AdBandit:
A New Algorithm For Multi-Armed Bandits

Flávio Sales Truzzi[1], **Valdinei Freire da Silva**[2],
Anna Helena Reali Costa[1], **Fabio Gagliardi Cozman**[3]

[1]Laboratório de Técnicas Inteligentes (LTI)
Universidade de São Paulo (USP)
Av. Prof. Luciano Gualberto tv. 3, 158
Caixa Postal 05508– 900 – São Paulo – SP – Brazil

[2]Escola de Artes, Ciências e Humanidades
Universidade de São Paulo (USP)
Av. Arlindo Béttio, 1000, Ermelino Matarazzo
Caixa Postal 03828–000 — São Paulo, SP

[3]Decision Making Lab
Universidade de São Paulo (USP)
Av. Prof. Luciano Gualberto tv. 3, 158
Caixa Postal 05508– 900 – São Paulo – SP – Brazil

{flavio.truzzi,valdinei.freire,fgcozman,anna.reali}@usp.br

Abstract. *In this paper we present a new algorithm for the finite-horizon stochastic multi-armed bandit problem with Bernoulli rewards called AdBandits. It is based upon the Thompson Sampling combined with a frequentist exploitation strategy. We report the results of this new algorithm comparing it with classical solutions to the multi-armed bandit such as: UCB, UCB-Bayes and Thompson Sampling.*

Resumo. *Neste artigo apresentamos um novo algoritmo para o problema de multi-armed bandit com tempo finito e recompensas binárias. Ele foi criado com base no algoritmo Thompson Sampling, combinando-o com uma estratégia frequentista de explotação. Relatamos os resultados deste novo algoritmo comparando-o com algoritmos clássicos como: UCB, UCB-Bayes e Thompson Sampling.*

1. Introduction

In the finite-horizon stochastic multi-armed bandit problem there are n arms that can be pulled. A decision maker needs to choose at each instant of time t which arm to pull, yielding an immediate reward drawn from an underlying, fixed, but unknown distribution associated with the chosen arm. The aim of the decision maker is to maximize the total expected reward over the finite-horizon.

This problem has been studied extensively since its proposal by Robbins [Robbins 1952] and it is motivated by several applications, the most classical being the scheduling of n jobs carried out by a single machine.

We rely in another motivation, the on-line advertising which revenues are skyrocketing [PwC 2012]. In this problem there is an agent called Ad Network which serve ads to users, and the click probabilities of each ad is unknown. The agent needs to discover the underlying probabilities at the same time it is exploiting the ads. This situation exhibit the exploitation-exploration trade-off.

2. Problem Formulation

We consider that the Ad Network has a fixed pool of ads $\mathcal{A}$, known in advance, and each element $a \in \mathcal{A}$ is defined by its underlying click probability, unknown to the Ad Network.

It is considered that the ads are working in the cost per click model without a budget constraint. In this model there are no limits to the number of times one advertise can be used, and the revenue of the Ad Network is measured by the number of conversions, i.e. the amount of times ads are clicked.

At each time t, $0 < t < \tau$, a user generates a request to a site in the Ad Network's inventory, the site makes an ad request to the Ad Network, and finally the Ad Network chooses which ad to display to the user, who may click or not.

We model this problem as a finite-horizon stochastic multi-armed, each arm of the bandit corresponds to an advertise a available in the Ad Network. Each advertise is associated with an unknown Bernoulli distribution $\mathcal{B}(\mu_a)$, where μ_a is the click probability of the advertise a. The aim of the Ad Network is to maximize the total expected reward until time τ, or equivalently, to minimize the expected cumulative regret, defined to be:

$$\mathcal{R}(\tau) = \tau\mu^* - \mathbb{E}\left[\sum_{t=1}^{\tau} r_t\right] = \sum_{a\in\mathcal{A}}(\mu^* - \mu_a)\mathbb{E}[N_{a,\tau}], \tag{1}$$

where $\mu^* = \max_a \mu_a$ denotes the expectation of the best action, r_t is the reward at time t, and $N_{a,\tau}$ is a random variable meaning the number of draws of arm a accumulated until $t = \tau$.

