# Ad campaigns management on the web[*]

Victor Gabillon[†] Jérémie Mary[†] Philippe Preux[†]

**Abstract**

Advertisement is the main source of income for major actors of the Internet. Technically, optimizing the income based on ad campaigns on the web may be seen as a matter of repeatedly solving a time evolving combinatorial optimization problem, under uncertainties. To deal with this category of problems, we propose an iterative algorithm based on the multi-armed bandit framework to estimate the uncertainties, based on which linear programming provides an "optimal" solution. This approach is presented, argued, and demonstrated on a problem based on real data provided by an Internet operator. Noticeably, this ad campaign optimization problem is merely one of a class of problems in which a combinatorial problem is relying on quantities that have to be estimated, and have to be repeatedly solved to keep with the actual observation of stochastic events, in a virtually non ending loop. Hence, the scope of application of the proposed algorithm is much wider that the application case discussed in this paper.

## 1 Introduction

At any given time, the management of ad display on a website may be summarized as follows: on one side, a webpage context which is able to host one, or more, commercial links; on the other side, a pool of commercial links. From the website manager point of view, the goal is to maximize the number of clicks on displayed ads; this maximization takes place in a dynamic context: each ad campaign lasts a certain amount of time (its lifetime), a certain amount of clicks (its budget) is expected during this lifetime, the clicks should be more or less obtained uniformly during this laps of time, and, obviously, actual performance are sought, rather than optimal performance obtained in the limit of infinite periods of time.

In the recent years, this market began to switch from "pay per impression" (CPM) contracts to "pay per click" (CPC) contracts. CPM means that an announcer pays for a certain amount of displays of his/her ads on webpages based on certain characteristics (*e.g.*, the content of the page, or the profile of the visitor); CPC means that the announcer pays for a certain amount of clicks on his/her ads. This is a major change because in the CPM model, the economical risk is assumed by the advertiser: websites rent spaces for displaying ads, and the announcers exploit them at his best (*i.e.* design the ads to target the visitors of the website). But in the CPC model, a website operator has to optimize the display of ads in order to get the maximum amount of clicks on them. Moreover, as there is no risk for the announcer (who only pays the actual clicks), the number of ads is larger than the space availabilities on web pages, so the website operator has to select adequately the ads to be displayed, to fulfill the contracts with the announcers.

The ability to present to the website visitors advertisements they are likely to click on is a critical issue. Even small improvements are crucial because any percent of increase in the click rate is a percent of increase of the income without almost no additional costs. So, not surprisingly many works have been initiated in this field by Yahoo! and Google [9, 11, 4].

In this paper, we only deal with the CPC model. We wish to design effective algorithms to obtain near optimal policies to select the ads to be displayed on webpages. A major originality of this paper stems from the fact that we wish to cope with the real problem. When the problem setting is filled with realistic figures, asymptotically optimal algorithms do not provide an optimal solution, and in the limited time span of the real problem, they may perform very badly.

In the sequel, we first state the problem to be solved in more precise terms in the next section. Then, we review the relevant literature in section 3. Then, in section 4, we delve into the description of our approach which is an iterative process based on a combination of linear programming, and multi-armed bandits. Section 5 provides some experimental results of this approach, before we conclude in section 6.

## 2 The problem and the idea of our solution

In this section, we formalize the problem we face, and provide the idea of the solution we have designed for it.

[*]This research was supported, and partially funded by Orange Labs, under externalized research contract number CRE number 46 146 063 - 8360.

[†]SequeL team, INRIA & Computer Science lab. (LIFL, UMR CNRS), Lille, France. firstname.lastname@inria.fr

**2.1 Problem statement** We consider the following problem. The problem is essentially sequential in time, having virtually no end. At a given time $t$, there is a pool of $K^t$ running ad campaigns $Ad_1, Ad_2, \ldots, Ad_{K^t}$. At time $t$, each ad campaign $Ad_j$ has its remaining click budget $B_j^t$ which is the number of clicks that this ad has to receive during a certain amount of time, its remaining lifetime $L_j^t$. The $B_j^t$ and $L_j^t$ are known (these are specified by the contract between the announcer, and the website manager). Finally, at each time, some ad campaigns reach the end of their lifetime, and new ad campaigns, along with their particular budget and lifetime, may appear with probability $u$. For this purpose, we may assume that there exists an unknown generative model $\mathcal{M}$ that generates ads at each time step.

Each time a visitor requests a certain page (url) on the website, we assume some side information may be available about the visitor (for instance because of the presence of a cookie on the visitor's computer, or because he/she logged in on the website and some information has been kept about him/her); this information may or may not be accurate with regards to the current visitor, that is, the real person being behind the computer, and navigating this website. Based on this information and the requested webpage itself, or more likely, the category of this requested webpage, each visitor is assigned to one among $N$ profiles, denoted $Profile_i, i \in 1, \ldots, N$. To each profile is associated its daily probability of visit on the website $R_1, \ldots, R_N$ ($\sum_1^N R_i = 1$). The requested page is part of the profile so that two requests of the same visitor of two different webpages are associated to two different profiles. We also assume that $\forall i$, $R_i$ is known: indeed, these quantities are easily estimated from weblogs: the number of visits is so large that the confidence on these estimates is very high.

