V, trouver

$$q^* = \underset{KL(p;q) \leq \delta}{\text{argmax}}\ q' V$$



- Solution explicite de cette maximisation : maximisation d'une fonction linéaire sur un espace convexe.
- Pour $\nu > \max_{i:p_i>0} V_i$ on définit :

$$f(\nu) = \sum_i p_i \log(\nu - V_i) + \log\left(\sum_i \frac{p_i}{\nu - V_i}\right)$$

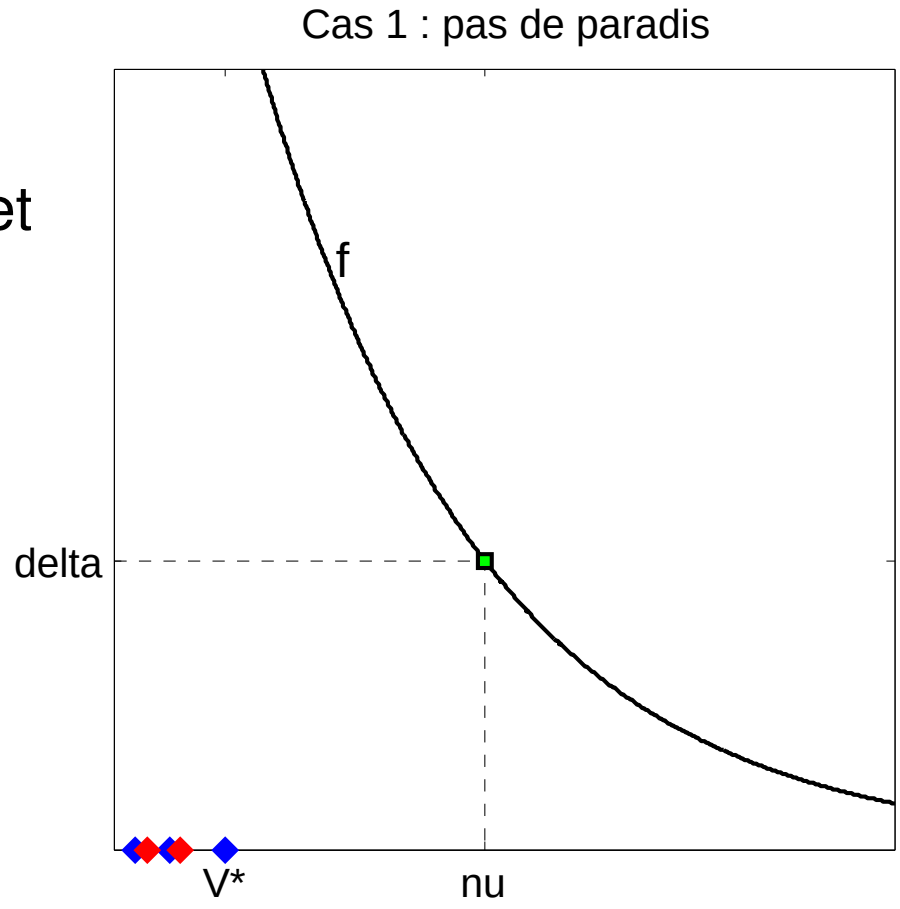
Trouver le maximum

- Soit $i^* = \text{argmax } V_i$. Deux possibilités :
- **Cas 1:** si $p_{i^*} > 0$ alors $f(\nu) = \delta$ et

$$q_i \propto \frac{p_i}{\nu - V_i}$$

- **Cas 2:** Si $p_{i^*} = 0$, 2 cas :
  - **Cas 2.A:** si $f(V_{i^*}) \geq \delta$, alors cf. Cas 1
  - **Cas 2.B:** si $f(V_{i^*}) < \delta$, alors $q_{i^*} > 0$, $\nu = V_{i^*}$ et