3. Related Work

In the infinite horizon stochastic multi-armed bandit problem the objective is to find a policy that maximizes the infinite horizon expected discounted reward, given by:

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t r_t(\pi(t))\right], \tag{2}$$

where $0 < \beta < 1$ is a fixed discount factor, and $\pi(t)$ is the arm chosen by the policy π at time t.

Gittins [Gittins 1979] proved that the optimal solution to this problem can be achieved by a dynamic allocation index, and it could be solved numerically using the Gittins indices.

According to Kaufmann [Kaufmann et al. 2012a] it is possible to show that the Gittins indices can be extended to the finite horizon problem, however computing the

finite-horizon variant for the Gittins indices is only feasible for moderate horizons due to the need to perform repeatedly dynamic programming recursions.

Lai and Robbins [Lai and Robbins 1985] proved that in this problem the regret grows at least logarithmically, therefore an algorithm is said to solve the multi-armed bandit problem if it can match this lower bound, i.e. $\mathcal{R}(\tau) = O(\log \tau)$. In this section we will introduce some of the usual algorithms to the multi-armed bandit problem that are proved to respect the previously stated regret bound.

3.1. Upper Confidence Bound (UCB)

The Upper Confidence Bound (UCB1) [Auer et al. 2002] is an implementation of the idea of optimism in face of uncertainty proposed by [Lai and Robbins 1985]. This policy achieves logarithmic regret.

UCB1 index-based policy created by the sum of two terms, where the first term is the average reward and the second term is related to the one-sided confidence interval for the average reward according to the Chernoff-Hoeffding bounds [Hoeffding 1963], which provides an upper bound on the probability that the sum of random variables deviates from its expected value. The UCB algorithm can be viewed in Algorithm 1.

Algorithm 1: UCB1 Algorithm

Input: τ (horizon),
$\mathcal{A}$ (arms)

Play each arm a once
Observe rewards r_a
Set $k_a = 1, \forall a \in \mathcal{A}$
Set $\hat{\mu}_a = \frac{r_a}{k_a}$
for $t = |\mathcal{A}|$ *to* τ **do**
  Play arm $\hat{a} = \arg\max_a \left(\hat{\mu}_a + \sqrt{\frac{2\ \ln(t)}{k_a}} \right)$
  Observe Reward r
  $r_{\hat{a}} = r_{\hat{a}} + r$
  $k_{\hat{a}} = k_{\hat{a}} + 1$
  and update $\hat{\mu}_{\hat{a}} = \frac{r_{\hat{a}}}{k_{\hat{a}}}$
end

Initially each arm is played once, and after this initialization, at each time t, the algorithm chooses the arm $\hat{a}$ as follows:

$$\hat{a} = \arg\max_a \left(\hat{\mu}_a + \sqrt{\frac{2\ \ln(t)}{k_a}} \right), \tag{3}$$

where $\hat{\mu}_a$ is the current estimation of the success probability of the arm a, and k_a is the number of times the arm a has been played. It is shown that the expected regret of UCB1 is bounded by:

$$8 \sum_{a:\mu_a<\mu^*} \frac{\ln(t)}{\Delta_a} + \left(1 + \frac{\pi^2}{3}\right) \sum_{a=1}^{\mathcal{A}} \Delta_a, \tag{4}$$

where $\Delta_a = \mu^* - \mu_a$. Despite the algorithm is said to solve the multi-armed bandit problem, in fact it has poor performance compared to the other algorithms shown in this section.

3.2. UCB-Bayes

The UCB-Bayes algorithm [Kaufmann et al. 2012a] assumes a Bayesian modeling of the multi-armed bandit problem, where the parameter $\theta = (\theta_1, \theta_2, \ldots, \theta_{|\mathcal{A}|})$ are drawn from independent prior distributions $\{\pi_a\}_{1 \leq a \leq |\mathcal{A}|}$, and the parameter θ_a corresponds to an estimation of μ_a.

Let $\prod^t$ denote the posterior distribution of θ after t instants of time, with $\prod^0$ denoting the initial prior distribution.