Now, when a visitor requests a certain page of the website, one ad is selected to be displayed on the page. The goal is that during its lifetime, each ad is clicked according to its budget; at the end of a given ad campaign, an inferior amount of clicks decreases the income of the web operator, the decrease being rather sharp. For ease of presentation, we suppose that the pay-off associated to a successful ad campaign (the budget of clicks has been obtained), as well as the cost of not fulfilling it is the same for all campaigns. Moreover, the ads of a given campaign are expected to be clicked at a quite uniform rate along its lifetime.

In this paper, we assume that the problem is stationary, that is the profiles, and the probability of visits belonging to each profile, and the probability that a visit of a profile produces a click on an ad, and the ad campaign generative model are constant in time.

To be precise, we define the following words: a "visitor" is a real person who is currently viewing web pages; a "request" is a request to a web server to open a url along with the available (if any) side information; an "announcer" is an entity who is responsible for an ad campaign; the announcer contracts the "website manager" to buy a certain amount of clicks on some ads.

**2.2 Towards our solution** In order to maximize the expected total number of clicks, a very valuable piece of information is the probability of click (usually called *Click-Through Rate* – CTR) of any profile, on any ad. Let us denote $p_{i,j}$, the probability that a request belonging to $Profile_i$ results in a click on ad $Ad_j$. The probabilities $p_{i,j}$ are unknown. An accurate prediction of these $p_{i,j}$ results in the display of attractive commercial banners to the website visitors. Still, as this learning is performed online, the main issue is to balance the estimation (exploration) of the unknown parameters, with the exploitation of their current estimates. This problem can be formulated as a multi-armed bandit problem (with the ads in the role of the arms).

However, the existence of finite click budgets, along with ad mortality after a finite and rather small lifetime, requires non asymptotic optimal, or to the least good, solution: indeed, we can not satisfy ourselves with an optimal policy that would be reached after a very large numbers of page visits, and clicks, waiting for the law of large numbers to be acceptable. This finiteness of click budgets and lifetimes makes the problem a combinatorial one, along with stochasticity, uncertainty, and an evolution in time. Finiteness creates dependencies between the allocation of the visits to the different ads. The optimal solution is thus no longer to display its favorite ad to each profile that is, the ad which is the most likely to be clicked by this visitor. The pure greedy approach to satisfy each profile would fail to balance the odds of click with the necessary fulfillment of contracts, which is the real goal.

A workable technique to obtain an ad allocation policy is linear programing (LP), based on the current estimators of the probabilities $p_{i,j}$. However, as the LP needs to be specified a finite total number of visits, this method is designed for finite time problems. Though at first sight this finite number of visits seems to lose its meaning in a dynamic environment, we observe that it actually balances policies between greedy behaviors and too far-sighted ones on the set of current ads. Finally, one has to choose the best behavior with respect to the arrival of new ads. Hence in this paper, we endeavor the use of a linear programming approach in the context of a never ending traffic of website visitors and uncertain upcoming arrival of new ads.

## 3 Related work

We review the existing work on the problem of ad selection for display on web pages. The most relevant works may be split into two groups: those based on a bandit approach and those based on a linear programming approach.

**3.1 Multi-armed bandits** The choice of an ad for a visitor is just another formulation for the choice of a slot machine in the multi-armed bandit (MAB) setting. Born with the seminal work of Robbins [12], this problem of the optimal allocation of arm pulls among the bandits witnessed a breakthrough with the UCB algorithm in 2002 [3]. Facing a set of arms, each arm having a fixed, and unknown, probability of success, the goal is to minimize the number of unsuccessful arm pulls; the idea in then to sample the arms and based on the observed successes, tend to select the arm with the highest probability of success as the number of trial increases.

Then, a flurry of variations of the basic bandit problem has been studied in different contexts. "Contextual bandits", also known as "side information bandits", is a development in which extra information is available, hopefully to help choosing the best arm to pull [10, 7, 13, 6]. In the on-line advertisement problem, these information would typically be the information available on the Internet user, the page he/she is visiting, the day, the time of the day, the season, ... Among these works, Pandey *et al.* [10] considered clusters on the website pages and on the ads and built a two-stage bandit method: select the cluster first, then select the ad in this cluster. For Kakade *et al.* [6], the side information is a vector $x \in \mathbb{R}^d$. Inspired by the perceptron, their "Banditron" is an algorithm which goal is to classify those vectors into $K$ labels, *i.e.* the $K$ bandit arms. Here, the ad serving problem is reduced to an on-line multiclass prediction problem with bandit feedback.

Chakrabarti *et al.* [4] analyzed a version of MAB where ads lifetime could be budgeted and revealed to the player, or even be stochastic to model uncertain events which are beyond the control of the ad system. Their results are based on the knowledge of a prior on the arm reward distribution, and they focus on finding rapidly a good arm to stick with among a very large set of ads.

Pandey and Olston [11] proposed an adaptation of the MAB framework to tackle an instance of the problem with click budgets on the ads in the multiple ad display context.

These methods could be useful in our model if they helped us to infer the click probabilities. But, most of the time, they only select the best ads in each context. Yet, to be optimal, the allocation of the different clusters of profile on limited resources has to be scheduled. This planning can be computed with a linear program. Moreover, and most importantly, these works do not consider the limitation on budgets, and the mortality of ads in their planning.

**3.2 Linear Programming** Abe and Nakamura [1] mixed the multi-armed bandit setting with a linear program resolution for on-line advertising with constraints on the impressions proportions. The relevance of the model was demonstrated through simulations. Still, they only considered a context with unlimited resources and a static set of ads.

