Ad Network Optimization: Evaluating Linear Relaxations

Flávio Sales Truzzi*, Valdinei Freire da Silva†, Anna Helena Reali Costa* and Fabio Gagliardi Cozman*
*Escola Politécnica - Universidade de São Paulo (USP)
Av. Prof. Luciano Gualverto tv. 3, 158 - Caixa Postal 05508-900 - São Paulo - SP - Brazil
†Escola de Artes, Ciências e Humanidades - Universidade de São Paulo (USP)
Av. Arlindo Béttio, 1000, Ermelino Matarazzo - Caixa Postal 03828-000 - São Paulo - SP - Brazil
Email: {flavio.truzzi,valdinei.freire,anna.reali,fgcozman}@usp.br

***Abstract*—This paper presents a theoretical and empirical analysis of linear programming relaxations to ad network optimization. The underlying problem is to select a sequence of ads to send to websites; while an optimal policy can be produced using a Markov Decision Process, in practice one must resort to relaxations to bypass the curse of dimensionality. We focus on a state-of-art relaxation scheme based on linear programming. We build a Markov Decision Process that captures the worst-case behavior of such a linear programming relaxation, and derive theoretical guarantees concerning linear relaxations. We then report on extensive empirical evaluation of linear relaxations; our results suggest that for large problems (similar to ones found in practice), the loss of performance introduced by linear relaxations is rather small.**

Keywords - Ad Network, Markov Decision Process, Linear Programming.

I. Introduction

In this paper we offer a theoretical and empirical analysis of Ad Network optimization, one of the main computational operations in online marketing. The practical impact of this sort of optimization can be understood when we see that online marketing revenue has grown quickly since mid nineties, with a compound annual growth rate of 20.2%. In the first semester of 2012, this market achieved a revenue of 17 billion dollars (USD), an astonishing growth of 14% over the first semester of 2011. The display of ads on banners on websites account for up 21% of this market [1]. For all its importance, not much is known about the theoretical properties of some common models used in Ad Network optimization. In this paper we present a worst-case analysis, based on Markov Decision Processes, of a state-of-art approach to Ad Network optimization based on linear programming.

To understand our problem and our analysis, consider that most online advertising companies follow one of two business models: an online model and an offline model. The online model is represented by real-time bidding, in which advertisers participate in auctions, competing for ad displays, and bidding for particular user profiles. In the offline model, advertisers enter a contract with an Ad Network [2]; that is, with a company that has an inventory of sites. The Ad Network create campaigns with a set of ads, specifying how much each campaign pays for a click, the available time for the campaign, its budget and the minimum number of impressions (i.e. the minimum number of times its ads will be displayed to an user). The Ad Network chooses how to distribute ads to users.

We focus on the offline model of Online Marketing.

There are several pricing schemes in the offline model; the most important are the Cost per Impression (CPI), Cost per Action (CPA), and Cost per Click (CPC). In the CPI scheme the advertiser pays by the number of impressions, i.e., the number of displays of an ad to a user. In the CPA scheme the advertiser pays only when a user makes a specific action, e.g., buy something from the advertisers' store. In the CPC scheme the advertiser pays only when the user clicks on the advertisement. We are focusing on the CPC pricing scheme, because the other models can be converted to it, as we will show later.

The offline business model is, in essence, a sequential decision process: the Ad Network must decide which campaign to display to each user at a specific time, given campaign budgets, values that campaigns pay per click, time constraints of the campaigns, and the relationship between campaigns and user profiles. Ad Network decisions are evaluated based on some utility function; for example, expected revenue.

This sequential decision process can be modeled as a Markov Decision Process (MDP) [3]. The solution of this MDP yields the policy for the Ad Network with the best decision for each possible combination of user profile, and budgets and time of the campaigns. However this approach is computationally intractable even for small problems, as the state space grows exponentially.

One way to avoid the curse of dimensionality in Ad Network optimization is to convert this decision process into a simpler, relaxed problem. Instead of deciding which campaign to allocate for each user profile at each time step, one then selects only the *number* of impressions of each campaign in a given interval of time. Some well known formulations of this relaxed problem resort to linear programming (LP) [5], [6]. These relaxations of the sequential decision process have produced excellent results, but, to the best of our knowledge, no analysis has been published yet on the quality of such relaxations. That is, no theoretical nor empirical analysis has indicated how much is lost by using relaxations, in any sort of "worst-case" sense.

