Bandits et Plannification

Arpad Rimmel

SUPELEC

15 novembre 2012

Table des matières

Exemple de problème

Problème :

- Vous allez au restaurant tout les 3-4 jours pendants 1 an (100 fois).
- Il y a 10 restaurants dans le quartier.
- Vous voulez déterminer quel est LE meilleur restaurant du quartier.

Exemple de problème

Solutions ? :

- Tester chaque restaurant 10 fois ?
- Tester chaque restaurant 4 fois puis les 3 meilleurs 20 fois ?
- Tester chaque restaurant 1 fois puis les 3 meilleurs 30 fois ?

Intuition

la plupart des algorithmes seront basés sur un compromis entre :

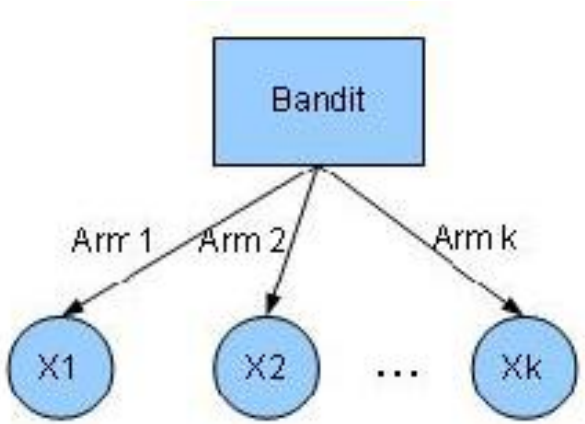
- sélectionner le restaurant qui a obtenu la meilleure note :
- essayer un restaurant peu essayé jusqu'ici :

EXPLOITATION vs EXPLORATION

Autres Exemples de problèmes

- Les bandits manchots : trouver la machine qui rapporte le plus.
- Essais cliniques : trouver le traitement qui fonctionne le mieux.
- Sélection d'un serveur dans un réseau : trouver le serveur avec le temps de réponse le plus faible.
- Publicité ciblée : trouver le type de pub qui intéressera le plus un utilisateur.
- ...

Définition formelle



- un ensemble de bras $A = \{1, ..., K\}$.
- chaque bras est associé à une distribution de probabilité X_k d'espérance μ_k.
- l'algorithme choisit un bras a à chaque pas de temps.
- le bandit retourne une récompense r : une réalisation de X_a.
- les tirages successifs sur un même bras sont indépendant et identiquement distribués.

Notations supplémentaires

- $T_i(n)$: le nombre de fois que le bras i a été sélectionné au pas de temps n.
- $\mu^* = \max_{1 \leq i \leq K} \mu_i$
- $\Delta_i = \mu^* - \mu_i$
- $\Delta = \min_{i:\Delta_i > 0} \Delta_i$

Table des matières

Objectif

Le but est d'optimiser le regret R_n défini comme suit :

$$R_n = \mu^* n - \mathbb{E} \sum_{j=1}^{K} T_j(n) \mu_j$$

$$R_n = \sum_{j=1}^{K} \Delta_j \mathbb{E}[T_j(n)]$$

Borne inférieure

Pour toute stratégie d'allocation et pour tout bras non optimal :

$$\mathbb{E}[T_j(n)] \geq \frac{\log n}{D(p_j||p^*)}$$

$$\text{où } D(p_j||p^*) = \int p_j \log \frac{p_j}{p^*}$$

On en déduit que le meilleur regret atteignable est en $\log(n)$.

[Lai and Robbins, 1985]

Epsilon greedy

Principe de l'algorithme :
A chaque pas de temps t

- avec probabilité $1 - \epsilon_t$, on sélectionne le bras avec la moyenne empirique la plus élevée.
- avec probabilité ϵ_t, on sélectionne un bras au hasard.

ϵ_t décroit avec le temps.