$$\text{pour } i \neq i^*, \quad q_i \propto \frac{p_i}{\nu - V_i}$$



Trouver le maximum

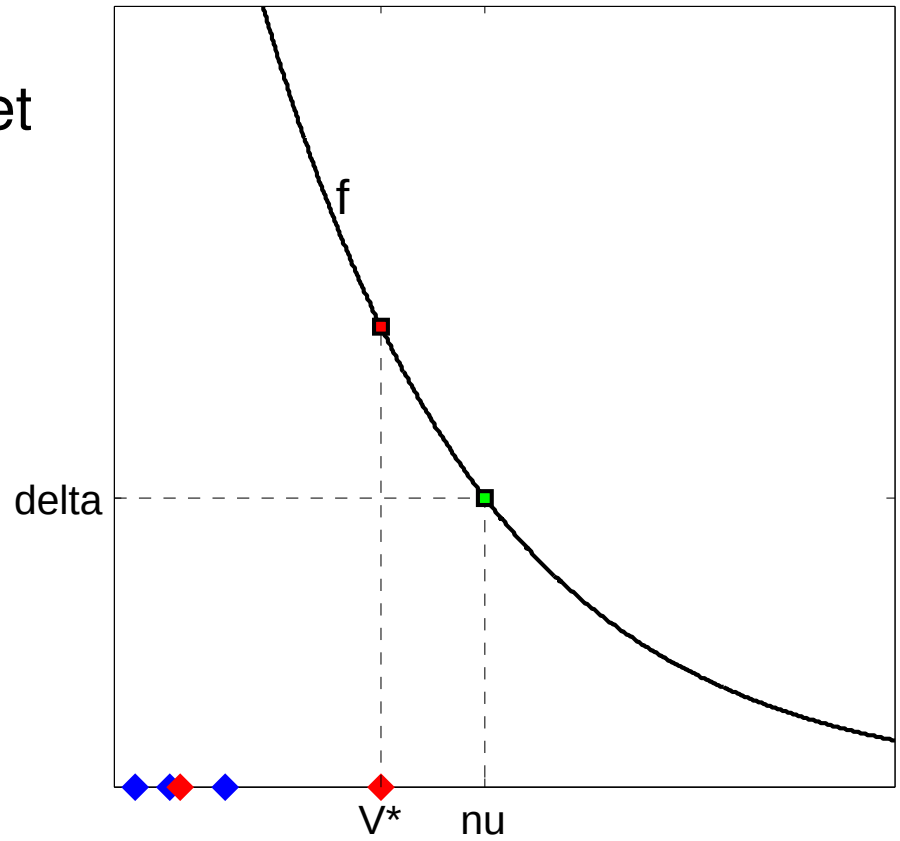
- Soit $i^* = \text{argmax}\, V_i$. Deux possibilités :
- **Cas 1:** si $p_{i^*} > 0$ alors $f(\nu) = \delta$ et

$$q_i \propto \frac{p_i}{\nu - V_i}$$

- **Cas 2:** Si $p_{i^*} = 0$, 2 cas :
  - **Cas 2.A:** si $f(V_{i^*}) \geq \delta$, alors cf. Cas 1
  - **Cas 2.B:** si $f(V_{i^*}) < \delta$, alors $q_{i^*} > 0$, $\nu = V_{i^*}$ et

$$\text{pour } i \neq i^*, \quad q_i \propto \frac{p_i}{\nu - V_i}$$

Cas 2.A : renoncement au paradis



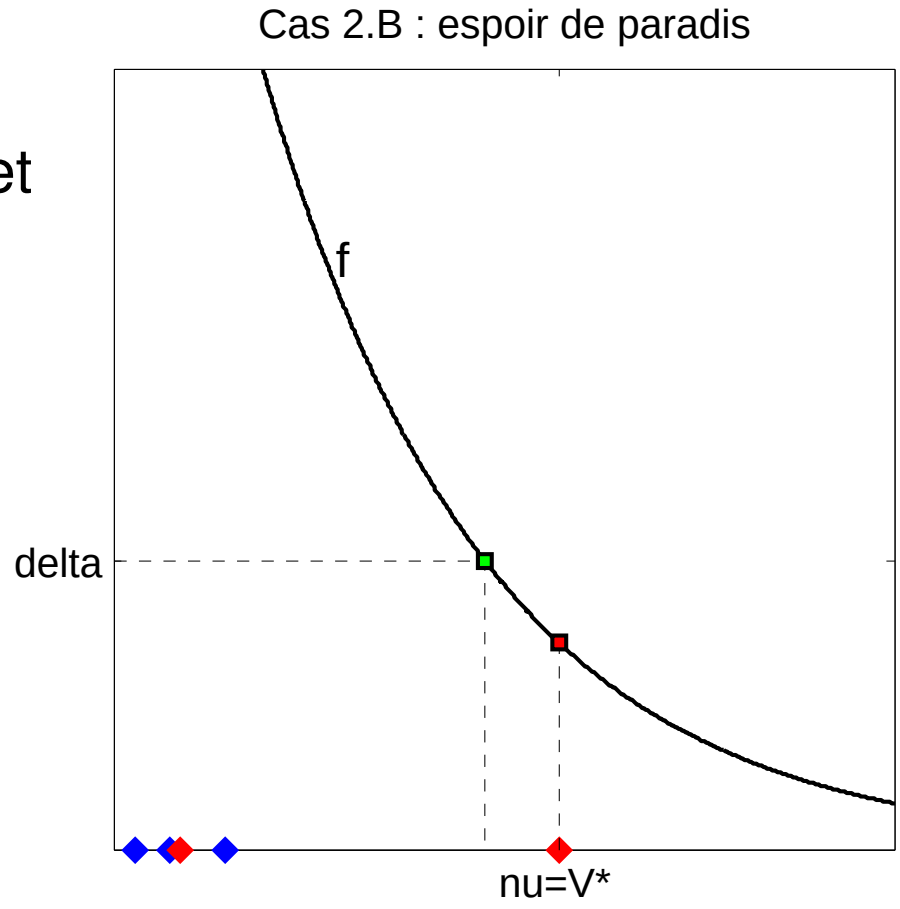
Trouver le maximum

- Soit $i^* = \text{argmax } V_i$. Deux possibilités :
- **Cas 1:** si $p_{i^*} > 0$ alors $f(\nu) = \delta$ et