In this article we offer such an analysis. We build a Markov Decision Process that encodes the worst-case behavior of linear programming relaxations, and derive exact results on the loss of performance imposed by relaxations. We then report on extensive experiments that compare linear relaxations and Markov Decision Processes. Our results indicate that for large problems, with sizes close to the size of real problems, the more the budgets of the campaigns grow, the smaller is the difference between the two methods, and the difference may even be ignored in practice. Hence our analysis supports the existing interest in linear relaxations for this important optimization task.

The remainder of this paper is organized as follows. Section II formalizes the problem of Ad Network optimization. In Section III, we formulate the problem as an MDP, and in Section IV we formulate the problem as a relaxed problem to be solved by linear programming (LP). Section V builds a worst-case analysis for the LP formulation. Section VI describes experiments that allow us to highlight and discuss the differences between the MDP and the LP solutions. Finally, Section VII concludes the paper.

II. Problem Definition

Figure 1 depicts the flow of ad distribution in online marketing. Initially advertisers contract the service of an Ad Network to display ads of campaigns in websites. We assume that the advertisers define the campaign previously. Every time a user requests a page in a website (step 1 in Figure 1), the website requests an ad to be displayed (step 2). Users are characterized by their profile (known by the Ad Network). The Ad Network decides which campaign to allocate to the request received, and an ad of the selected campaign is sent to the website (step 3). Then an impression is shown to the user (step 4), who may or may not click on the ad (step 5).

This whole sequential process can be formalized as follows.

At each time t there is a probability P_{req} that a request is issued to a site in the Ad Network's inventory. We consider that the requests follow a Bernoulli distribution with a success probability P_{req}. The set of possible user profiles is $\mathcal{G}$, and $P_G : \mathcal{G} \rightarrow [0,1]$ with $\sum_{i \in \mathcal{G}} P_G(i) = 1$, P_G is the probability function of a user being of a given user profile i.

Let $\mathcal{C}$ be the set of campaigns. A campaign $k \in \mathcal{C}$ is described by a tuple $< B_k, S_k, L_k, cc_k >$, where B_k is the budget of campaign k in number of clicks, S_k is the starting time of the campaign, L_k is the lifetime of the campaign, and cc_k is monetary value that the campaign pays per click. Campaigns can be active or inactive, and only active campaigns can be chosen by the Ad Network. A campaign is active at a specific time t if $S_k \leq t < S_k + L_k$ and $B_k > 0$.

Once the campaign k is selected, its ad is displayed to the user with profile i in a website (an impression is made). The user may or may not click on this ad with probability $CTR(i,k)$, where CTR stands for click-through rate. That is, the CTR is the probability of a click given a pair of user profile and campaign, $CTR : \mathcal{G} \times \mathcal{C} \rightarrow [0,1]$. In real problems CTR values are typically on the order of 10^{-4} [5]. One click generates a revenue equals to cc_k, a portion of this amount goes to the website and the other stays with the Ad Network. The goal of the Ad Network is to choose which campaign to allocate to each request, while maximizing a utility function. We assume the Ad Network to be interested in maximizing expected revenue.

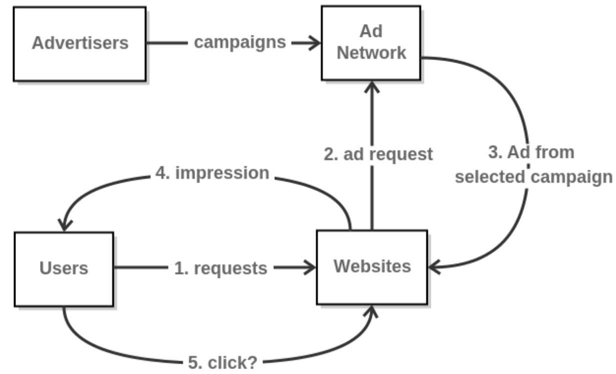

Fig. 1. Dynamics of the ad distribution process.

III. Ad Network as a Markov Decision Process

We now formulate the Ad Network problem as a Markov Decision Process (MDP). The formulation is based on our previous analysis of ad network optimization [3].

MDPs offer a general framework for sequential decision problems. An MDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \tau \rangle$ [8] where $\mathcal{S}$ is the set of all states of the process, $\mathcal{A}$ is the set of all possible actions to be executed at each state $s \in \mathcal{S}$, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ is the transition function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and $\tau \in \mathbb{N}$ is a finite horizon.