Mehta *et al.* [9] treated this problem as an on-line bipartite matching problem with daily budget constraints. However, it assumed that we have no knowledge of the sequence of appearance of the profile, whereas in practice we often have a good estimate of it. Mahdian and Nazerzadeh [8] tried then to take advantage of such estimates while still maintaining a reasonable competitive ratio, in case of inaccurate estimates. Extensions to click budget were discussed in the case of extra estimates about the click probabilities. Nevertheless, the daily maximization of the income is not equivalent to a global maximization.

## 4 Hybridizing Bandits and Linear Programming

To sum-up, we wish to optimize a combinatorial problem in order to determine the probability of allocation $s_{i,j}$ of a certain $profile_i$ on a certain $Ad_j$. However, a key ingredient for such a traditional approach is missing which is the probability of click $p_{i,j}$ of any $profile_i$ on any $Ad_j$. These probabilities have to be estimated. This two-fold problem calls for a combination of combinatorial optimization method, along with a learning algorithm. Linear programming and multi-armed bandits provide such tools to be used in sequence: first MAB to estimate the probabilities, and then LP to obtain the optimal solution; however, the word "optimal" may be misleading since we mean here, optimal up to the accuracy of the estimation. However, we are not yet solving the real problem in which:

1. requests observed during a certain time span are not distributed according to the estimated probability of visits $R_i$,

2. real visitors do not click according to estimated

probabilities,

3. new ad campaigns will appear in a near future. The way they appear, their frequency and quality, influences the way we should allocate our ads in the present.

These three issues create discrepancies between the "optimal" solution computed at a certain time, and what may be considered optimal later on, when visitors, and clicks are actually observed (even not mentioning the upcoming ads).

To cope with these observed events, a first solution is to continuously adapt the solution of the problem to new observed conditions. Basically, this is done by merely iterating the process of estimation of probabilities, and solving the combinatorial optimization problem with those current estimates on the current ads.

Before delving into the details of our algorithm, let us define precisely the following:

- $p_{i,j}$ is the probability of click of any $profile_i$ on a displayed $Ad_j$. For the sake of simplicity, we assume that these probabilities are stationary.

- $\hat{p}^t_{i,j}$ is the estimated probability of click of any $profile_i$ on a displayed $Ad_j$, thus $\hat{p}$ is an estimate of $p$. Even though $p$ is stationary, $\hat{p}$ depends on time $t$ because this estimate relies on the series of events that have been observed up to time $t$.

- $x^t_{i,j}$ is the "optimal" probability of allocation of $profile_i$ to a displayed $Ad_j$. This would provide the optimal policy if we were relying on the real $p$, not on this estimate $\hat{p}^t$ to compute $x$. Henceforth, $x$ depends on time as $\hat{p}$ depends on time.

- $s^t_{i,j}$ is the effective probability of allocation of $profile_i$ to a displayed $Ad_j$. If $p$ was known, $x$ would be optimal, and $s$ would be equal to $x$; however, since we rely on $\hat{p}$ to compute $x$, we should not stick to (exploit) the current allocation $x$, but also explore alternate allocations. $s$ is thus a combination of exploitation (rely on $x$) and exploration.

## 4.1 Our hybrid algorithm: MAB+LP

Let us define the linear program based on the notation introduced in the section 2. We note $H$ the horizon, that is, an expected number of ad requests. $x$, the result of the LP is a matrix where $x_{i,j}$ is the number of visits from profile $i$ to allocate on ad $j$. Given $(N, K^t) \in \mathbb{N}^2$, the number of profiles and the number of currently running ad campaigns respectively, $p \in [0,1]^{N \times K^t}$, the click probabilities, $(B^t, L^t) \in \mathbb{N}^{K^t} \times \mathbb{N}^{K^t}$ the budgets, and $R \in [0,1]^N$, the profiles appearance proportions, find $x^t \in \mathbb{R}^{N \times K^t}$, the allocation policy:

So we have to solve a linear problem with $N \times K^t$ variables and $N \times K^t + N + K^t$ constraints. The objective (eq. (4.1)) is to maximize the number of clicks. Inequalities (4.2) express the constraints on the click budget of the ads. Inequalities (4.3) express the constraints on the number of visits in the profiles. Inequalities (4.4) express the constraints on the lifetime of the ads. For the latter inequalities, the idea is to constrain, for each profile, the allocation on an ad to be less than the number of people that the ad will see during its life. But the allocation on ad $A$ has also to take into account the number of persons already taken by the ads which will die before $A$. Note that the $L_j$ may be considered as a number of ad requests, as we will assume the number of visits is constant over days (which is false in the general case, but do not significantly alter the solution).

If the click probabilities $p_{i,j}$ were known, the planning induced by $x$ would be, in expectation, the best non adaptive policy to follow. Still, the CTRs have to be learned on-line. A simple idea, already used in [1], is to use the current estimate of $p$ to compute the planning. For example, after having observed a certain amount of visits, a straightforward estimate for $p_{i,j}$ is $\widehat{p}^t_{i,j} = \frac{NC^t_{i,j}}{NI^t_{i,j}}$ with, for each profile $i$ and ad $j$, $NC^t_{i,j}$ being the number of clicks observed up to time $t$ and $NI^t_{i,j}$ the number of displays performed up to time $t$. This involves that each time a new visit is observed, $\widehat{p}^t_{i,j}$ can be updated because there is a pair $(i,j)$ for which $NI_{i,j}$ has been incremented (and perhaps even the associated $NC_{i,j}$). Then one should immediately take into account this improved estimator and recompute the associated allocation policy. This leads to the algorithm outlined in table Tab. 1.

