SYSTÈME DE RECOMMANDATION : ALGORITHMES ET APPLICATION À LA PLATEFORME KEESEEK

Vidal Agniel[1], Frédéric Bertrand [2], Emmanuelle Claeys [3], Alexandre Delyon[4], Myriam Maumy-Bertrand [5], Titin Agustin Nengsih [6]

[1] *Laboratoire Paul Painlevé, UMR 8524, 59 655 Villeneuve d'Ascq Cedex, vidal.agniel@univ-lille.fr*
[2] *IRMA, UMR 7501, 7 rue René Descartes, 67084 Strasbourg, fbertran@math.unistra.fr*
[3] *IRMA, UMR 7501, 7 rue René Descartes, 67084 Strasbourg, claeys@math.unistra.fr*
[4] *IRMA, UMR 7501, 7 rue René Descartes, 67084 Strasbourg, adelyon@math.unistra.fr*
[5] *IRMA, UMR 7501, 7 rue René Descartes, 67084 Strasbourg, mmaumy@math.unistra.fr*
[6] *IRMA, UMR 7501, 7 rue René Descartes, 67084 Strasbourg, nengsih@math.unistra.fr*

Résumé. Cet article résume le déroulement du projet proposé par la société KeeSeeK dans le cadre de la Semaine d'Étude Maths-Entreprises (SEME) de Strasbourg organisée en novembre 2018. Ce projet porte sur la recommandation d'offres d'emploi dans un moteur de recherche en ligne. Nous nous plaçons dans le cas où le moteur de recherche doit proposer des offres d'emploi susceptibles de maximiser les clics des candidats. Ces offres ont été préalablement sélectionnées et l'utilisateur (possédant le moteur de recherche) souhaite améliorer sa sélection. Pour s'intéresser à ce problème, nous avons proposé une version revisitée d'un algorithme d'apprentissage par renforcement (THOMPSON-SAMPLING) lorsque le nombre d'offres d'emploi disponibles est important (~ 100).

Mots-clés. Allocation dynamique, bandit manchot multi bras, système de recommandations, *partial pooling* et données réelles.

Abstract. This article summarizes the progress of the project proposed by the KeeSeeK company as part of the "SEME" organized in November 2018 at Strasbourg. This project focuses on the recommendation of job offers in a web search engine. We place ourselves in the case where the search engine must propose job offers likely to maximize the clicks of the candidates. These offers have been pre-selected and the user (with the help of the search engine) wants to improve this selection. To tackle this problem, we proposed a revised version of a Thompson-Sampling algorithm when the number of available job offers is high (~ 100).

Keywords. Dynamic allocation, multi-armed bandit, recommender systems, partial pooling, evaluation on real data.

1 Introduction : contexte, données, enjeux

La société KeeSeeK (`https://www.keeseek.com/`) est une plateforme nationale de recherche combinant à la fois une offre d'emploi et une offre de logement. Dédiée aux besoins des entreprises en recherche de candidats à la mobilité professionnelle, la société KeeSeeK propose des offres de logements meublés facilitant la prise du poste et l'intégration dans la région de l'offre du poste. La Figure 1 montre l'interface présentée à un candidat.

FIGURE 1 – Plate-forme KeeSeeK dédiée à la recherche d'un emploi et d'un logement

Les données personnelles stockées par KeeSeeK sont relatives :

- au candidat à une offre d'emploi (identifiant, CV, fichiers optionnels,...)
- au logement disponible (identifiant, situation géographique, superficie,...)
- à l'offre d'emplois (salaire, expérience requise, compétences requises,...)