The dynamics of an MDP is as follows. At any time $t < \tau$: (i) the process is at state $s \in \mathcal{S}$, (ii) the action $a \in \mathcal{A}$ is executed, (iii) the process generates reward $r_t = \mathcal{R}(s_t, a_t)$, and ($iv$) the process transits to some state $s' \in \mathcal{S}$ with probability $P(s_{t+1} = s' | s_t = s, a_t = a) = \mathcal{T}(s, a, s')$.

To solve an MDP is to find a policy that maximizes the accumulated reward sequence. A non-stationary deterministic policy $\pi : S \times \{0, 1, \ldots, \tau - 1\} \rightarrow A$ specifies which action will be executed at any state $s \in \mathcal{S}$ and at any time $t < \tau$.

Under a policy π, at any time $t < \tau$ every state can be associated with a value that consists of the accumulated reward process induced by the MDP. The expected total reward of a policy π at time i is defined for any state $s \in \mathcal{S}$ as:

$$V^{\pi}(s,i) = \mathrm{E}\left[\sum_{t=i}^{\tau-1} \mathcal{R}(s_t, \pi(s_t, t)) \middle| s_i = s \right]. \tag{1}$$

The value function $V^*(\cdot)$ of an optimal policy can be defined recursively for any state $s \in \mathcal{S}$ and time $t < \tau$ by:

$$V^*(s,t) = \max_{a \in \mathcal{A}_t} \left\{ \mathcal{R}(s,a) + \sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') V^*(s', t+1) \right\}, \tag{2}$$

where $\mathcal{A}_t$ is the subset of $\mathcal{A}$ which contains the possible actions at time t and to be applyed in state s,$V^*(s,\tau) = 0$ for any state $s \in \mathcal{S}$. This recursive approach combines backward induction and Bellman's Optimality Principle [9].

Given the optimal value function $V^*(\cdot)$, an optimal policy can be chosen for any state $s \in \mathcal{S}$ and time $t < \tau$ by:

$$\pi^*(s,t) = \arg\max_{a \in \mathcal{A}_t} \left\{ \mathcal{R}(s,a) + \sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') V^*(s', t+1) \right\}. \tag{3}$$

We now model the Ad Network problem as an MDP by specifying its states, actions, transitions, and rewards.

A. States

The state is modeled as $s = [B_1, B_2, \dots, B_k, G]$, where B_k is the remaining campaign budget as defined in Section II and $G \in \mathcal{G}$ is the user profile that is generating a request; $G = 0$ means that there is no request to be solved. For example, considering 5 campaigns and 3 user profiles, a state is

$$\underbrace{[\overbrace{10,3,4,2,3}^{\text{Campaign Information}}, \overbrace{3}^{\text{Request Information}}]}_{\text{State}}.$$

Information about a campaign indicates how many ads that campaign can afford in that state. In this example, Campaign 1 can afford 10 clicks, Campaign 2 can afford 3 clicks, and so on. The request information contains the information of which user profile has generated a request, in this example user profile 3 has generated the request. From this state, possible next states are: $[9,3,4,2,3,G]$, $[10,2,4,2,3,G]$, $[10,3,3,2,3,G]$, $[10,3,4,2,2,G]$, and $[10,3,4,2,3,G]$, where G can be any user profile or even 0, if there are no requests.

B. Actions

An action gives the allocation of an ad from a campaign of the campaign set $\mathcal{C}$ to a request from a user profile from the set $\mathcal{G}$ in a decision epoch. Given our problem definition our set of actions can be defined by $\mathcal{A} = \{0, 1, \dots, |\mathcal{C}|\}$ with the following meaning:

$$a = k \tag{4}$$

where if $k > 0$, then k is the campaign index. If $k = 0$, then Ad Network does not allocate any campaign to the request.

Recall that campaigns can be active or inactive, hence at any time t a subset of actions $\mathcal{A}_t$ is available. We have that $0 \in \mathcal{A}_t$ for all $t \in [0, \tau - 1]$; and that $k > 0 \in \mathcal{A}_t$ if $S_k \leq t < S_k + L_k$ for all $t \in [0, \tau - 1]$ and $B_k > 0$.

C. Transitions

For all actions a and all states s and s' the function $\mathcal{T}$ must obey the following requirements: $0 \leq \mathcal{T}(s,a,s') \leq 1$, and $\sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') = 1$, i.e. the function defines a proper probability distribution over the possible next states.