Yet, in practice, repeatedly solving the LP after every web page request is not reasonable when facing very large amounts of website visitors per day. Hence, to make it effective, we can set a time period T ($T \gg 1$) and carry out this computation every $T$ visits.

Moreover, this is an on-line learning problem where one has to learn the $p_{i,j}$ which are used to compute the LP solution, from which the probability of display of ads to the different visit profiles is deduced (the $s^t_{i,j}$ in the algorithm 1). Then, we have to balance between exploiting the current estimates and exploring the set of possible actions to improve this estimate. This "exploration vs exploitation" trade-off is naturally addressed in the multi-armed bandit setting. Based on the MAB framework, different ways to introduce exploration in the allocation policy are possible, among

Sort the set of ads so that $Ad_1$ dies first and $Ad_{K^t}$ dies last.

$$\textbf{Maximize} \quad \sum_{\substack{1\leq i\leq N\\ 1\leq j\leq K^t}} x_{i,j}\, p_{i,j} \tag{4.1}$$

$$\textbf{Subject to} \quad \sum_{1\leq i\leq N} x_{i,j}\, p_{i,j} \leq B_j^t \qquad \forall j \in \{1,\ldots,K^t\} \tag{4.2}$$

$$\sum_{1\leq j\leq K^t} x_{i,j} \leq R_i * H \qquad \forall i \in \{1,\ldots,N\} \tag{4.3}$$

$$\sum_{1\leq k\leq j} x_{i,k} \leq L_j^t * R_i \qquad \forall i \in \{1,\ldots,N\}, \forall j \in \{1,\ldots,K^t\} \tag{4.4}$$

$$x_{i,j} \geq 0 \qquad \forall i \in \{1,\ldots,N\}, \forall j \in \{1,\ldots,K^t\} \tag{4.5}$$

Table 1: The iterated routine.

.

**Loop at time $t$:**

**Allocation policy:**

Current estimators: $\widehat{p}_{i,j}^t$

Compute the linear program: $x^t = LP(\widehat{p}^t, B^t, L^t, R, H)$

Compute the probability allocations: $s_{i,j}^t = \frac{x_{i,j}^t}{\|x_i^t\|}\ \forall i \in \{1\ldots N\}, \forall j \in \{1\ldots K^t\}$
with $\|x_i^t\| = \sum_{j=1}^{K^t} x_{i,j}^t$

**Visit:** A visit, the $t^{th}$, occurs from $Profile_i$.

**Display:** One ad, $Ad_l$, is chosen, each $Ad_j$ being displayed with probability $s_{i,j}^t$.

**Click:** A click occurs or not.

**Update:** $\widehat{p}_{i,l}^t$ is updated.
$L_k^{t+1} \leftarrow L_k^t - 1 \qquad \forall\, k \in \{1,\ldots,K^t\}$
$B_l^{t+1} \leftarrow B_l^t - 1 \qquad$ if a click has occurred.
A new Ad appears with probability $u$ (update $K^t$ if so).

which we mention the following two:

**LP-$\varepsilon$:** $\varepsilon$-greedy methods can be applied. This means following the LP solution with a (high) probability $\varepsilon$ and with probability $1-\varepsilon$, choose an ad at random.

**Exp-LP:** the $\widehat{p}^t_{i,j}$ can be modified to compel the planning to include exploration. For this purpose, Abe and Nakamura used Gittins' indices [1]. We may also substitute the $\widehat{p}^t_{i,j}$ for the associated UCB indices, or with a value sampled from the posterior Beta distribution over the expected reward (See [5]).

**4.2 The Horizon of allocation** In Sec. 5, we will show the performance of this first method on a 4 days simulation. For this finite time simulation, the horizon parameter $H$ can be precisely set to the remaining number of visits to be treated. Then $H$ becomes $H^t$ and is decreased by one after every ad request. Still, in the real problem, time does not stop after a fixed number of days. The flow of website visitors is infinite and new ads are constantly created, and others are stopped. Moreover, we wish to maximize the total number of clicks, rather than a daily amount of clicks. Therefore, the performance of our policies is dependent on the way new ad campaigns are created, and also on their quality (whether the ad is attractive or not). At this point, an important question is raised: should we be greedy with our current ads and just give to every visitor his/her favorite ad because really soon other interesting ads will enrich our pool of ads? Or should we be more far-sighted and allocate as if a long time will pass before a new ad campaigns appears?