Lorsqu'un candidat arrive sur la plateforme, il indique via une requête ses critères de sélection comme par exemple le salaire, la localisation et le type de profession. Il peut affiner sa recherche par des informations complémentaires comme par exemple les compétences requises. Cependant, le remplissage de ce formulaire peut être fastidieux et les utilisateurs renseignent généralement le nombre minimal d'informations requises pour obtenir une liste d'offres d'emploi et de logements de la plate-forme. Les approches traditionnelles de recommandations utilisant la factorisation matricielle reposent sur un nombre conséquent de variables, décrivant à la fois le candidat et les objets à lui recommander. Dans notre contexte, la plate-forme obtient beaucoup de résultats pouvant intéresser le candidat. Par exemple si le candidat tape « Strasbourg »+ « Commercial », il obtient alors plus de 150 résultats.

L'objectif de la société KeeSeek est de proposer, pour un couple de mots clés (Emploi/Localisation) fixé, les offres [1] **générant le plus de clics** (le candidat doit cliquer sur l'offre) **dans les premiers résultats de la recherche**. Pour répondre à cet objectif, nous proposons d'utiliser une approche de bandits manchots qui sera décrite dans la section suivante, couplée à une stratégie de *partial pooling*.

1. Par souci de simplicité, nous associons au terme « offres » un résultat combinant une offre d'emploi et une offre de logement

2 Formalisation

Le problème des bandits manchots offre un cadre théorique à la prise de décision itérative dans un environnement incertain et réalise une exploration adaptée à la différence entre plusieurs offres. Les algorithmes pour le problème des bandits manchots offrent de bonnes performances (Lattimore & Szepesvári, 2019) lorsqu'un utilisateur est face au dilemme d'*exploration* (estimer la récompense potentielle de chaque offre) et d'*exploitation* (proposer l'offre la plus performante de manière à maximiser les gains). Introduit par Lai et Robbins (Lai & Robbins, 1985), un agent dispose d'un ensemble $\mathcal{A} = \{a_1, ..., a_K\}$ contenant $K \in \mathbb{N}$ offres (définies comme des *bras*, par analogie aux machines à sous). À chaque itération $t \in \{1, .., T\}$ avec $T \in \mathbb{N}$, l'agent choisit un bras a_t et obtient une récompense $X_{a_t} \in \mathbb{R}$. Dans le problème de bandits dit « stochastiques » (Auer, Cesa-Bianchi, & Fischer, 2002)), les récompenses X_{a_t} sont tirées depuis une distribution stationnaire de moyenne $\mu_{a_t} \in \mathbb{R}$.
La performance d'un algorithme de bandit manchot peut se faire selon la différence entre son espérance de gain obtenu à T et celle obtenue par une autre stratégie (par exemple une stratégie de choix uniforme ou encore une stratégie qui choisirait systématiquement le bras $a^* = \text{argmax}_{a \in \mathcal{A}}(\mu_a)$). Si nous considérons le problème de bandits manchots comme un problème d'estimation, trouver le bras optimal revient à trouver celui qui maximise son espérance totale de gain, de plus rapidement possible. Ce temps nécessaire à l'estimation peut être réduit lorsque nous supposons des *prior* sur la distribution des bras. Lorsque la récompense est une valeur binaire, l'algorithme de THOMPSON-SAMPLING (Thompson, 1933) offre des performances notables (Lattimore & Szepesvári, 2019). Pour réintroduire notre contexte dans la problématique de la société KeeSeek, nous définissons les offres pré-sélectionnées pour une requête comme les bras, et les clics/non clic des candidats comme les récompenses obtenues.

THOMPSON-SAMPLING : l'algorithme THOMPSON-SAMPLING considère les moyennes $\mu_{a_1} \ldots, \mu_{a_K}$ comme des variables aléatoires suivant K lois *a priori*. Dans cas de récompenses binaires (loi de Bernoulli), nous considérons initialement que $\mu_a \sim \mathcal{U}(0, 1)$, c'est-à-dire une loi dont la densité de probabilité est uniforme sur $[0; 1]$. Notons $\Pi_a(t)$ la loi *a priori* des moyennes d'un bras a à l'itération t, $N_a(t)$ le nombre d'essais d'un bras a à l'intération t, et $S_a(t)$ la somme de ses récompenses cumulées à l'itération t. Étant données les récompenses passées (choix d'un bras et récompenses observées), la loi *a priori* $\Pi_a(t)$ est recalculée à chaque itération t :

$$\begin{aligned}\Pi_a(t) &= \mathcal{L}(\mu_{a_t} | X_1, \ldots, X_{N_a(t-1)}) \\ &= \text{Beta}(S_a(t-1) + 1, N_a(t-1) - S_a(t-1) + 1).\end{aligned}$$