[Auer and all, 2002]

Epsilon greedy

Propriétés théorique de l'algorithme avec $\epsilon_t = \min(\frac{6K}{\Delta^2 t}, 1)$.
Quand $t \geq \frac{6K}{\Delta^2}$, la probabilité de choisir un bras sous optimal est bornée par $\frac{C}{\Delta^2 t}$ où C est une constante strictement positive.
En conséquence, $\mathbb{E}[T_j(n)] \leq \frac{C}{\Delta^2} \log(n)$ et donc

$$R_n \leq \sum_{i:\Delta_i>0} \frac{C\Delta_i}{\Delta^2} \log(n)$$

Epsilon greedy

Défauts de l'algorithme :

- exploration naïve des bras
- nécessite de connaitre Δ

UCB

Principe de l'algorithme :

- A partir des informations disponibles au temps t, on calcule la borne de confiance supérieur (UCB) correspondant à chaque bras.
- On choisit le bras qui a la valeur UCB la plus grande.

[Auer and all, 2002]

UCB

Calcul de la valeur UCB pour le bras i au pas de temps t :

$$\hat{\mu}_{i,t-1} + \sqrt{\frac{3\log(t)}{2T_i(t-1)}}$$

où $\hat{\mu}_{i,t-1}$ correspond à la moyenne empirique du bras i.

UCB

Borne sur le regret :

$$R_n \leq 6 * \sum_{i \neq i^*} \frac{\log(n)}{\Delta_i} + K(\frac{\pi^2}{3} + 1)$$

UCB en pratique

- Ajout d'un paramètre p de contrôle de l'exploration :

$$\hat{\mu}_{i,t-1} + p\sqrt{\frac{\log(t)}{T_i(t-1)}}$$

- Ajout de connaissances a priori $C_i(t)$:

$$\hat{\mu}_{i,t-1} + p\sqrt{\frac{\log(t)}{T_i(t-1)}} + C_i(t)$$

Table des matières

Nombreuses Variantes

- différents environnements : stochastique, adversarial, non-stationnaire, ...
- différents buts : regret cumulé, regret simple, trouver la meilleur valeur, ...
- nombre de bras fini ou infini
- ajout de règles : paiement à chaque essai, paiement pour changer de bras, nombre de bras qui varie, ...
- forme de la distribution connue

nombre de bras infini

Problème quand le nombre de bras est infini ou grand par rapport aux nombres d'essais :

- les algorithmes classiques commencent par essayer une fois chaque bras.

Exemples d'application :

- recherche du meilleur restaurant à Paris
- recherche d'emplacement de ressources précieuses (or, pétrole, ...)

progressive widening

Utiliser uniquement $K(t)$ bras au temps t

$$K(t) = t^{\alpha} \text{ avec } \alpha \in [0.25, 0.5]$$

Les bras peuvent être choisis selon une heuristique ou au hasard.

[Coulom, 2007]
[Chaslot, 2007]

Maximiser la meilleure valeur

Différences avec le bandit classique :

- Seul le meilleur résultat est conservé.
- Il ne faut plus trouver le bras avec la meilleure espérance, la variance doit aussi être prise en compte.
- La notion de regret ne s'applique plus.

Exemples d'applications :

- recherche de meilleur trajet

Threshold Ascent

Principe :

- Calcul d'une valeur pour chaque bras et à chaque pas de temps
- Sélectionner le bras avec la plus grande valeur
- La valeur est basée sur le compromis exploitation/exploration
- l'exploitation est basée sur le nombre de fois que le bras a obtenu une réponse faisant parti des s meilleures

[Streeter and Smith, 2006]

environnement non stationnaire

Différences avec le bandit classique :

- Les variables aléatoires varient au court du temps
- vrai dans beaucoup de cas en pratique
- Nécessité d'imposer des contraintes sur ces variations

Exemples d'applications :

- recherche de restaurant (changement de chef)
- publicité ciblée (les préférences des utilisateurs changent)
- ...

environnement non stationnaire

Cas ou les changements sont "brusques" :

- entre 2 pas de temps, un variable aléatoire peut être modifiée
- la nouvelle variable aléatoire est alors fixe pendant un nombre de pas de temps aléatoire

Solution :

- Discounted UCB : le terme d'exploitation est calculé en mettant un poids plus fort aux derniers résultats.
- Sliding-Window UCB : le terme d'exploitation est calculé sur une fenêtre de temps glissante.