The variable G in the state does not depend on the previous state. The component of the state B_k depends only on the previous B_{k+1} and on the occurrence of click events. Given $s = [B_1, B_2, \dots, B_j, G]$ and $s' = [B'_1, B'_2, \dots, B'_j, G']$, the transition function $\mathcal{T}$ is:

$$\mathcal{T}(s,a,s') = P_t(G') \times \prod_{k \in \mathcal{C}} P(B'_k | B_k, a, G), \tag{5}$$

where $P(B'_k | B_k, a, G)$ is equal to:

$$\begin{cases} 1 & \text{if } B'_k = B_k \text{ and } (a \neq k \text{ or } G = 0 \text{ or } B_k = 0), \\ CTR(G,k) & \text{if } B'_k = B_k - 1 \text{ and } (a = k \text{ and } G > 0 \text{ and } B_k > 0), \\ 1 - CTR(G,k) & \text{if } B'_k = B_k \text{ and } (a = k \text{ and } G > 0 \text{ and } B_k > 0), \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

and

$$P_t(G') = \begin{cases} (1 - P_{req}) & \text{if } G' = 0, \\ P_{req} \times P_G(G) & \text{if } G' \in \mathcal{G}. \end{cases}$$

D. Rewards

The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ attaches a reward to each state given an action. In our problem, we have:

$$\mathcal{R}(s, k = a) = \begin{cases} cc_k \times CTR(G,k) & \text{if } k > 0, \ G > 0 \text{ and } B_k > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

where cc_k and $CTR(G,k)$ were defined in Section II and specify respectively the CPC for campaign k and the CTR between campaign k and user profile G. The intuition behind the reward function is that it represents a local evaluation and in the case of Ad Networks it represents the local revenue after choosing to display an ad from campaign k.

IV. A Linear Programming Relaxation

Linear programming focuses on maximization or minimization of a linear function over a polyhedron [10]. In canonical form,

$$\begin{array}{ll} \max & c^T x \\ s.t. & Ax \leq b, \quad x \geq 0, \end{array}$$

where c is a vector that corresponds to the coefficients of the function that is being maximized, x is a vector of variables, A is a matrix, and b is a vector. There are several algorithms to solve a linear program, even strongly polynomial time algorithms [11]. The simplex method is the most commonly used [12]; despite its worst-case exponential time, this method is in average very efficient [7].

The Ad Network problem can be relaxed into a problem that can be solved with linear programming. In this relaxed problem we are interested in discovering the amount of ad displays to be allocated for each campaign in a given interval of time. The description that follows is based on previous efforts [5], [6], [13] with minor modifications.

Let $\mathcal{I}$ be the sorted list of the set defined by $\{S_k\} \cup \{S_k + L_k\}$, i.e. the ordered list of starting and ending times of all campaigns, and let $\mathcal{J}_j = [\mathcal{I}_{j-1}, \mathcal{I}_j[, 1 \leq j < |\mathcal{I}|$, i.e. the intervals defined by the campaign time constraints. Consider the value $\mathbb{T}_j = \sup\{\mathcal{J}_j\} - \inf\{\mathcal{J}_j\}$, this value corresponds to the length of the interval j.

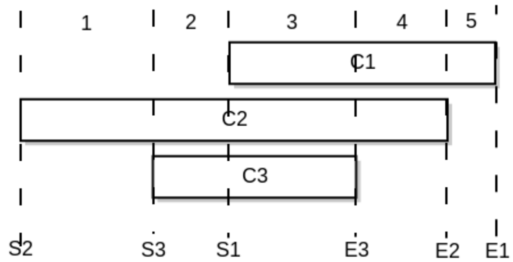

Fig. 2. Example of Interval definition.

For example, in Figure 2 we have three campaigns, with their starting times and ending times, defining 5 intervals, consider that $E_k = S_k + L_k$. In this example we have that: $\mathcal{I} = \{S_2, S_3, S_1, S_3 + L_3, S_2 + L_2, S_1 + L_1\}$, then $\mathcal{J}_1 = [S_2, S_3[$, $\mathcal{J}_2 = [S_3, S_1[$, $\mathcal{J}_3 = [S_1, E_3[$, $\mathcal{J}_4 = [E_3, E_2[$, $\mathcal{J}_5 = [E_2, E_1[$, and $\mathbb{T}_1 = S_3 - S_2$, $\mathbb{T}_2 = S_1 - S_3$, $\mathbb{T}_3 = E_3 - S_1$, $\mathbb{T}_4 = E_2 - E_3$, $\mathbb{T}_5 = E_1 - E_2$.