The draw of an arm a generates a reward r_t, and the update of the distributions occurs as follows: $\pi_a^t(\theta_a|r_t) \propto \nu_{\theta_a}(r_t)\pi_a^{t-1}(\theta_a)$, where $\nu_{\theta_a}(r_t)$ is the distribution of the reward r_t, since we model the events as a Bernoulli distribution, the conjugate is a beta function, and the update turns out to be:

$$\begin{aligned} \pi_a^t(\theta_a|r_t) &\propto \left(\theta_a^{r_t}(1-\theta_a)^{1-r_t}\right)\left(\theta_a^{\alpha-1}(1-\theta_a)^{\beta-1}\right) \\ &= \theta_a^{r_t+\alpha-1}(1-\theta_a)^{(1-r_t)+\beta-1} \end{aligned} \tag{5}$$

Let $Q(t, \alpha, \beta)$ be the quantile function associated to the beta distribution with parameters α and β, i.e. the inverse of the cumulative distribution function.

The UCB-Bayes algorithm is shown in Algorithm 2, the authors state that the term $(\log \tau)^c$ in Expression 7 is used to guarantee the finite-time logarithmic regret bounds when $c \geq 5$, and the number of draws of any sub-optimal arm a is upper bounded by:

$$\mathbb{E}[N_{a,\tau}] \leq \frac{1+\epsilon}{\delta(\mu_a, \mu^*)} \log(\mathcal{A}) + o_{\epsilon,c}(\log(n)), \tag{6}$$

where $\delta(\cdot, \cdot)$ is the Kullback-Leiber divergence and $\epsilon > 0$.

Given the definition of regret in Expression 1 it is clear that this algorithm also asymptotically attain to the optimal possible regret stated by Lai and Robbins. In our experiments we used $c = 0$ as suggested by the authors to achieve the best performance.

Algorithm 2: UCB-Bayes Algorithm

Input: τ (horizon),
$\mathcal{A}$ (arms),
$\prod^0$ (initial prior θ),
c (parameters of the quantile)

for $t = 1$ *to* τ **do**
  for *each arm* $a = 1, \ldots, |\mathcal{A}|$ **do**

$$q_a(t) = Q\left(1 - \frac{1}{t(\log \tau)^c}, \alpha_a^{t-1}, \beta_a^{t-1}\right) \tag{7}$$

  end
  draw arm $\hat{a}_t = \arg\max_a q_a(t)$
  Observe reward r_a^t and update $\prod^t$ according to Expression 5
end

3.3. Thompson Sampling

Thompson Sampling is an old algorithm [Thompson 1933] initially proposed to model medical allocation problems and clinical trials. This algorithm has only recently been proved to respect the regret bound stated in Section 2 [Agrawal and Goyal 2012, Kaufmann et al. 2012b]. It uses a Bayesian formulation to the multi-armed bandit problem like the UCB-Bayes algorithm.

Thompson Sampling algorithm is shown in Algorithm 3. The idea is to choose an arm according to its probability of being the best arm. The great difference between the UCB-Bayes algorithm to the Thompson Sampling algorithm is that the UCB-Bayes uses the quantile function as an upper bound to choose which arm to pull, whereas Thompson Sampling samples from the distributions. In the algorithm, S_a corresponds to the number of times the chosen arm yielded a positive reward r and F_a corresponds to the number of times the chosen arm yielded a reward $r = 0$.

The regret shown by Thompson Sampling [Kaufmann et al. 2012b] is upper bounded by:

$$(1+\epsilon) \sum_{a \in \mathcal{A}: \mu_a \neq \mu^*} \frac{\Delta_a(\ln(\tau) + \ln(\ln(\tau)))}{\delta(\mu_a, \mu^*)} + C(\epsilon, \mu_1, \ldots, \mu_{|\mathcal{A}|}), \tag{8}$$

where $\epsilon > 0$, $C(\epsilon, \mu_1, \ldots, \mu_{|\mathcal{A}|})$ is a problem-dependent constant and $\delta(\cdot, \cdot)$ is the Kullback-Leibler divergence.

Algorithm 3: Thompson Sampling

Input: τ (horizon),
$\mathcal{A}$ (arms),
α, β (prior parameters of a beta distribution)

$S_a = 0,\ F_a = 0, \forall a \in \mathcal{A}$
for $t = 1$ *to* τ **do**
  for *each arm* $a = 1, \ldots, |\mathcal{A}|$ **do**
    Draw θ_a according to $Beta(S_a + \alpha_a, F_a + \beta_a)$
  end
  draw arm $\hat{a} = \arg\max_a \theta_a$ and observe reward r
  if $r = 1$ **then**
    $S_{\hat{a}} = S_{\hat{a}} + 1$
  else
    $F_{\hat{a}} = F_{\hat{a}} + 1$
  end
end

4. Ad Bandit

In this section we present our algorithm called Ad Bandit. It is inspired by Thompson Sampling combined with a frequentist exploitation strategy. The algorithm needs to know the priors of the distribution, the horizon τ and the exploration factor ϵ. Prior can be used to tune the algorithm if something is known about the distributions.