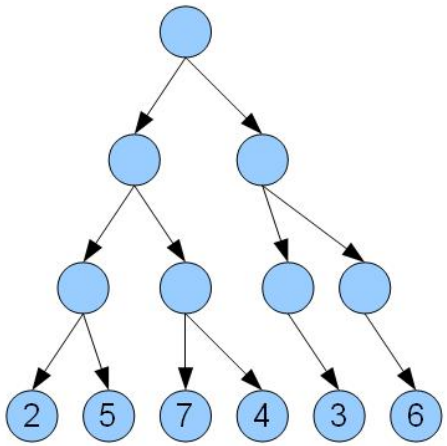
[Garivier and Moulines, 2008]

Table des matières

définition du problème

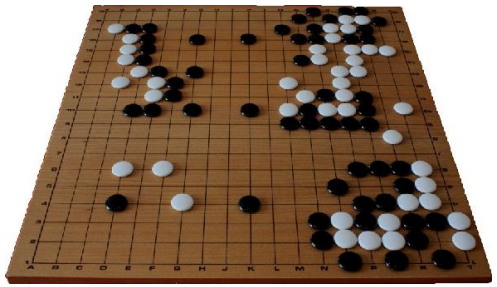
Le but est de prendre une décision dans un environnement :

- observable
- discret
- à horizon fini
- avec un espace d'état trop grand pour être exploré en entier



Exemples d'applications

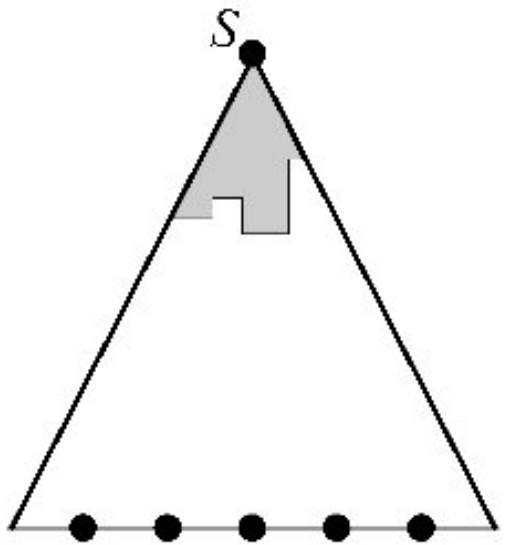
- gestion de production
- apprentissage actif
- multiplication rapide de matrices
- jeu de go



présentation de l'algorithme

Principe :

- construction itérative d'un sous arbre en mémoire
- sous arbre déséquilibré vers les parties intéressantes de l'arbre