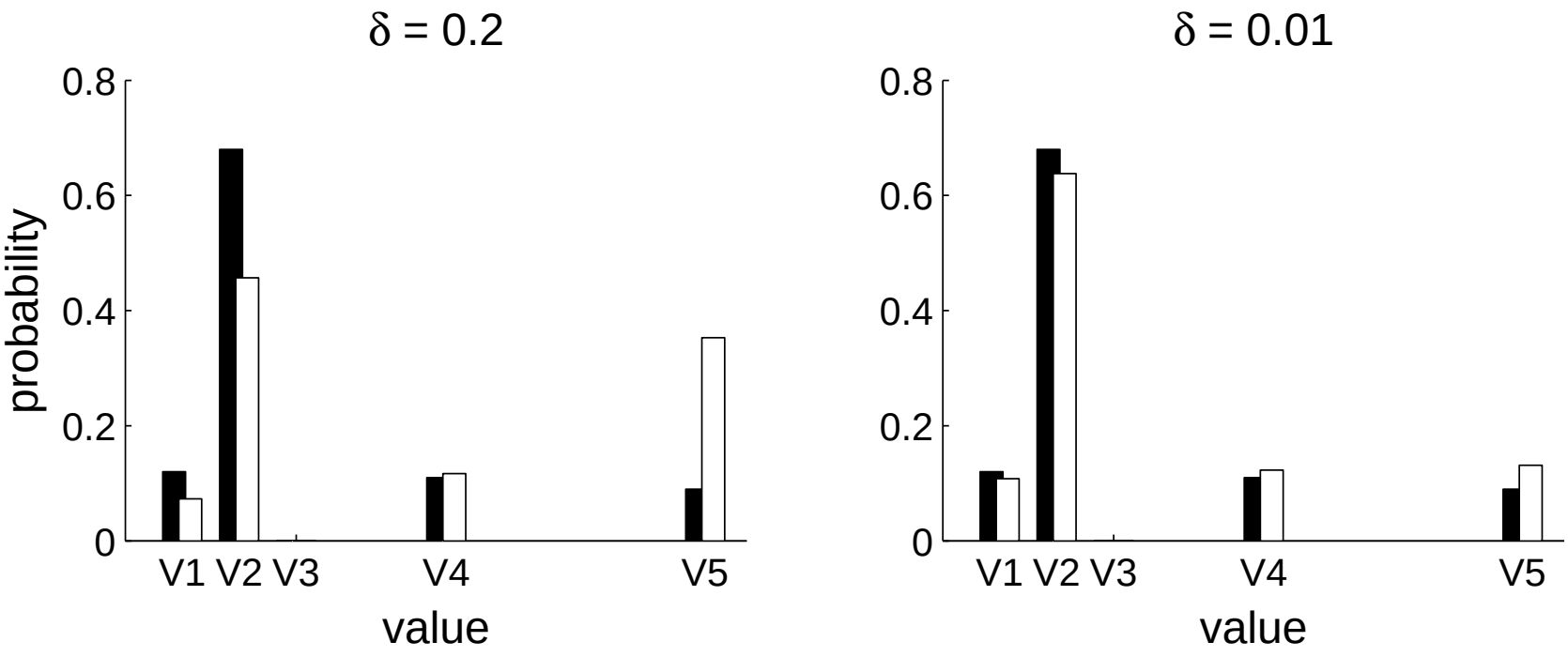
$$q_i \propto \frac{p_i}{\nu - V_i}$$

- **Cas 2:** Si $p_{i^*} = 0$, 2 cas :
  - **Cas 2.A:** si $f(V_{i^*}) \geq \delta$, alors cf. Cas 1
  - **Cas 2.B:** si $f(V_{i^*}) < \delta$, alors $q_{i^*} > 0$, $\nu = V_{i^*}$ et

