# Markov Decision Processes for Ad Network Optimization

**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 examine a central problem in a particular advertising scheme: we are concerned with matching marketing campaigns that produce advertisements ("ads"), to impressions — where "impression" is a general term for any space in the internet that can display an ad. In this paper we propose a new take on the problem by resorting to planning techniques based on Markov Decision Processes, and by resorting to plan generation techniques that have been developed in the AI literature. We present a detailed formulation of the Markov Decision Process approach and results of simulated experiments.*

## 1. Introduction

Internet marketing has skyrocketed in recent years. Most companies adopt rudimentary forms of marketing, while others resort to sophisticated ideas based on collaborative, or even social, recommendations [Kiang et al. 2000, Nagurney 2010]. Sources report that Internet advertising revenues in the United States of America totaled $31.7 billion for the full year of 2011, and a compound annual growth rate (CAGR) of 20.3% since 2002 [Pwc 2012].

In this paper we examine a central problem in a particular advertising scheme. We are concerned with matching marketing campaigns to impressions. A campaign offers products through advertisements, termed *ads*. An *impression* is any space in the internet that can be occupied by an ad. For example, the space for a banner in a blog page is an impression. Campaigns have ads to be displayed, and impressions from publishers (blogs and websites alike) request ads to be displayed by them. Campaigns pay publishers to display ads; for instance if a blog carries an ad for a car maker, the car maker will pay the blogger — either a fixed amount per impression, or a fixed amount per actual sale based on the impression; these are contractual arrangements between the campaign and the publisher. Clearly there is interest in placing ads correctly, so that a campaign is presented to the right public, thus increasing the effect of the campaign, and the revenue of the publisher. A simple example: we should expect that a blog specialized in motherhood should carry campaigns geared toward women.

To make matters concrete, we consider a fixed set of campaigns, a stream of impressions that request ads to be selected, and a broker, normally referred to as an *ad network*, that matches campaigns and impressions. The revenue of the broker comes from the campaigns, and again it can vary depending on contractual obligations; we present details about this later.

The problem of matching campaigns and impressions has received attention, as our references, discussed later, indicate. The problem is normally modeled as a constrained optimization problem that maximizes the revenue subject to constraints such as budget limits and inventory availability. The state of the art is to solve the optimization problem off-line as a linear programming problem and serve ads using this optimal solution as a guidance to the online ad allocation [Chen et al. 2011]. However, such a solution ignores the obvious fact that impressions are generated dynamically, and the whole problem is one of dynamic, sequential decision making; that is, a planning problem. In this paper we propose a new take on the problem by resorting to planning techniques based on Markov Decision Processes, and by resorting to plan generation techniques that have been developed in the AI literature. Section 2 defines our problem formally, setting the stage for our solution.

Our strategy was to build a simulator that can reproduce the behavior of campaigns, publishers, and the ad network. Even though the simulator is not a contribution in itself, we describe it in some detail in Section 3 because its behavior encodes some of the assumptions we are making concerning ad networks and internet advertising in general. Sections 4 and 5 respectively present the state of the art solution for ad-impression matching, based on linear programming, and our own proposal using Markov Decision Processes. Finally, Section 6 shows our proposal in action, while Section 7 concludes the paper.

## 2. Problem Definition: Basic Concepts and Assumptions

An Ad Network is a network of sites that promotes the distribution of ads from advertisers to publishers. There are rules and constraints concerning the distribution of advertisements, and there is potential revenue for advertisers and publishers as users act in response to advertisement. This paper consider the revenue model based on *cost per click* (CPC) [Devanur and Kakade 2009]. This model returns a fixed monetary value each time a user clicks in an ad. The *effective cost per impression* (eCPI) is the total revenue produced by the revenue model in terms of impressions.

Important concepts are [Muthukrishnan 2009]:

- *Advertisers* produce ads to be presented and allocate them into campaigns.
- *Publishers* are sites containing ads sent by the Advertisers and each ad spot in a publisher is a *source*.
- *Campaigns* are designed by the Advertisers, and are mainly characterized by: a set of ads, a starting date, an ending date, and a revenue model. It is also common

that an advertiser contracts a minimum volume of impressions and if CPC is considered as revenue model, then a campaign is usually constrained by a maximum budget to be spent [1].

- *Ads* contain the material to be distributed and the manifestation of an ad is considered an *Impression* from a campaign (advertiser) to a site (publisher).
- *Requests* are issued by sources (sites, publishers) that need ads to be presented.

The main objective of an Ad Network is to maximize eCPI across a period of time; in other words, to find an optimal policy to distribute ads from campaigns to publishers, so as to fulfill requests and to satisfy constraints on minimum volume and budget.

We assume that the ad network considers only the level of campaigns, i.e., ads in a campaign are considered equally. Moreover, we assume that the advertiser of a campaign does not interfere in the allocation of ads to publishers. We also merge sites into groups, such that sites in the same group present similar responses to impressions from the same campaign.