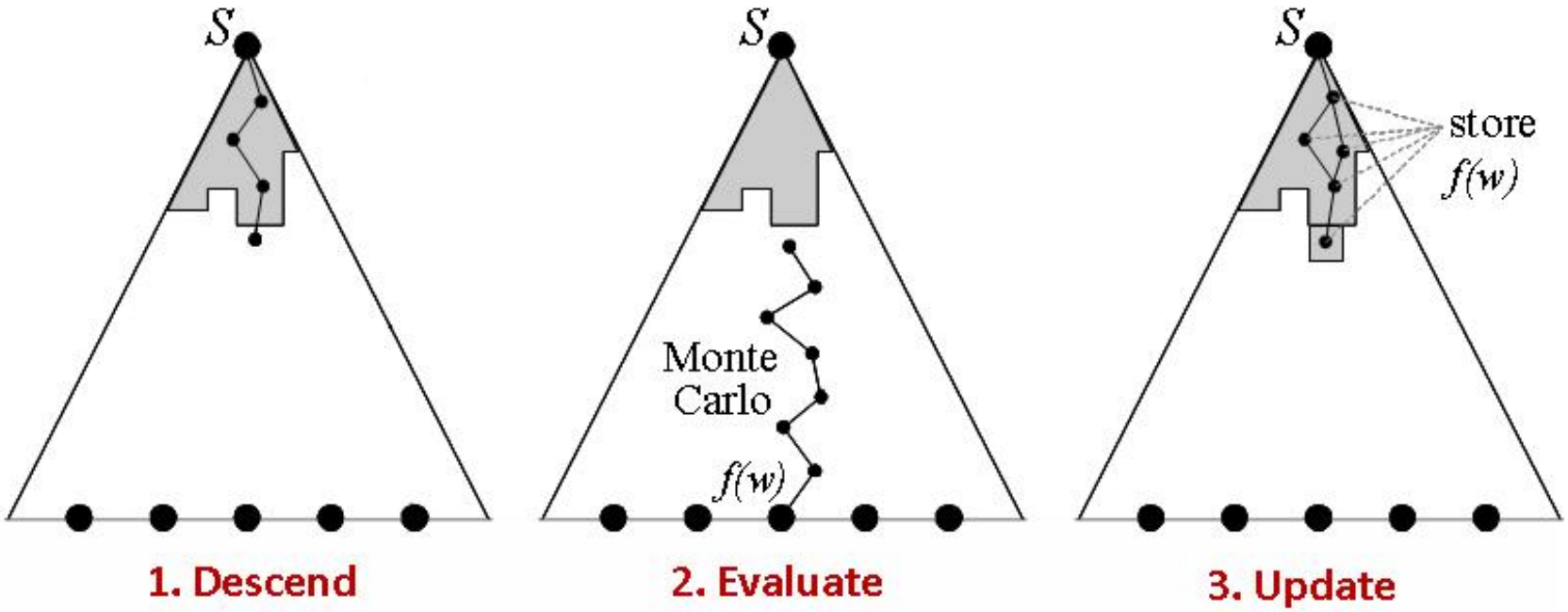


Algorithme

Pour cela, on répète ces 3 étapes jusqu'au critère d'arrêt :

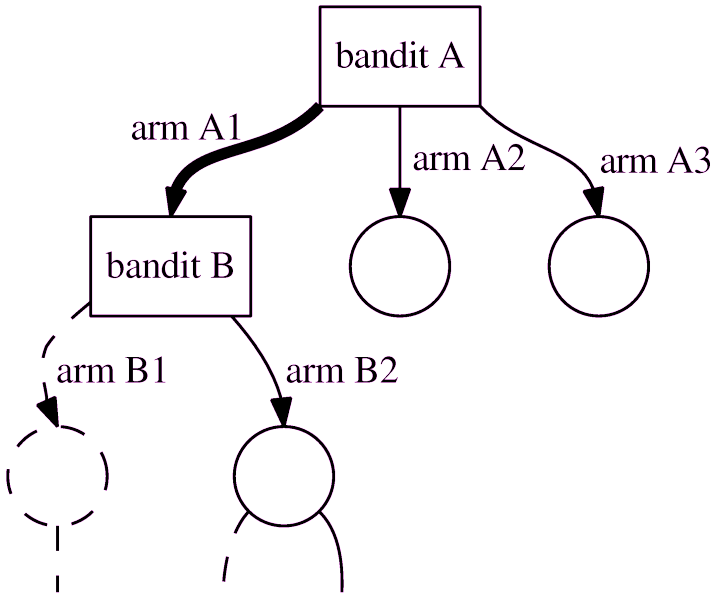
- Descente
  On parcourt le sous arbre en mémoire jusqu'à atteindre un nœud à la frontière de l'arbre.
- Évaluation
  On évalue ce nœud grâce à une fonction d'évaluation non déterministe (Monte Carlo ou autre).
- Mise à jour
  On ajoute le nœud à l'arbre et on met à jour les informations de l'arbre.

Algorithme



Descente dans l'arbre

La descente dans l'arbre se fait en considérant que chaque choix d'une branche est un problème de bandit.



En pratique

Le bandit qu'il faudrait considérer est différent du bandit classique :

- proche des feuilles du sous arbre, le nombre de tentative est faible face au nombre de possibilités
- regret simple
- les variables aléatoires évolues en fonction du temps (pas de manière brusque)
- les tirages ne sont pas iid

Exemples

Exemples d'applications :

- Jeux de plateau à 2 joueurs : Go, Havannah, ...
  UCB modifié + Progressive Widening + connaissances
- Jeux à un joueur : multiplication de matrices
  Threshold Ascent

Remarques

- fonction d'évaluation a beaucoup d'impact
- ordre pour explorer les bras a beaucoup d'impact
- garantie de convergence à l'infini (car arbre exploré en entier)
- pas de garantie sur la vitesse de convergence

Peu se comporter moins bien qu'une recherche uniforme sur certains problèmes

[Coquelin and Munos, 2007]

conclusion

- Le problème de bandit classique a beaucoup été étudié et de nombreux algorithmes ont été proposés qui possèdent de bonnes performances en théorie et en pratique.
- En fonction de l'application, une variante du bandit doit être utilisée.
- Beaucoup de variantes du bandit n'ont pas ou peu été étudiées.
- L'utilisation de bandits pour explorer des arbres donne de bons résultats en pratique mais a peu été étudié de manière théorique.