Actually, tuning the horizon parameter, *i.e.* the number of visits we are planning to process, let us balance between a greedy and a far-sighted strategies on the current set of ads. Indeed, if we allocate the next visits as if there were just a little number of visitors yet to come, then the budget constraints would not prevent any visitor from getting its farovite ad, and the algorithm would play greedily. On the contrary, forecasting the arrival of people on a long term basis would render MAB+LP more far-sighted on how to manage the current set of ads. $H$ will now be a fixed parameter with which the MAB+LP algorithm will compute the policy applied to next visit. To illustrate this point, let us consider the following problem displayed in a tabular format that provides the $p_{i,j}$ (Note that they are set here to unrealistic values for the sake of comprehension):

| | Ad 1 | Ad 2 |
|---|---|---|
| Profile 1 | 0.8 | 0.1 |
| Profile 2 | 0.8 | 0.5 |

Now assume both ad campaigns have a budget of 100 clicks, and 2 visit profiles, each with $H$ expected visits. In the two tables below, the allocation computed by the LP is given for two horizons, $H = 20$, and $H = 300$:

The figures inside these 2 arrays are the numbers of visits allocated to each (profile, ad) pair. To obtain the amount of clicks for each pair, these figures have to be multiplied by the $p_{i,j}$. With $H = 20$, the allocation is greedy whereas when $H = 300$, the algorithm tries to take into account the constraints because this big horizon makes it think that a large number of visits will have to share the limited budgets. However, the setting of $H$ strongly depends of the ad campaigns that will come into play in the future. But as two different values of $H$ results in distinct policies, one has to carefully choose the right horizon for its problem.

## 5 Experiments

In this section, we report on experiments designed to assess the validity of our approach, as well as to evaluate its performance. The goals are two-fold: compare the performance of the MAB+LP approach with regards to an optimal solution to the problem, and to other algorithms; then, study the impact of the horizon $H$ on the performance of MAB+LP.

Our approach was tested on a toy-model designed with experts from Orange Labs to fit the real problem; in particular, the orders of magnitude of significant variables are respected. We took care that each ad has its own characteristics that more or less appeal to the different visit profiles.

**5.1 Settings** For the experimental settings, the parameters of the model we work on are set as follows:

- on average, there are 4 million requests per day. In the sequel, ad campaign lifetimes are often expressed in number of clicks instead of days. However, the conversion is easy: $120 \times 10^6$ clicks correspond to 1 month (30 days),
- the values $p_{i,j}$ have a base value of $10^{-4}$,
- in simulations the mean number of living ads were 45; among them, approximately 25% have their budget already exhausted on average,
- there are 54 different profiles of visitors,
- relying on weblogs of the web server, we assumed a uniform distribution of visits along the day.

| $H$ | $R_i$ | Budget | 100 | 100 |
|---|---|---|---|---|
| | | | Ad 1 | Ad 2 |
| **20** $\nearrow$ | 1/2 $\rightarrow$ | Profile 1 | **10** | 0 |
| $\searrow$ | 1/2 $\rightarrow$ | Profile 2 | **10** | 0 |

Planning with $H = 20 \rightarrow$ **Greedy**
Profile 1: 100% on $Ad_1$
Profile 2: 100% on $Ad_1$

| $H$ | $R_i$ | Budget | 100 | 100 |
|---|---|---|---|---|
| | | | Ad 1 | Ad 2 |
| **300** $\nearrow$ | 1/2 $\rightarrow$ | Profile 1 | **125** | **25** |
| $\searrow$ | 1/2 $\rightarrow$ | Profile 2 | 0 | **150** |

Planning with $H = 300 \rightarrow$ **Far-sighted**
Profile 1: 73% on $Ad_1$, 17% on $Ad_2$
Profile 2: 100% on $Ad_2$

Additional technical details are provided in the technical report [2].

**5.2 Playing with a limited number of ads** To get some intuition on the problem, it is easier to start with a first model assuming that ad campaigns are created once for all, and have a boundless lifetime.

Our first point is to assess the improvements in performance brought by the introduction of LP, as well as its limitations. We compare the following algorithms:

**Anytime LP-Best** is LP solver working in the ideal, unrealistic, case in which the $p_{i,j}$ are known. This provide an upper bound on the performance.

**Random policy** provides a lower bound on the performance since it only consists in displaying ads at random, as long as their budget is not null. This is clearly the most obvious solution do select ads, and we are doomed to do better.

**Anytime LP-$\varepsilon$** is our hybrid approach which merges linear programming and $\varepsilon$-greedy exploration (instead of a multi-armed bandit). The $\varepsilon$ parameter was tuned empirically and set to 0.08.

**Bandit Blind-$\varepsilon$** runs bandits concurrently on each profile; that is, this algorithm is only trying to give to each visitor its favorite ads, not taking into account the budget constraints on the other ads. The selection of arms is based on $\varepsilon$-greedy strategy.

Fig. 1 presents some results. The positive impact of using LP can be easily seen: after 7 million simulated requests and until we run out of budget, "Anytime LP-$\varepsilon$" outperforms the "Bandit Blind-$\varepsilon$". Above this 7 M limit, considering the limited budget is hence fruitful. The improvement reaches up to 5% and fills up half the gap between the "Bandit Blind-$\varepsilon$" and the "Anytime LP-Best" policy. Handling the limited budget and the interactions between ads is precisely the role of LP.

The weakness of the LP is that it requires the precise knowledge of the number of visits that will be received on the website, another clearly irrelevant expectation in a realistic context. Thus, in Fig. 1, the two LP curves do not represent the performance of one simulation as a function of time. Actually, each point of abscissa $x$ is the final result of one optimization in which the LP was solved considering that $x$ visits were going to happen, and indeed, $x$ visits happened. To cope with the impossibility of knowing the precise number of visits in advance, "16M LP-$\varepsilon$" displays the performance of one LP-$\varepsilon$ optimized with 16 million visits, and then used to precess the amount of visits really happening. One can see that its performance is linear until 16 M, showing that the algorithm has rapidly good estimates on the important (Profile, Ad) pairs, making it fast to stick to a good allocation strategy. The comparison of this curve to the blind bandits reveals that, in the context of a static set of ads, an over-estimated number of visits could result in really poor results.