The basic idea is to model each one of the arms as a Beta distribution, which is parametrized by two parameters α and β. The α parameter can be understood as the number of times that a particular arm yielded a reward, and the β counts the number of times that the arm did not yielded a reward.

The algorithm has two different stages: the exploration and the exploitation stage, both occuring at random. At the end of each stage, the algorithm checks if the pulled arm yielded a reward or not, and it updates the beta distribution of the pulled arm in a bayesian way.

The exploration stage is based on Thompson Sampling. The algorithm tries to explore the arm with greater probability of being the best arm. In order to do it, it samples from each one of the beta distributions and use the arm which yielded the greater sample, i.e. the arm with the greater probability of being the best arm.

The exploitation stage is based on a frequentist strategy. It uses an expectation of beta distributions as an index to exploit. The rationale for this method is that the learned distributions takes too much time to converge to the real underlying distribution. With that idea in mind the algorithm start to exploit based on the expected mean reward of the distributions $\hat{\mu}_a$:

$$\hat{\mu}_a = \frac{S_a}{S_a + F_a}, \tag{9}$$

where S_a is the number of successful trials and F_a is the number of unsuccessful trials of

the arm a. The algorithm then chooses the arm with the maximum expected mean reward, as follows:

$$\hat{a} = \arg\max_{a}(\hat{\mu}_a) \tag{10}$$

where $\hat{a}$ is the arm chosen to be pulled. The algorithm is shown in Algorithm 4.

Algorithm 4: Adbandit Algorithm

Input: τ (horizon),
$\mathcal{A}$ (arms),
α, β (prior parameters of a beta distribution),
ϵ (exploration factor)

$S_a = 0, F_a = 0, \forall a \in \mathcal{A}$
for $t = 1$ *to* τ **do**
  g = random number between $]0, 1]$
  if $g > \frac{t}{\epsilon\tau}$ **then**
    for *each arm* $a = 1, \ldots, |\mathcal{A}|$ **do**
      Draw θ_a according to $Beta(S_a + \alpha_a, F_a + \beta_a)$
    end
    Draw arm $\hat{a} = \arg\max_a \theta_a$
  else
    Draw arm $\hat{a} = \arg\max_a \mu_a$
  end
  Observe reward r
  if $r = 1$ **then**
    $S_{\hat{a}} = S_{\hat{a}} + 1$
  else
    $F_{\hat{a}} = F_{\hat{a}} + 1$
  end
  $\hat{\mu}_{\hat{a}} = \frac{S_{\hat{a}}}{S_{\hat{a}} + F_{\hat{a}}}$
end

5. Experiments

In this section we present our experiments for the evaluation of the AdBandit algorithm. We created an environment with 10 different Bernoulli arms, their success probabilities can be viewed in Table 1, these are the same probabilities used in the experiments made in [Kaufmann et al. 2012b]. The best possible arm is the number four with its probability in bold.

The algorithms were all written in Python, and for the UCB, UCB-Bayes and Thompson Sampling we used the available implementation in the project pymaBandits 1.0 written by Olivier Cappé et al[1].

[1] Available in: http://mloss.org/software/view/415/ last time accessed in July 17.

For Thompson Sampling and AdBandits we started with no knowledge of the priors, i.e. $\alpha_a = 1$ and $\beta_a = 1$ for all arms a. And we used $c = 0$ for the UCB-Bayes as described by the authors for best performance.

Table 1. Arms probabilities for the experiments.

Arms	
Arm	Success Probability
1^{st}	0.02
2^{nd}	0.02
3^{rd}	0.02
4^{th}	$\mathbf{0.10}$
5^{th}	0.05
6^{th}	0.05
7^{th}	0.05
8^{th}	0.01
9^{th}	0.01
10^{th}	0.01

The performance of the algorithms are compared using the expected mean regret following Expression 1. We created a baseline for the optimal decision making: $r^*(t) = 0.1t$ and compared to the cumulative mean reward of each algorithm over $1,000$ simulations.