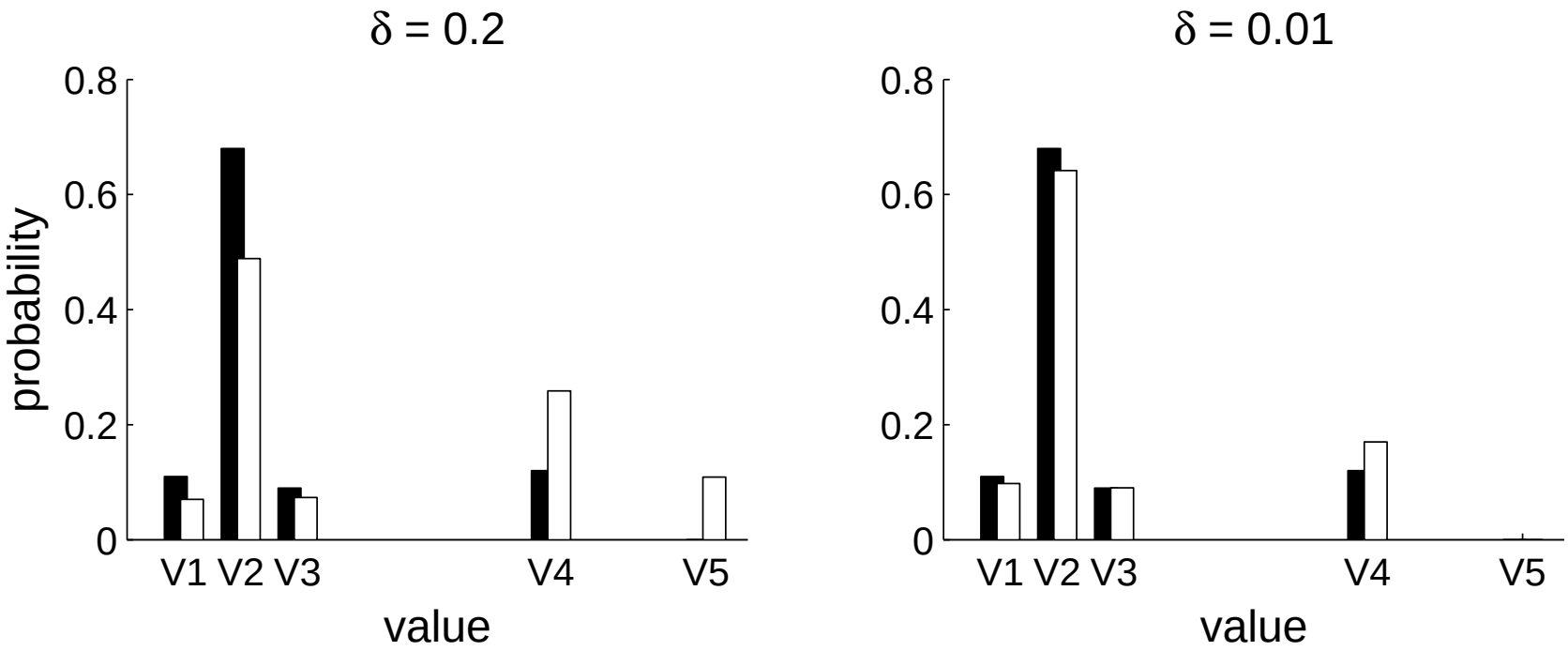
$$\text{pour } i \neq i^*, \quad q_i \propto \frac{p_i}{\nu - V_i}$$



Règle I
“Les meilleurs états sont favorisés”



Règle II
“Pas de Paradis si δ est trop petit”



Commentaires

- La maximisation ne pose donc aucun problème algorithmique et peut être résolue très rapidement en quelques itérations de Newton.
- Si aucune transition vers un ”paradis” n’a été observée, l’algorithme arbitre entre
  - ajouter de la probabilité à cette transition
  - reconnaître qu’elle est invraisemblable et ajouter de la probabilité à d’autres transitions

  en fonction
  - du *nombre de transitions* observées (dont dépend δ)
  - l’*intérêt relatif* de cet état (mesuré par son biais)

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

Majoration du regret

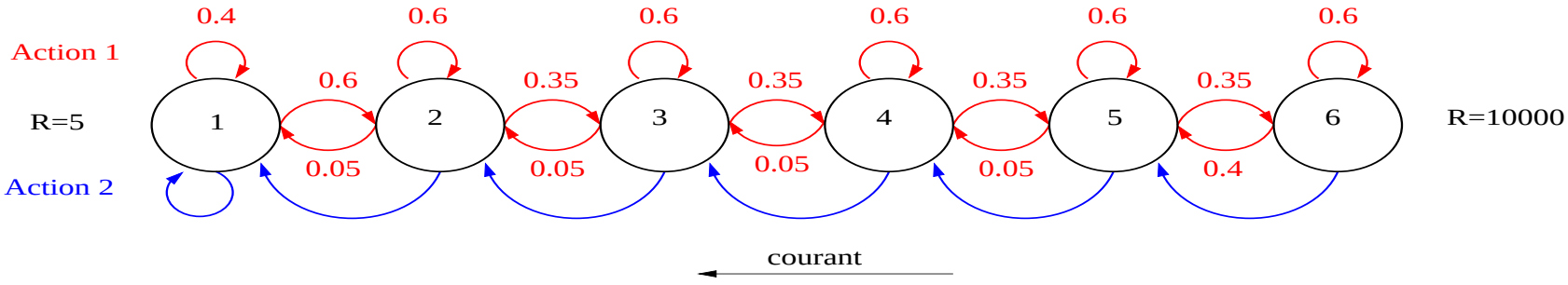
Théorème : Pour un horizon $n > 1$ assez grand, le regret moyen en utilisant l'algorithme KL-UCRL est borné par :