Then we can formulate this problem as follows:

$$\begin{array}{rc} \max & \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{G}} \sum_{k\in\mathcal{C}} cc_k CTR(i,k) x_{j,i,k} \\ s.t. & \sum_{k\in\mathcal{C}_j} x_{j,i,k} \leq P_{req} P_G(i) \mathbb{T}_j, \forall i \in \mathcal{G}, \forall j \in \mathcal{J} \\ & \sum_{i\in\mathcal{G}} \sum_{j\in\mathcal{J}} CTR(i,k) x_{j,i,k} \leq B_k, \forall k \in \mathcal{C} \\ & x_{j,i,k} \geq 0, \forall j \in \mathcal{J}, \forall k \in \mathcal{C}, \forall i \in \mathcal{G} \end{array}$$

Variables $x_{j,i,k}$ estimate how many ads from campaign k should be displayed to users with user profile i at the interval j. The objective function aims to maximize the total expected revenue of the Ad Network. The first set of constraints ensures that the solution does not exceed the expected number of requests for each user profile i in interval j. The second set of constraints ensures that the expected number of clicks for each campaign does not exceed its budget. The last set of constraints ensures that the solution is feasible for real problems. Without the last set of constraints, it would be possible to create requests for allocations with negative values of $x_{j,i,k}$. Clearly $x_{j,i,k}$ should be integer because there is no possibility to allocate a fraction of an ad, but we can ignore this for now.

Note that this approximation estimates how many ads from campaigns should be shown to each user profile at each interval on average, but it does not provide any clue on how to use this solution. Since clicks and ad requests occur following a Bernoulli distribution, we have near 0.5 probability that $x_{j,i,k}$ is over estimated and near 0.5 probability that $x_{j,i,k}$ is underestimated in any given instance; and estimates of $x_{j,i,k}$ are hard to obtain.

Girgin et al. [5] proposed two ways to use this solution. The Highest LP Policy (HLP) chooses the campaign with $\pi_{LP}(i,j) = \arg\max_k \ x_{j,i,k}/\sum_k x_{j,i,k}$. The Stochastic LP Policy (SLP) chooses stochastically with respect to $x_{j,i,k}/\sum_k x_{j,i,k}$.

This CPC formulation is very versatile as it can be converted to a CPI formulation by just setting

$$CTR(i,k) = 1 \quad \forall i \in \mathcal{G}, \forall k \in \mathcal{C}.$$

It turns out that the CPA and CPC has the same formulation, but $CTR(i,k)$ values would be lower.

To finish this section, we compare the complexity of the MDP formulation and the LP relaxation.

In the LP formulation, if boundary constraints 8 are not considered, the number of constraints is of order $O(|\mathcal{J}| \times |\mathcal{G}| + |\mathcal{C}|)$, but by definition $1 \leq |\mathcal{J}| \leq 2 \times |\mathcal{C}|$. So the number of constraints is of order $O(|\mathcal{G}||\mathcal{C}|)$w, while the number of variables is of order $O(|\mathcal{G}||\mathcal{C}|^2)$ for the same reason.

The size of the policy to be found is equal to $|\mathcal{S}| \times |\{1, 2, \ldots, \tau\}|$, and $|\mathcal{S}| = (|\mathcal{G}| + 1) \times \prod_{k\in\mathcal{C}} (B_k + 1)$, if we consider $B_{min} = \min_{k\in\mathcal{C}}\{B_k\}$, it follows that $|\mathcal{S}| \geq (|\mathcal{G}| + 1) \times (B_{min} + 1)^{|\mathcal{C}|}$. This makes the MDP solution intractable even for small problems because of its memory requirements. In real settings there are hundreds of campaigns with budgets of thousands of clicks.

V. MDP VERSUS LP: A WORST-CASE ANALYSIS

In the previous section we showed that solving the LP approximation is less computationally intensive than solving the MDP. Whereas the LP formulation grows quadratic within the number of campaigns, the MDP formulation grows exponentially within the number of campaigns. LP formulation also approximates discrete variables to continuous, so that in LP formulation computational cost does not depend on budget and horizon sizes, whereas in the MDP formulation it does. Despite LP formulation being much more desirable regarding computational cost, it does not present optimality. In this section we show why LP formulation is not optimal and design a worst case regarding revenue performance.