L'idée intuitive est qu'après quelques observations, l'intervalle où peut se trouver les moyennes μ_a va évoluer selon les distributions de probabilité. L'Algorithme 1 détaille le fonctionnement de l'algorithme THOMPSON-SAMPLING.

Algorithm 1 ALGORITHME THOMPSON-SAMPLING

Pour chaque $t \in \{1, \ldots, T\}$ **:**
  Tirer K posterior $\theta_a(t) \sim \Pi_a(t)$
  Choisir le bras $a_t = \arg\max_{a \in \{1,\ldots,K\}} \theta_a(t)$
  L'environnement dévoile la récompense X_{a_t}
  Mise à jour de $\theta_{a=a_t}(t)$

L'algorithme THOMPSON-SAMPLING a prouvé son efficacité théorique et pratique pour sélectionner le bras susceptible de maximiser les récompenses après un nombre d'itérations, dépendant de la différence entre le premier et second meilleur bras. Cependant, dans notre contexte, on souhaite sélectionner plusieurs bras parmi un grand nombre de bras possibles ($K \simeq 100$).

3 Application à la plateforme KeeSeeK

Nous souhaitons recommander à chaque candidat cinq offres (bras) possibles en même temps. Nous considérons que les candidats arrivent les uns après les autres. Nous proposons la méthode de *partial pooling* suivante pour utiliser l'algorithme THOMPSON-SAMPLING dans le cas où un candidat peut visualiser cinq offres d'un coup. À chaque itération t, nous tirons sans remise j valeurs comprises entre 1 et K (nous initialisons $j = 5$ au début de l'expérience). Les j valeurs sélectionnées sont enregistrées dans un vecteur $s \in \mathbb{R}^j$. Les paramètres θ_a sont calculés pour $a \in s$. Nous affichons les cinq offres qui maximisent $\theta_{a \in s}$. Après l'observation de des récompenses, on met à jours les distributions de probabilité des cinq offres sélectionnées et incrémentons j d'une valeur choisie arbitrairement (noté *inc* tel que $inc << K$). En augmentant la sélection s de inc valeur à chaque itération, nous sélectionnons des offres qui n'ont encore jamais été proposées, mais également des offres potentiellement intéressantes. À la fin de l'expérience, l'ensemble des θ_a sont calculés. L'algorithme résumant une telle approche est détaillé dans l'algorithme 2.

Nous testons cette approche avec un jeu de données composé de 100 offres soumises à 13404 candidats. Si le candidat a cliqué sur une des offres, la récompense associée est de 1, 0 sinon. Pour faciliter la lecture des résultats, les offres sont classées selon leur performance. En effet, l'offre ayant le moins de clics est l'offre 1, l'offre ayant eu le plus de clics est l'offre 100.
Nous souhaitons savoir si, à la fin de l'expérience, l'algorithme a surtout soumis les offres ayant une performance élevée. Après plusieurs itérations (~ 100) nous présentons dans la table 1 le top cinq des offres les plus sélectionnées par notre algorithme. Nous fournissons également un histogramme des choix réalisés lors d'une expérience (Figure 2). Nous comparons notre approche avec une stratégie randomisée qui choisirait uniformément cinq