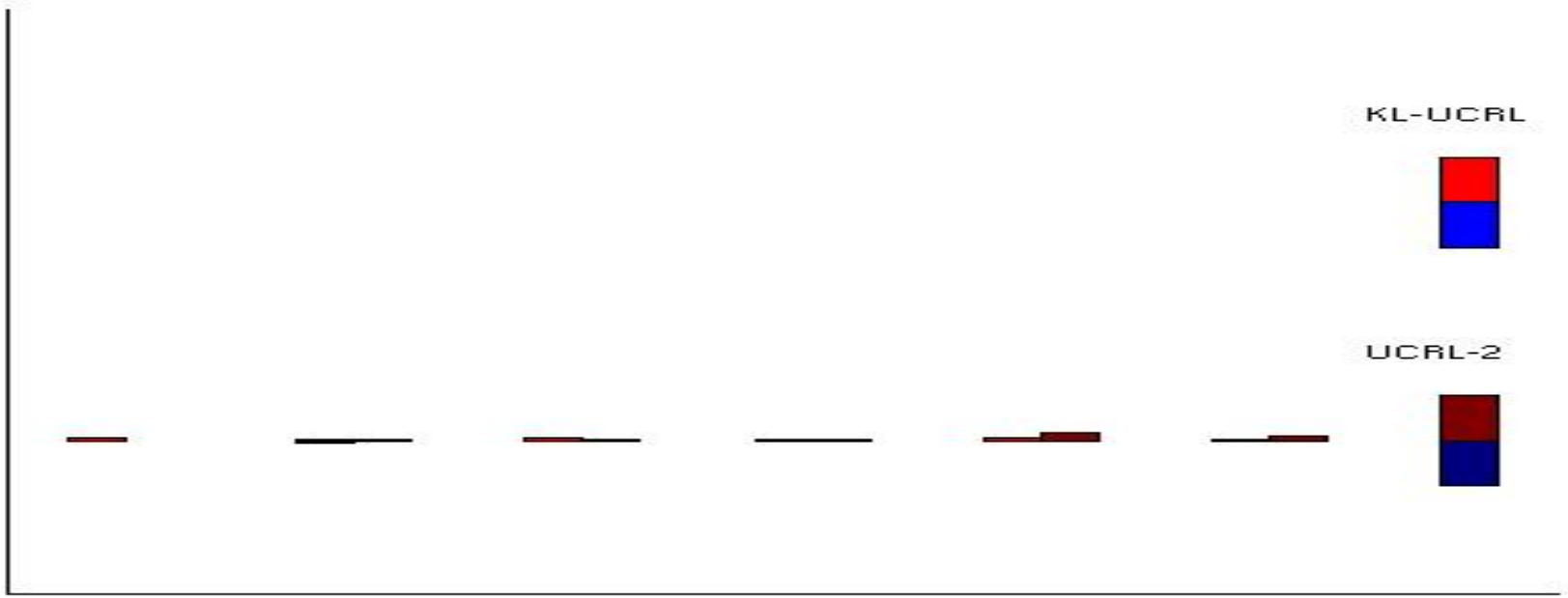
$$\mathbb{E}(\text{Regret}(n)) \leq C|\mathcal{S}|^2|\text{A}|\log(n) \ ,$$

C étant une constante dépendant de

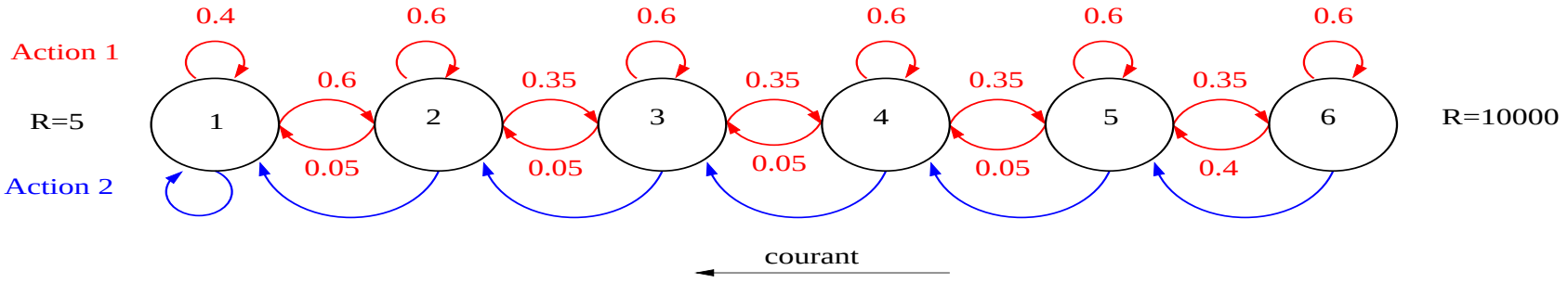
$$\Delta(\mathcal{M}) = \min_{\rho^\pi < \rho^*} \rho^* - \rho^\pi$$