5.1. AdBandit Evaluation

First we compare the AdBandit algorithm for different values of the exploration rate parameter ϵ.

Figure 1 shows the behavior of the algorithm for different values ϵ. Inspecting the expected regret at the end of the simulation ($t = \tau = 15,000$), it falls with the decrease of the exploration ratio, until the value of $\epsilon = 0.2$, and starts growing again for $\epsilon = 0.15$, this effect comes from trade-off between exploitation and exploration.

In our experiments we consider that an exploration ratio of 50% is safe against bad probability estimation, but the parameter could be tuned for different environments.

It also can be concluded that exploration ratios under $\epsilon = 0.2$ can be a good option under small horizons or whether the arms probabilities are not very different from each other. We can see this effect in the times between 2×10^3 and 1×10^4 for $\epsilon = 0.15$.

The distribution of the regret can be viewed in the box plot in Figure 2. We can see that the median (indicated by the red line inside each box) is near the regret value of 60 for all values of ϵ. The upper 75% of the values of the simulation (upper line of the box) are under a regret of 100 and the lower 25% (lower line of the box) are all near the value of 50.

We can see that the median is consistently falling with the decrease of parameter ϵ, but the outliers also grow (the outliers are plotted individually with crosses). The lower number of outliers were achieved with an exploration ratio of 40%.

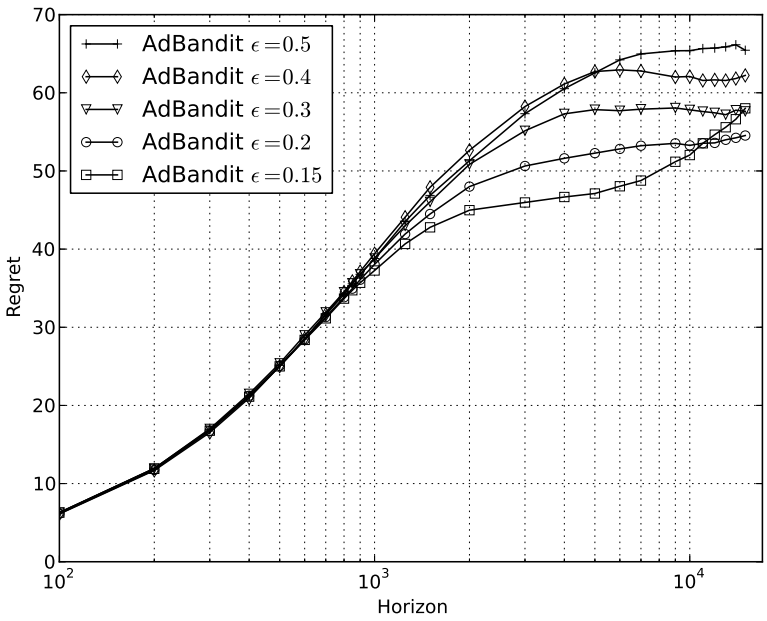

Figure 1. AdBandit mean regret for 15,000 horizon with different values of ϵ over 1,000 simulations.

5.2. Comparison with algorithms in literature

In order to evaluate the AdBandit performance we compared the mean regret of the algorithm with $\epsilon = 0.5$ against the performance of the Thompson Sampling, UCB-Bayes and UCB algorithms, all of them usual algorithms in the multi-armed bandit literature.

In Figure 3 we compare all four algorithms against each other, in this figure we see that the UCB algorithm performs very badly compared to the other algorithms with a mean regret near 700.

In Figure 4 we show only the best three algorithms: AdBandit, Thompson Sampling and UCB-Bayes. Among them the UCB-Bayes has the worst performance with a mean regret near 110, followed by Thompson Sampling with a mean regret near 85, and AdBandit with a mean regret near 65 achieving the best performance.

In Figure 5 we show the distributions of the three best algorithms. AdBandit proved to be the best of them with the lower median, with 75% of its values near the median of the second best algorithm (Thompson Sampling). UCB-Bayes was the algorithm with the smaller number of outliers, but with the higher value of regret of all, with a median near 110, whilst Thompson Sampling had a median near 90.