LP formulation is constructed by considering a deterministic problem, but the Ad Network problem is a stochastic one; LP formulation approximates such a stochastic problem by its expected counterpart. First, user profile requests follow multinomial distribution with means $(1 - P_{req}), P_{req}P_G(1), P_{req}P_G(2), \ldots, P_{req}P_G(|\mathcal{G}|)$; since Ad Network actions do not affect user profile requests, making user profile requests deterministic does not affect optimal solution. Second, clicks in ads occur following a binomial distribution with success rate given by $CTR(i,k)$ but limited to budget; since in every instance obtained after executing a solution (be it HLP or SLP) clicks in each campaign are limited to the respective budget, in this case making clicks deterministic does affect optimal solution. Because LP formulation does not consider the click dynamics, in the worst case, i.e., when clicks occur bellow mean, LP solution allocate campaigns with low CTR, whereas high CTR campaigns should be tried. Girgin et al. [5] suggest artificially increasing budget of high CTR campaign, although is difficult to say how much should be increased to improve the results, and even in this case we can not reach optimality.

Since LP solution is inherently suboptimal, we can ask how far from the MDP solution is the LP solution. We answer such a question by considering a simple set-up: 1 user profile and 2 campaigns. In this set-up we can devise 3 cases: (*i*) campaigns that do not share time intervals, (*ii*) campaigns with time intervals partially overlaped, and (*iii*) a campaign with time intervals included on the time intervals of the other.

For clarity consider $ctr_k = CTR(1,k)$. We analyze specific instances of the case (*iii*) when: $P_{req} = 1$, $S_1 < S_2$,

$L_1 \to \infty$, $L_2 \to \infty$, $ctr_2 \to 0$, $B_2 = 1$, $cc_1 = cc_2 = 1$, $ctr_1 = \frac{B_1}{P_{req}(S_2 - S_1)}$, and $0 < \lim_{L_2 \to \infty, ctr_2 \to 0} L_2 \times ctr_2 < \epsilon$, and this limit exist if an appropriate path to L_2 and ctr_2 is chosen [1].

With this setting it is clear that the optimal solution to this problem is exploiting the campaign 1 until its budget be depleted; and when budget from campaign 1 is equal to zero start exploiting campaign 2. Since $\mathbb{T}_1 = S_2 - S_1$ and $\mathbb{T}_2 \to \infty$, the LP solution to this problem gives that for the first time interval it would allocate $P_{req} \times \mathbb{T}_1$ requests to campaign 1, whilst for the second time interval it would allocate $P_{req} \times \mathbb{T}_2$ requests to campaign 2. We can now enunciate the following propositions.

Proposition 1: The value $V^{\pi_{MDP}}$ obtained under MDP formulation for the worst case above is restricted to:

$$B_1 < V^{\pi_{MDP}} < B_1 + \epsilon.$$

Proof: Consider P^t_{remain} the probability that at time t there still remains budget in campaign 1. Since $\mathbb{T}_2 \to \infty$ and $ctr_1 > 0$, there exists $t_{\epsilon_1} > 0$ such that $P^{t_{\epsilon_1}}_{remain} > \frac{\epsilon_1}{B_1}$. Consider $\epsilon_1 = \lim_{L_2 \to \infty, ctr_2 \to 0} L_2 \times ctr_2$ which represents the value of presenting campaign 2 infinitely. Then,

$$V^{\pi_{MDP}} > B_1(1 - P^{t_{\epsilon_1}}_{remain}) + \epsilon_1 > B_1.$$

The upper bound is trivial given that $\lim_{L_2 \to \infty, ctr_2 \to 0} L_2 \times ctr_2 < \epsilon$. ■

Proposition 2: The value $V^{\pi_{LP}}$ obtained under LP formulation for the worst case above is restricted to:

$$\begin{aligned} V^{\pi_{LP}} < & \sum_{b=1}^{B_1} b \times \binom{\mathbb{T}_1}{b} (ctr_1)^b (1 - ctr_1)^{\mathbb{T}_1 - b} \\ & + \sum_{b=B_1+1}^{\mathbb{T}_1} B_1 \times \binom{\mathbb{T}_1}{b} (ctr_1)^b (1 - ctr_1)^{\mathbb{T}_1 - b} + \epsilon \end{aligned}.$$