Algorithm 2 ALGORITHME PARTIAL POOLING DE THOMPSON-SAMPLING

Requis : K,T

Initialisation : inc= 5

$j = 5$

Pour chaque $t \in \{1, \ldots, T\}$ **:**

- Tirer j valeurs uniformément selon la loi hypergéométrique $\mathcal{H}\left(K, j, \frac{1}{K}\right)$. Ces valeurs composent le vecteur $s \in \mathbb{R}^j$
- Calculer les posteriors $\theta_{a \in s}(t) \sim \Pi_{a \in s}(t)$
- Choisir les 5 bras qui maximisent $a_t = \arg\max_{a \in s} \theta_{a \in s}(t)$
- L'environnement dévoile les récompenses X_{a_t}
- Mettre à jour les $\theta_{a=a_t}(t)$
- $j = j+inc$

offres (sans remises, voir Figure 3) .

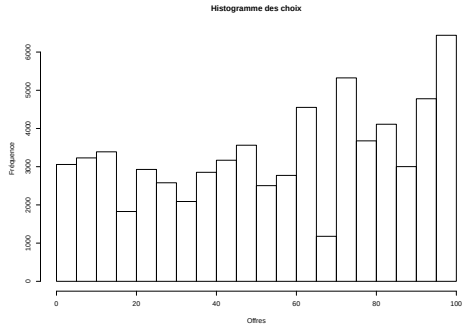

FIGURE 2 – Choix réalisés via l'algorithme *partial pooling* de THOMPSON-SAMPLING.

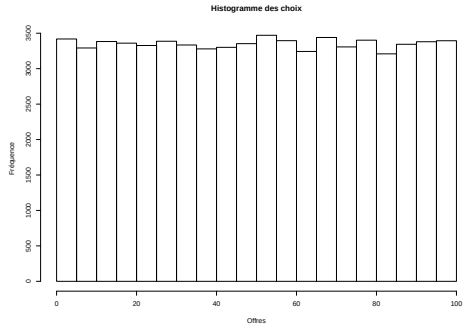

FIGURE 3 – Choix réalisés via une stratégie sans allocation dynamique (tirage uniforme sans remise).

TABLE 1 – Les cinq offres les plus proposées après 100 itérations. Les offres sont numérotées selon leur performance (l'offre 100 maximise le taux de clics)

Rang	Offre
1	74
2	100
3	98
4	84
5	97

4 Conclusion

Dans le contexte de la recommandation sur une plate-forme de recherche, il est parfois difficile d'observer des grandes différences sur la métrique d'intérêt (ici le taux de clics) entre les offres, notamment à cause de la variance engendrée par un nombre important d'offres possibles. L'hypothèse de ne considérer que cinq offres à recommander à un candidat plutôt que le nombre total qu'il est possible de lui afficher fait sens à la fois sur la stratégie d'échantillonnage, mais également sur le côté pratique. En effet, les travaux associés aux moteurs de recherches montrent que l'ordre d'affichage des premières offres impacte fortement la probabilité de clics des suivantes (Bottou et al., 2013). Nous pouvons alors considérer qu'après la cinquième position, une offre n'aurait de toute façon pas intéressé le candidat, quelque soit sa pertinence. Des expériences supplémentaires sont actuellement en cours et permettront de confirmer cette hypothèse. Cette étude de faisabilité pour la société KeeSeeK a été rendue possible grâce à la semaine d'étude maths-entreprise organisée à Strasbourg en novembre 2018 et subventionnée par l'AMIES.

Références

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, *47*(2-3), 235-256.

Bottou, L., Peters, J., nonero Candela, J. Q., Charles, D., Chickering, D., Portugaly, E., ... E.Snelson (2013). Counterfactual reasoning and learning systems : The example of computational advertising. *Journal of Machine Learning Research*, *14*, 3207-3260.

Lai, T. L., & Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, *6*(1), 4–22. Consulté sur `http://www.cs.utexas.edu/~shivaram`

Lattimore, T., & Szepesvári, C. (2019). *Bandit algorithms*. Cambridge University Press (preprint).

Thompson, W. R. (1933). On the Likelihood that one Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*, *25*, 285–294.