6. Conclusion

In this paper we proposed a new algorithm for the multi-armed bandit problem called AdBandit, it was crafted upon an existing algorithm called Thompson Sampling, combining a bayesian exploration with a frequentist exploitation. We reported the experimental results of the algorithm by comparing its performance measured by the expected cumulative regret against the algorithms UCB, UCB-Bayes and Thompson Sampling.

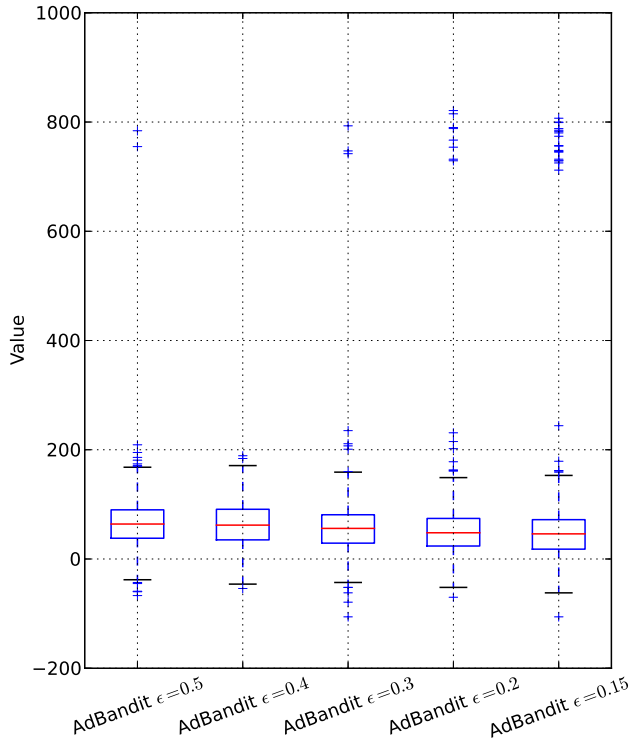

Figure 2. Distribution of AdBandit for 15,000 horizon with different values of ϵ. over 1,000 simulations.

The algorithm AdBandit proved to be the best of the four algorithms for all the values tested of its exploration parameter ϵ. We also compared the distribution of the $1,000$ simulation values using a box plot showing that it achieves the lower median, and the 75% of the values being lower than the median of all other algorithms.

Despite the excellent experimental results, one remaining question is whether this algorithm also presents the optimal regret behavior as described in Section 2. This question is currently under investigation, and we also want to extend the experiments to more diverse scenarios and applying this algorithm directly to our main motivation problem, the Ad Network problem.

Acknowledgments

Flávio Sales Truzzi is supported by CAPES. This research was partly sponsored by grant 11/19280-8 and grant 12/19627-0, São Paulo Research Foundation (FAPESP)[2] and CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico (Procs. 311058/2011-6 and 305395/2010-6).

References

Agrawal, S. and Goyal, N. (2012). Analysis of thompson sampling for the multi-armed bandit problem. In *Conference On Learning Theory (COLT)*.

[2]The opinions, assumptions, and conclusions or recommendations expressed herein are those of the authors and do not necessarily reflect the views of FAPESP.