**5.3 Playing in a dynamic environment** Now, we wish to evaluate the impact of the horizon (discussed in sec. 4.2) on the performance of our algorithm MAB+LP. So, we turn ourselves to the real problem, where new ad campaigns are created along time, and live during a certain lifetime.

On average, in practice, a new ad campaign is created every 16 hours. Figure 2 exhibits the performance of the different fixed horizons for two lifetimes. An intermediate horizon which has the best balance between being too greedy and too far-sighted performs the best. However, as the lifetime of the ads increases, larger horizons are providing better performance. The "120 M" experiment corresponds to ads with a lifetime of one month. This value seems to be a corner stone between carefully tuning the horizon and just taking the largest one (corresponding to the highest lifetime of all the current ads).

Fig. 2 displays the mean income per unit time from the different strategies. Hence, we are interested by the policy which scores best when a "stationary regime" has settled. In our experiments, the introduction of LP improves the performance whatever is the chosen horizon compared to parallel blind bandit strategy

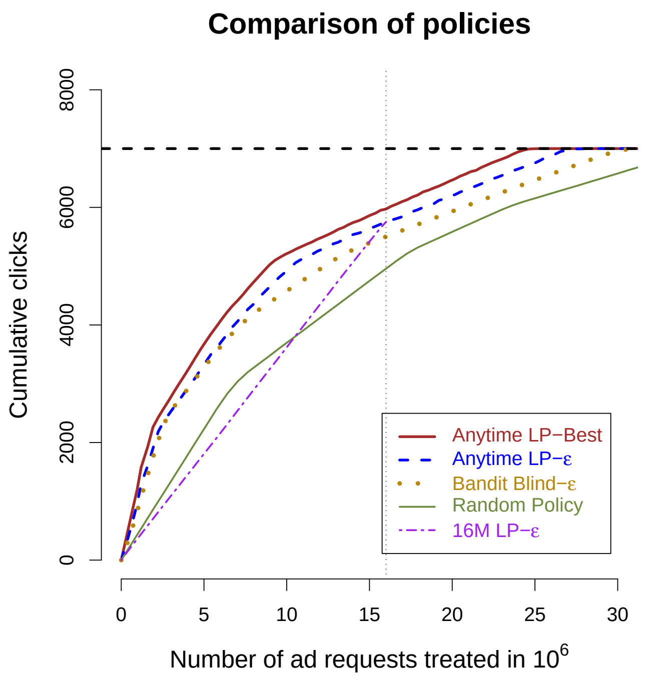

Figure 1: We plot the cumulative number of clicks obtained by 5 different algorithms against the number of observed requests. As explained in Sec. 5.2, these simulations are run without the creation, nor the mortality of ads. The highest curve (strongest, continuous curve, labeled “Anytime LP-Best”) is the optimal strategy, computed assuming that all the click probabilities are known, which is obviously unrealistic. The curve labeled “Random Policy” is a random allocation of the ads. See text for more details.

(when $H$ is small). When finely tuned, the benefit is about 4%. Moreover when comparing with the random policy, the improvement is even more important as in those contexts, it performs around 240 clicks in the stationary regime.

**5.4 Re-planning frequency** In this paragraph, we investigate the effect of relying on estimates of probabilities of clicks to solve the LP, in a stochastic environment. To this end, we compare the use of a constant LP solution, with the solution obtained by solving the LP again and again, after each ad display. As time flows, the discrepancy between the two approaches keeps on increasing, the latter solution performance getting worse and worse. This can be clearly seen on Fig. 3. For trivial computational time requirements, it is not possible to solve the LP after each ad display. So, we have to find an adequate frequency $T$ of solving the LP based on updated estimates $\hat{p}$ and actual requests. $T$ should optimize some trade-off between the computational requirements, and the performance.

As LP-$\varepsilon$ always keeps a part of its play for the exploration, it is less affected by the reduction of the scheduling frequency. To the opposite, Exp-LP strictly follows, between each new planning, a given schedule and performs no exploration. More generally, the two types of exploration methods introduced above are distinguished by their sensitivity to the re-planning frequency: LP-$\varepsilon$ is less sensitive to low frequency re-planning than the Exp-LP.

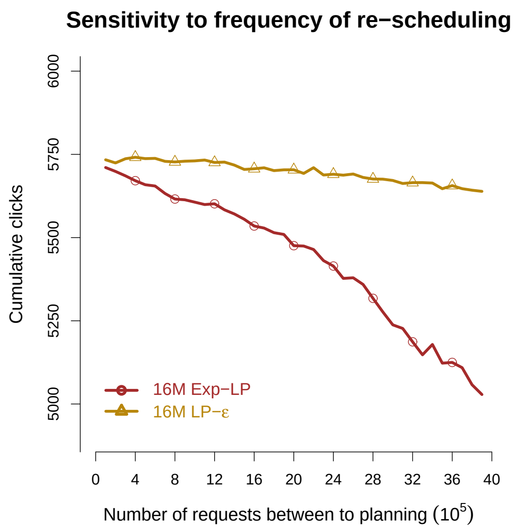

Figure 3: This plot illustrates the sensitivity to the frequency of re-plannings.