Proof: Since campaign 1 is explored only on the first interval, the Ad Network has $\mathbb{T}_1$ trials to consume budget B_1. Then, the value obtained in the first period is given by the expected number of success limited to B_1 in a binomial distribution with $\mathbb{T}_1$ trials and success rate ctr_1. The factor ϵ stand for the condition on $\lim_{L_2 \to \infty, ctr_2 \to 0} L_2 \times ctr_2$. ■

Proposition 3: The relative performance of MDP solution against LP solution in the worst case is given by the limit

$$\lim_{\mathbb{T}_1 \to \infty} \left[1 - \left(1 - \frac{1}{\mathbb{T}_1} \right)^{\mathbb{T}_1} \right]^{-1} = \frac{e}{e-1} \approx 1.582.$$

Proof: The relative performance is given by the ratio $\frac{V^{\pi_{MDP}}}{V^{\pi_{LP}}}$. Taking into account previous propositions and that $ctr_1 = \frac{B_1}{\mathbb{T}_1}$, we have the worst case for $B_1 = 1$, and $\epsilon \to 0$. Then, $V^{\pi_{MDP}} \to B_1 = 1$ and

$$V^{\pi_{LP}} \to \sum_{b=1}^{\mathbb{T}_1} \binom{\mathbb{T}_1}{b} (ctr_1)^b (1 - ctr_1)^{\mathbb{T}_1 - b} = 1 - \left(1 - \frac{1}{\mathbb{T}_1} \right)^{\mathbb{T}_1}.$$

$V^{\pi_{LP}}$ attains its minimum when $\mathbb{T}_1 \to \infty$. Since $V^{\pi_{MDP}} \to 1$, the relative performance is simply given by $(V^{\pi_{LP}})^{-1}$. ■

The relative performance denotes how much better the MDP is solution compared to the LP solution. We have that in the worst case the relative performance is 1.582, i.e., the MDP solution can be 58.2% better than the LP solution. However, usually $B_k \gg 1$ in real problems and we show empirically that the relative performance is much smaller than 1.582 in this case.

[1]For example: $L_2 = t^2$, $ctr_2 = \frac{1}{t^2}$, with $t \to 0$.

VI. Experiments

In order to compare the results of the MDP and the Approximation by Linear Programming we conduct a simple experiment within the (iii) set-up of section V, but consider different settings for parameters B_1, B_2, $\mathbb{T}_1$, $\mathbb{T}_2$, and $CTR(1,2)$, whilst still setting

$$CTR(1,1) = \frac{B_1}{P_{req}(S_2 - S_1)} = \frac{B_1}{P_{req}\mathbb{T}_1}.$$

We conduct experiments to show how the ratio $\frac{V^{\pi_{MDP}}}{V^{\pi_{LP}}}$ evolves when the budget increases in 4 different scenarios. In every scenario we set $\mathbb{T}_1 = 50,000$. The first scenario considers the worst case, i.e., $B_2 = 1$, $\mathbb{T}_2 \to \infty$, $CTR(1,2) \to 0$. The second scenario relaxes the size of the second interval by using $\mathbb{T}_2 = 50,000$, in this case the Ad Network would not have infinity time to consume budget of campaign 1. The third scenario relaxes the budget and CTR of campaign 2 by using $B_2 = B_1$, $CTR(1,2) = 0.1 \times CTR(1,1)$; in this case campaign 1 is much more attractive than the campaign 2, but the LP solution can now make some revenue in the second interval. Finally, the fourth scenario relaxes CTR of campaign 2 to be closer to CRT of campaign 1 by setting $CTR(1,2) = 0.5 \times CTR(1,1)$. Note that, from scenario 1 to 4, the value of the LP solution get closer and closer to the value of MDP solution.

Figure 3 shows the relative performance of the solutions in the four scenarios. We can clearly see that the difference between the optimal solution to the approximation is a decreasing function of the budget size. For clarity, Figure 3 shows only until $B_k = 50$. However we run the experimentuntil $B_k = 200$, we got a difference of about 2.5% in the worst scenario and about 2.0% in the better scenario.

We also calculate in the four scenarios the relative performance for a budget equals to 500 clicks, and Table I shows the result. Real life problems have in average a budget of 10,000 clicks. However, since in this case the MDP model presents a huge state space ($|\mathcal{S}| = 10^8$), we only calculate the relative performance in the worst case with $\mathbb{T}_1 = 10^8$ and $CTR(1,1) = 10^{-4}$. In this case we have a relative performance of 1.0040, meaning that the MDP formulation offers less than 0.4% of improvement when compared to the LP solution.