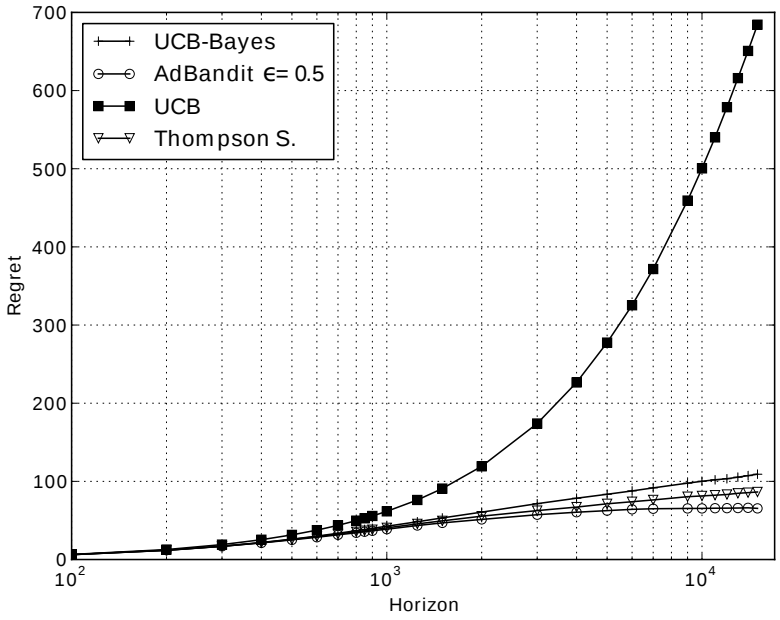

Figure 3. Mean regret for 15,000 horizon for different algorithms over 1,000 simulations.

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

Gittins, J. C. (1979). Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.

Kaufmann, E., Cappé, O., and Garivier, A. (2012a). On bayesian upper confidence bounds for bandit problems. In *International Conference on Artificial Intelligence and Statistics*, pages 592–600.

Kaufmann, E., Korda, N., and Munos, R. (2012b). Thompson sampling: An asymptotically optimal finite-time analysis. In Bshouty, N., Stoltz, G., Vayatis, N., and Zeugmann, T., editors, *Algorithmic Learning Theory*, volume 7568 of *Lecture Notes in Computer Science*, pages 199–213. Springer Berlin Heidelberg.

Lai, T. L. and Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22.

PwC (2012). IAB Internet Advertising Revenue Report 2012. (October).

Robbins, H. (1952). Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

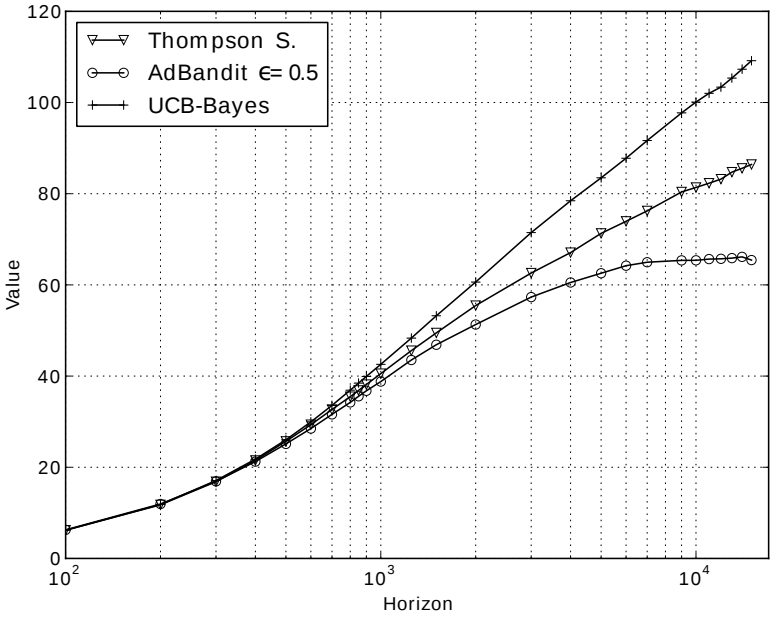

Figure 4. Mean regret for 15,000 horizon for the best three algorithms over 1,000 simulations.

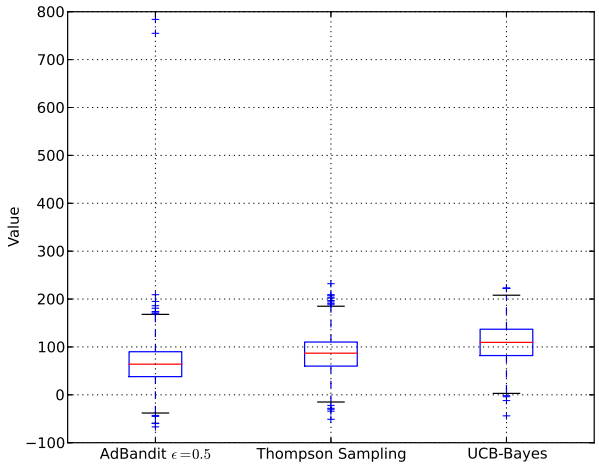

Figure 5. Distribution of the regret for 15,000 horizon for different algorithms over 1,000 simulations.