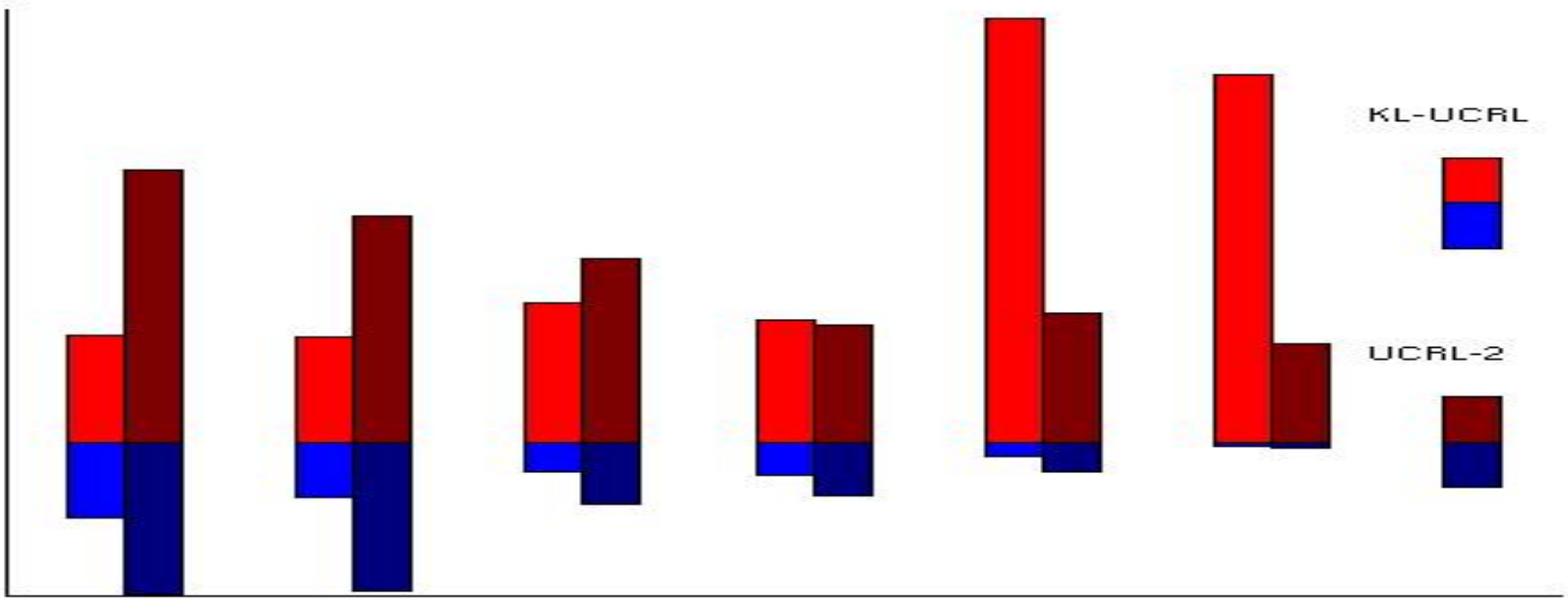
Simulations : RiverSwim





Simulations : RiverSwim





Simulations : RiverSwim

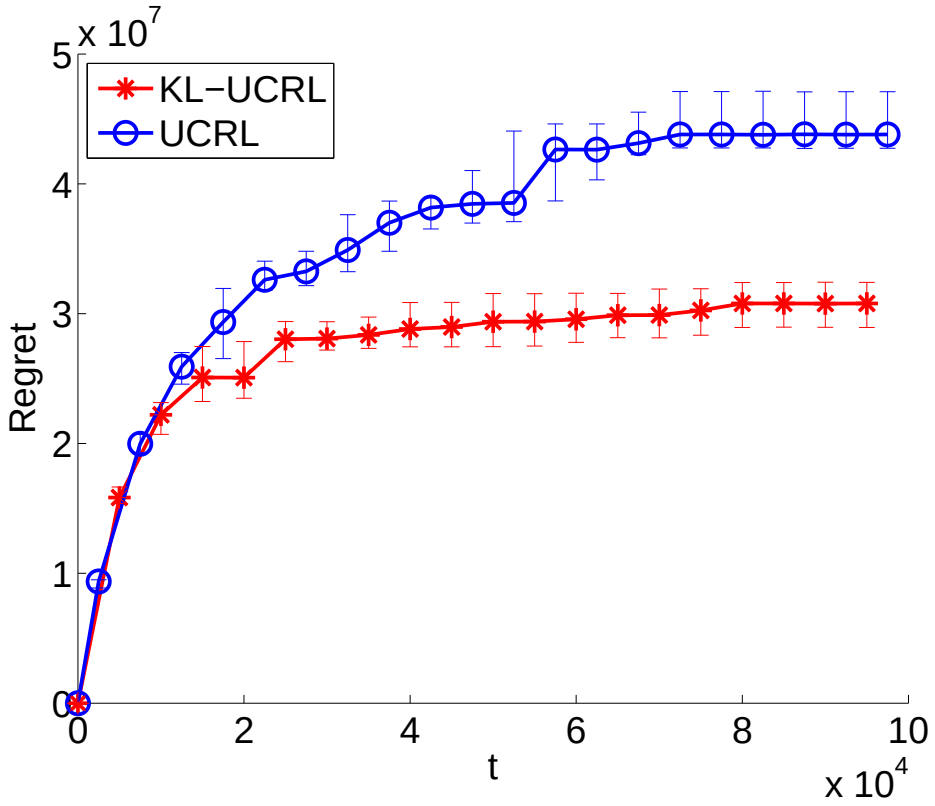


Figure: Comparaison des regrets des algorithmes UCRL-2 et KL-UCRL.