preuve de UCB

Rappel :

$$R_n = \sum_{j=1}^{K} \Delta_j \mathbb{E}[T_j(n)]$$

Principe :

- Borner $\mathbb{E}[T_j(n)]$ pour $j \neq j^*$

Moyen :

- Borne de Chernoff-Hoeffding

Borne de Chernoff-Hoeffding

- Soit $X_1, ..., X_n$ des variables aléatoires dans $[0, 1]$ tel que $\mathbb{E}[X_t | X_1, ..., X_{t-1}] = \mu$
- Soit $S_n = X_1 + ... + X_n$
- alors pour tout $a \geq 0$

$$\mathbb{P}\{1/n * S_n \geq \mu + a\} \leq e^{-2na^2}$$

- remarque : cette borne permet de relier la moyenne empirique d'une variable aléatoire avec sa moyenne théorique.

preuve UCB

on en déduit pour tout s, tel que $1 < s < t$:

$$\mathbb{P}\{\hat{\mu}_{k,s} + \sqrt{\frac{3\log(t)}{2s}} \leq \mu_k\} \leq e^{-3\log(t)} = t^{-3}$$

$$\mathbb{P}\{\hat{\mu}_{k,s} - \sqrt{\frac{3\log(t)}{2s}} \geq \mu_k\} \leq e^{-3\log(t)} = t^{-3}$$

preuve UCB

On considère le fait qu'un bras $i \neq i^*$ est été tiré au temps t.
On pose $c_i = \sqrt{\frac{3\log(t)}{2T_i(t-1)}}$
on est dans l'un des 3 cas suivants :

- soit la moyenne théorique du bras i n'est pas dans son intervalle de confiance (ev1) :

$$\hat{\mu}_{i,T_i(t-1)} - c_i \geq \mu_i$$

- soit la moyenne théorique du bras i^* n'est pas dans son intervalle de confiance (ev2) :

$$\hat{\mu}_{i^*,T_{i^*}(t-1)} + c_{i^*} \leq \mu^*$$

preuve UCB

- Sinon, on est dans le cas 3.
  comme ev1 est faux :

$$\hat{\mu}_{i,T_i(t-1)} - c_i < \mu_i$$

$$\hat{\mu}_{i,T_i(t-1)} + c_i < \mu_i + 2 * c_i$$

  comme ev2 est faux :

$$\hat{\mu}_{i^*,T_{i^*}(t-1)} + c_{i^*} > \mu^*$$

  comme le bras i a été choisi :

$$\hat{\mu}_{i^*,T_{i^*}(t-1)} + c_{i^*} \leq \hat{\mu}_{i,T_i(t-1)} + c_i$$

  On en déduit :

$$\mu^* \leq \mu_i + 2 * c_i$$

preuve UCB

$$\mu^* \leq \mu_i + 2 * c_i$$

On remplace c_i :

$$\mu^* \leq \mu_i + 2 * \sqrt{\frac{3\log(t)}{2T_i(t-1)}}$$

On en déduit :

$$T_i(t-1) \leq \frac{6\log(t)}{\Delta_i^2}$$

preuve UCB

On pose $u = \frac{6\log(t)}{\Delta_i^2} + 1$

On s'intéresse maintenant au nombre de fois ou le bras i a été tiré.

$$T_i(n) \leq u + \sum_{t=u+1}^{n} (\text{ev1 ou ev2})$$

On en déduit donc :

$$\mathbb{E}\,T_i(n) \leq u + 2 * \sum_{t=u+1}^{n} \sum_{1}^{t} t^{-3}$$

$$\mathbb{E}\,T_i(n) \leq \frac{6\log(t)}{\Delta_i^2} + 1 + \pi^2/3$$

preuve de UCB

Rappel :

$$R_n = \sum_{j=1}^{K} \Delta_j \mathbb{E}[T_j(n)]$$

On en déduit donc

$$R_n \leq 6 * \sum_{i \neq i^*} \frac{log(n)}{\Delta_i} + K(\frac{\pi^2}{3} + 1)$$