**5.5 Uniformity of clicks along ad campaigns** One of the constraints of the problem is that clicks should be gathered at a rather uniform pace along one campaign. To investigate this point, once in the stationary regime, we compute the mean of the clicks over the lifetime of the ads with different values of the horizon $H$. The results are presented in Fig. with ads having a lifetime equal to 50M. 4. Short term strategies ($H = 1$ M) tends to have high click rates in the beginning whereas high values ($H = 50$ M) lead to high click rates at the end of the campaigns. In between ($H = 30$ M), a more uniform click rate is observed. This observation may be used to tune $H$ depending on the website manager, or the announcer preferences. One can prefer smaller values for $H$ (in order to have immediate and sure money), or tune it to achieve a uniform click rate. An other point worth mentioning is the interaction of these observations, and the effect of the horizon of prediction $H$ discussed in sec. 5.3. The use of intermediate values of $H$ (here H=30M) which leads to quite good overall performance in this context (see Fig. 2.a ) results in addition in a quite uniform flow of the click budget.

**5.6 About the importance of having information about the visitors** An other point of interest is the dependence of the performance on the side information available on the visitors. The availability of information about the visitor, the real human being behind the computer, is of well-known high importance. In our study, this information is rather poor: each request holds three binary attributes, which are not very often filled-in (only 20 % of requests have this information filled-in), and when filled-in, the information may be irrelevant. So, we investigated what would be the benefit of having more reliable information. The results are presented in Fig. 5. We can see that in our case, it is almost useless to have this information on more than 50-60% of the visitors.

## 6 Conclusion and future work

In this paper, we have studied the problem of displaying ads to the visitors of a commercial website in order that the company managing the website maximizes its revenue. We have defined the problem, sticking to the management of limited resources (number of ads to display, duration of ad campaigns), in an uncertain environment. In this context, we designed and proposed an algorithm MAB+LP which is a combination of a bandit algorithm to estimate the probability of click of any visitor on any ad, and a combinatorial optimization algorithm to find an optimal solution to the problem. However, due to the discrepancy between estimated

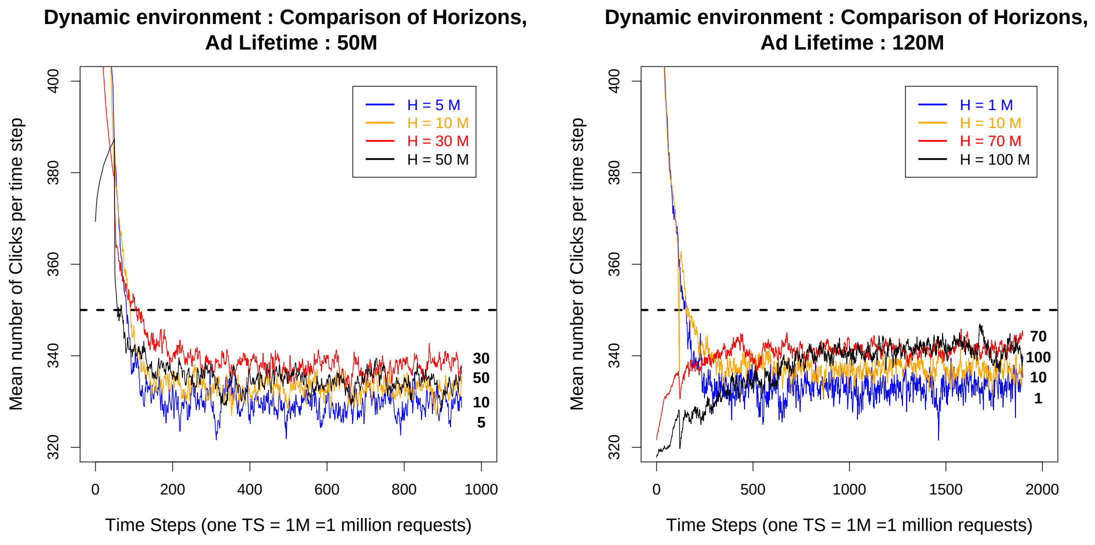

Figure 2: Role of the planning horizon $H$ in a dynamic ad environment, with ads only available during a given lifetime. Results are averaged over 1000 simulations. The frequency of appearance of the new ads $u$ was chosen so that the stationary regime is saturated (more ads to display than the space available on webpages). But still, to fit the reality, we have elected a regime which is not too saturated. The optimal horizon seems to be related to the lifetime of the ads. A new LP is solved every $T = 10,000$ requests (approximately every 3.5 minutes).

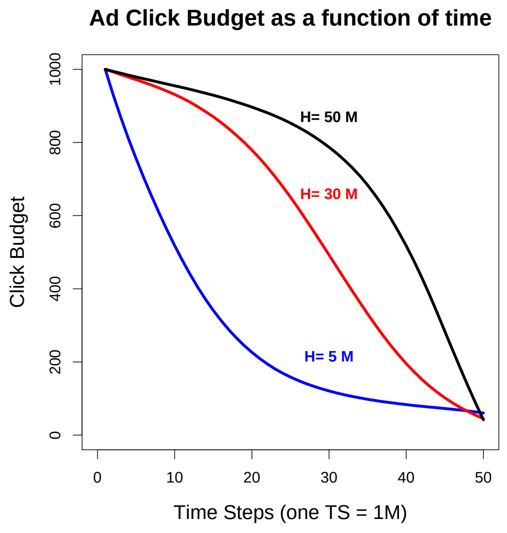

Figure 4: Evolution of the budget of an ad along their lifetime as with regards to the chosen horizon. 40 time steps correspond to 10 days since there is an average of 4 million requests per day.