TABLE I. Relative performance of MDP against LP solution for interval size 50,000 and $B_1 = 500$.

Scenario 1	Scenario 2	Scenario 3	Scenario 4
1.0181	1.0181	1.0164	1.0120

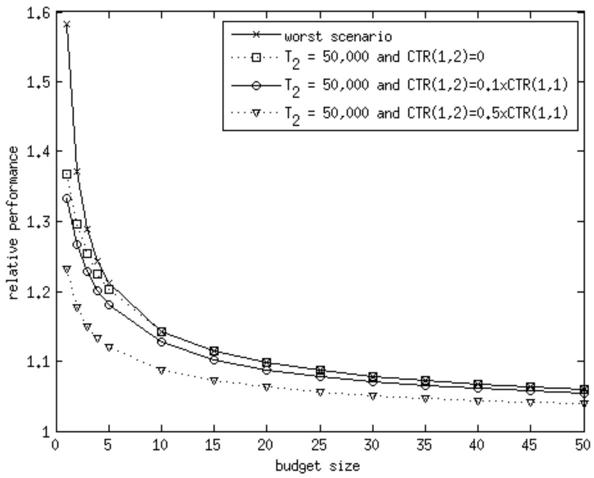

Fig. 3. Relative performance of MDP against LP solution. Fixed time and increasing budgets.

VII. CONCLUSION

In this paper we investigated the difference between the optimal solution and an approximation formulated as a linear program in the problem of displaying ads on web pages using the cost per click model.

We constructed four scenarios, including a worst case scenario, and for a budget of one click we calculated the relative performance of the MDP solution when compared to the LP solution. In the worst case scenario the MDP solution can be 58.2% better than the LP solution. However, our experiments shows that as the budget of campaigns grows, this difference falls quickly. For a small budget of 50 clicks, this difference falls to about 6.0% in the worst case scenario and to about 4.0% in a better scenario. For a budget equal to 500 clicks, this difference drops to 1.81% in the worst case and 1.2% in the better scenario. Finally, when we have a budget of real problems such as 10,000 clicks, the difference in the worst case is only of 0.4%.

With this knowledge of the relative performance in mind, we can conclude that the LP solution is not far of the optimal solution, and this difference may be disregarded, since real sized problems have hundreds of campaigns and their budgets are in the order of thousands of clicks, making this difference even smaller.

ACKNOWLEDGMENTS

Flávio Sales Truzzi is supported by CAPES. This research was partly sponsored by FAPESP – Fundação de Amparo à Pesquisa do Estado de São Paulo (Procs. 11/19280-8 and 12/19627-0) and CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico (Procs. 311058/2011-6 and 305395/2010-6).

REFERENCES

[1] I. A. Bureau, "IAB Internet Advertising Revenue Report 2012," no. October, 2012.

[2] S. Muthukrishnan, "Ad exchanges: Research issues." in *Workshop on Internet and Network Economics (WINE 2009)*, 2009, pp. 1–12.

[3] F. S. Truzzi, V. Freire, A. H. R. Costa, and F. G. Cozman, "Markov Decision Processes for Ad Network Optimization," in *Encontro Nacional de Inteligencia Artificial (ENIA)*, 2012.

[4] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[5] S. Girgin, J. Mary, P. Preux, and O. Nicol, "Managing advertising campaigns - an approximate planning approach," *Frontiers of Computer Science*, vol. 6, no. 2, pp. 209–229, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11704-012-2873-5

[6] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur, "Real-time bidding algorithms for performance-based display ad allocation," *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '11*, p. 1307, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2020408.2020604

[7] A. Schrijver, *Theory of Linear and Integer Programming*. Wiley, 1998.

[8] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, Apr. 1994.

[9] D. Bertsekas, *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall International, 1987. [Online]. Available: http://books.google.com.br/books?id=-6RiQgAACAAJ

[10] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Courier Dover Publications, 1998.

[11] E. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs," *Operations Research*, vol. 34, no. 2, pp. 250–256, 1986.

[12] G. B. Dantzig, "Maximization of a linear function of variables subject to linear inequalities," in *The Basic George B. Dantzig*, R. W. Cottle, Ed., 2003, pp. 24–32.

[13] M. Langheinrich, A. Nakamura, N. Abe, T. Kamba, and Y. Koseki, "Unintrusive customization techniques for web advertising," *Computer Networks*, vol. 31, no. 11, pp. 1259–1272, 1999.