Plan

Cadre général

Problèmes de bandits

MDP : Processus de Décision Markoviens

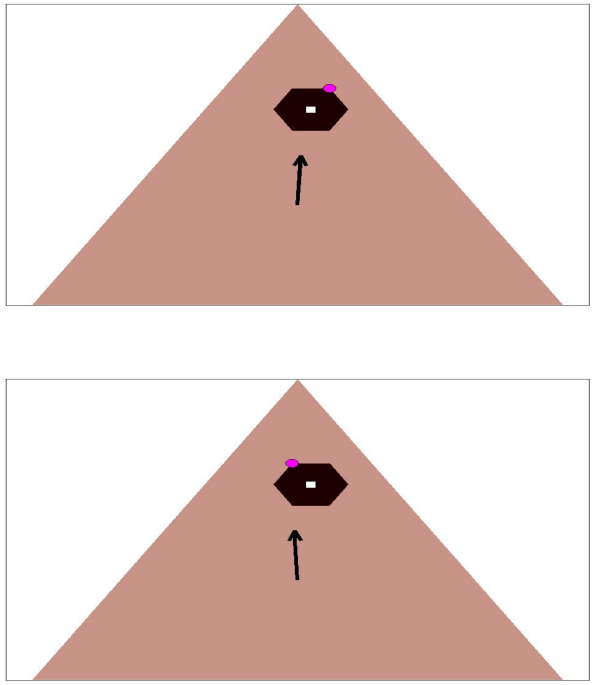
Une méthode indirecte : l'algorithme KL-UCRL

- Méthodes optimistes : l'algorithme UCRL-2
- Amélioration : l'algorithme KL-UCRL
- Estimation des transitions
- Description de l'algorithme
- Regret : bornes et simulations
- Propriétés de KL-UCRL

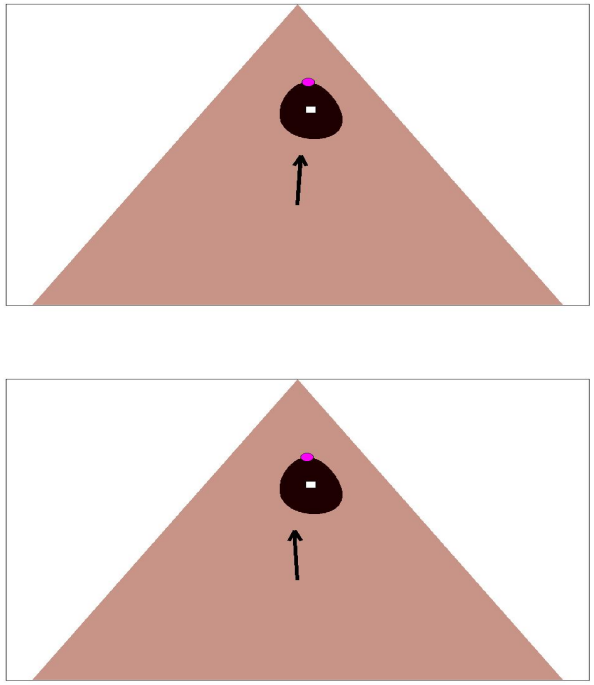
Continuité du modèle optimiste

De plus, le voisinage KL dépend plus continument des observations.