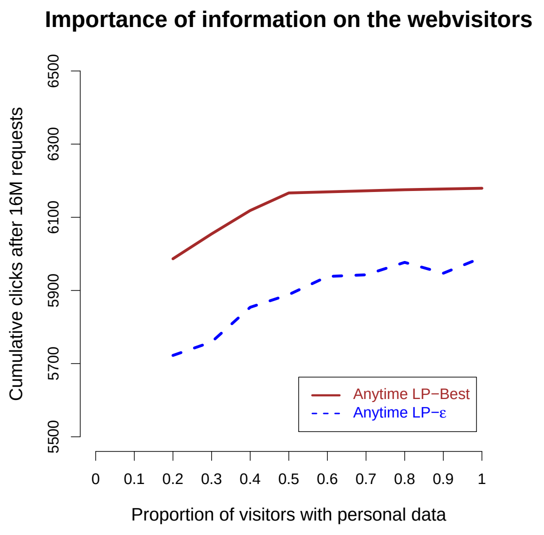

Figure 5: Performance (actual amount of clicks on ads) against the percentage of visitors being fully informed. In our case, the actual percentage is 20%. We see that there is a significant increase in performance up to 50-60%. So, in our setting, it is not necessary to have all requests accurately informed.

probabilities and actual events, this hybrid algorithm has to be iterated regularly along time to update the solution. The proposed algorithm is effective, and is able to increase the click rate on ads, in such a realistic setting. The obtained performance are close to the optimal performance that would be obtained if there was no uncertainty in the problem: this clearly shows that our algorithm MAB+LP is effective at solving this problem.

For this application, some points of our solution are quite useful:

- we tend to use all the ad campaign lifetime,
- one can use the planning in order to estimate the income from a Profile. Especially if you plan to sell a given number of displays of a profile in CPM, you can immediately evaluate the lost in due to the use of the CPC economic model. You just have to compute a new planning without the visits you want to estimate.

Many extensions are rather straightforward such as having different income associated to each ad campaign. More involved, but useful, studies will deal with displaying more than one ad on each webpage, with the way visits are clustered into profiles, and how ad campaigns may also be clustered, to reduce the amount of both.

Though presented here as dedicated to the problem of optimizing ad display on websites, the scope of applications of the algorithm we have proposed is in fact much more general. This means that for practical purpose, the scalability of MAB+LP is meaningful: actually, the slow part of MAB+LP is really the resolution of the combinatorial problem, here handled by LP; the complexity of the estimation of the unknowns is linear in the number of observed events. An other worth studying point is related to the sensitivity of the solution provided by the LP with regards to the uncertainty on the estimated quantities.

**Acknowledgment** The model used in this paper, as well as its parameters have been designed with, and validated by, Orange Labs, to keep up with the essential characteristics of the real problem.

The realization of simulations were carried out using the Grid'5000 experimental testbed, an initiative from the French Ministry of Research, INRIA, CNRS and RENATER and other contributing partners.

## References

[1] N. Abe and A. Nakamura. Learning to Optimally Schedule Internet Banner Advertisements. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 12–21, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[2] Anonymous. Machine Learning Tools for On-line Advertisement. Technical report, 2009.

[3] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning Journal*, 47(2-3):235–256, 2002.

[4] D. Chakrabarti, R. Kumar, F. Radlinski, and E. Upfal. Mortal Multi-Armed Bandits. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS, Neural Information Processing Systems*, pages 273–280. MIT Press, 2008.

[5] O-C Granmo. A Bayesian Learning Automaton for Solving Two-Armed Bernoulli Bandit Problems. *Machine Learning and Applications, Fourth International Conference on*, 0:23–30, 2008.

[6] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. Efficient Bandit Algorithms for Online Multiclass Prediction. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 440–447, New York, NY, USA, 2008. ACM.

[7] J. Langford and T. Zhang. The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 817–824. MIT Press, 2008.

[8] Mohammad Mahdian and Hamid Nazerzadeh. Allocating Online Advertisement Space with Unreliable Es-

timates. In *In ACM Conference on Electronic Commerce*, pages 288–294, 2007.

[9] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and Generalized On-line Matching. In *In FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 264–273. IEEE Computer Society, 2005.

[10] S. Pandey, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for Taxonomies: A Model-based A pproach. In *Proceedings of the Seventh SIAM International Conference on Data Mining*, 2007.

[11] S. Pandey and C. Olston. Handling Advertisements of Unknown Quality in Search Advertising. In Bernhard Schölkopf, John Platt, and Thomas Hoffman, editors, *NIPS, Neural Information Processing Systems*, pages 1065–1072. MIT Press, 2006.

[12] H. Robbins. Some Aspects of the Sequential Design of Experiments. *Bulletin of the American Mathematics Society*, 58:527–535, 1952.

[13] C.-C. Wang, S.R. Kulkarni, and H.V. Poor. Bandit Problems With Side Observations. *IEEE Transactions on Automatic Control*, 50(3):338–355, 2005.