Voisinage L^1



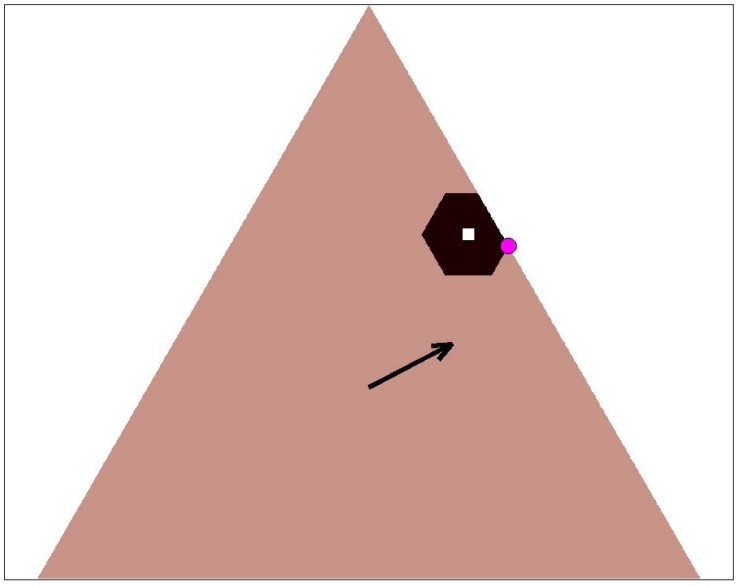
Voisinage KL



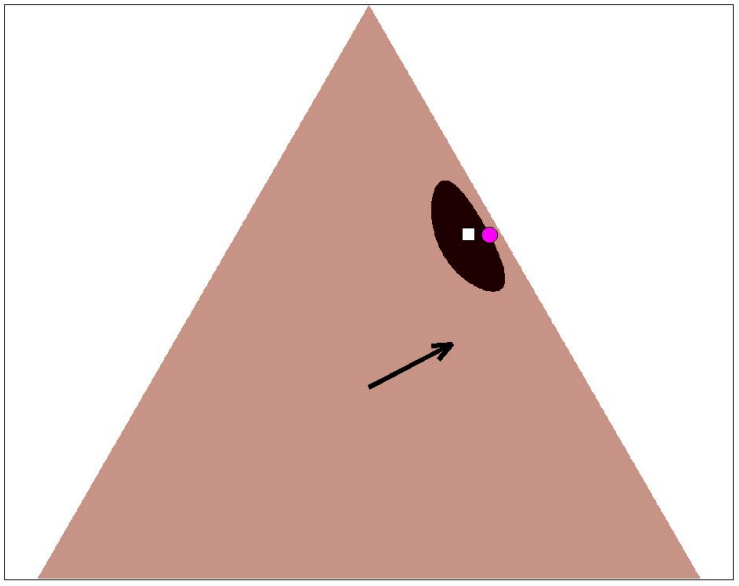
Compatibilité avec les observations

Le modèle optimiste donne toujours une probabilité non-nulle aux évènements observés

Voisinage L^1



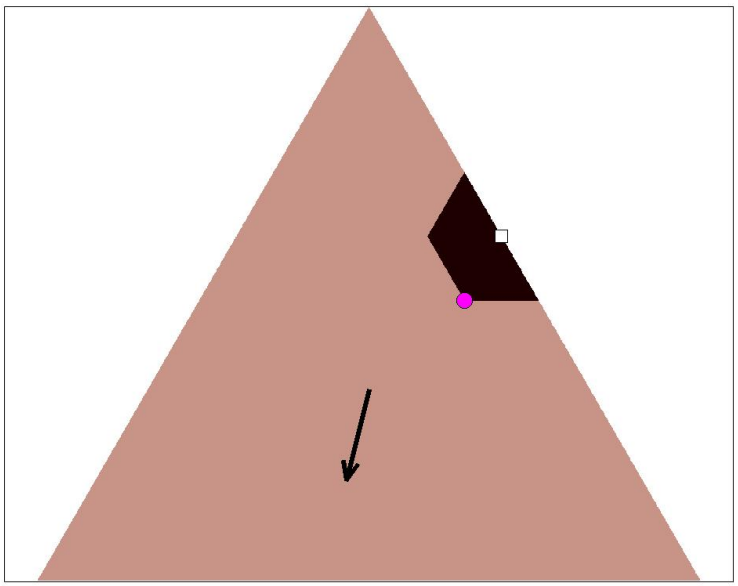
Voisinage KL



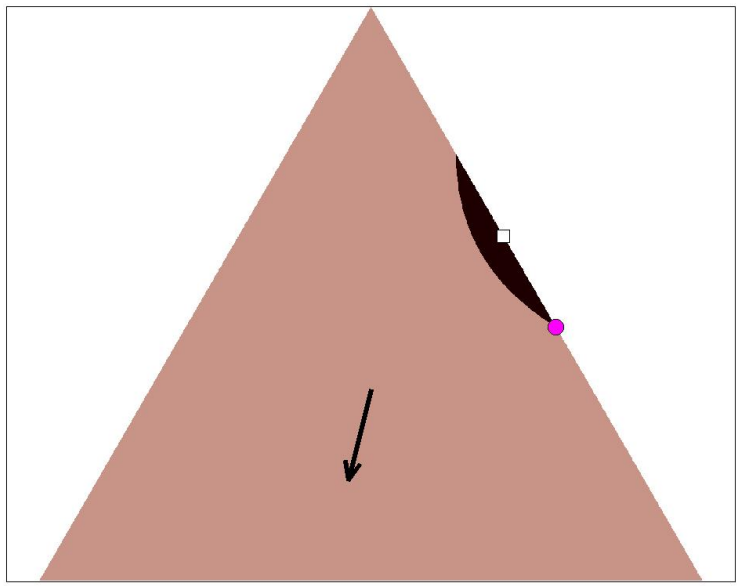
Propriétés des modèles optimistes

Quand une transition de x vers y n'a pas été observée, l'algorithme arbitre entre l'attractivité relative de y et les preuves statistiques accumulées contre l'existence d'une telle transition.

Voisinage L^1



Voisinage KL



Conclusion sur KL-UCRL

- Le calcul du modèle optimiste peut se faire très efficacement avec quelques itérations de Newton
- L'analyse de l'algorithme peut facilement être adaptée aux voisinages KL grâce à l'inégalité de Pinsker
- Il ne nécessite aucune connaissance a priori de la structure du MDP
- Les simulations montrent un comportement significativement meilleur en pratique