The response from a group $i$ to an impression of a campaign $j$ is modeled by the *Click-Through Rate* (CTR), the probability $p_{ij}$ that a click is made in some ad given a *group* and a *campaign*. Note that sites in the same group are assumed to have the same CTR regarding ads in the same campaign.

We also consider that ads in each campaign are available before the beginning of the requests, and that the ad network receives a stream of request which must be filled with such ads. This problem allows some trivial solutions in some cases. If there is no budget limit, then the optimal solution to the problem can be made considering only the eCPI, i.e. by choosing the campaign with the largest eCPI for each impression. On the other hand, if campaigns have no ending date, then the optimal solution is to choose campaigns with positive eCPI.

Common solutions in the literature are obtained in two steps. The off-line step assumes that an estimate of the stream of requests is known *a priori* and obtains an optimal solution for it. The on-line step adapts the off-line optimal solution to allocate ads to requests as they appear in the stream of request.

## 3. Simulator

We have built a simulator that generates a stream of requests from publishers, and a pool of ads from campaigns; from these the simulator manages a policy generator that can be for instance an MDP-based policy, and produces statistics concerning performance and revenue. The simulator was coded in the Python language and Matlab, and even though it is not by itself an innovative piece, it encodes a number of assumptions that are worth enumerating and justifying. First, we do not impose constraints on the minimum volume of impressions; these are not the most important constraints and we may introduce them in future work. Additionally, the following assumptions were adopted:

1. the ad network does not differentiate publishers, therefore is indifferent to consider publishers or groups (of publishers) provided that the click response to the impressions is sufficient when working with the group abstraction;

[1]It is also common that campaigns constrain publishers that may receive ads, for instance, by defining main and secondary categories, but we consider it to be irrelevant in this work.

2. although the campaigns are originated from the advertisers, the ad network can treat the requests at a campaign level, with no concern for the description of advertisers;
3. future requests are not influenced by requests occurred in the past;
4. time is discrete; and
5. campaigns start in the beginning of the simulation and finish in the end of the simulation.

A simulator can be defined by a tuple $< \tau, \mathcal{G}, \mathcal{C}, P_{CTR}, P_G, P_{request} >$, where:

1. $\tau \in \mathbb{N}$ specifies the length of the simulation;
2. $\mathcal{G}$ defines the group set; $\mathcal{C}$ defines the campaign set (includes CPC and budget);
3. $P_{CTR} : \mathcal{G} \times \mathcal{C} \to [0, 1]$ specifies the CTR for all group-campaign pairs, and it is considered that the CTR does not change over time; and
4. $P_G : \mathcal{G} \to [0, 1]$, specifies the probability of an impression belonging to a group $i$, and $\sum_{i \in \mathcal{G}} P_G(i) = 1$.

For each discrete time step $t < \tau$ the simulator may generate one impression following a Bernoulli process with success probability $P_{request}$, and the request is associated to one of the possible groups according to the probability $P_G$. The association of a request with a group is denoted by a vector $I = [I_1, I_2, \ldots, I_{|\mathcal{G}|}]$, where $I_i = 1$ if the request is associated with the group $i$, $I_i = 0$ otherwise. In this formulation we consider that only one request can occur per time step, this simplification can be made without loss of generality. After receiving the request vector $I$, the ad network must allocate impressions by choosing a campaign for each request, responding with a $|\mathcal{G}| \times |\mathcal{C}|$ matrix $A \geq 0$ such that for all $1 \leq i \leq \mathcal{G}$:

$$\sum_{j=1}^{|\mathcal{C}|} A_{ij} \leq I_i.$$

After receiving the allocation matrix $A$ the simulator generates the click that indicates whether the impression allocated to a campaign has been clicked in that simulation time according to a Bernoulli distribution with success probability equals to the $P_{CTR}$ of the pair campaign-group.

The simulator dynamics for each discrete time step $t$ is:

1. generates the impression vector $I$;
2. receives an allocation matrix $A$; and
3. generates the click event for the pair campaign-group.

## 4. Policy Generation Through Linear Programming

The state of the art solution for policy generation in ad networks seems to be the proposal by Chen et al [Chen et al. 2011], that we summarize here to fix notation and provide motivation for our own approach, to be presented in the next section. We have that:

1. $i$ indexes $|\mathcal{G}|$ groups, and $j$ indexes $|\mathcal{C}|$ campaigns;
2. $P_{CTR}(i, j)$ denotes the CTR of group $i$ impression assigned to campaign $j$, $q_j$ denotes the CPC for campaign $j$ (how much a campaign is paying for a click), and:

$$v_{ij} = P_{CTR}(i, j) q_j \tag{1}$$

is the expected cost per impression (eCPI) of such assignment;

3. $g_j$ is the budget for campaign $j$;
4. $h_i$ denotes the request availability constraint for group $i$; and
5. $x_{ij}$ denotes the number of impressions from group $i$ allocated to campaign $j$.

Ad allocation can be reduced to a linear program *if* we assume that all impressions are fixed at the beginning; in this case we have to solve:

$$\begin{aligned} \max_{x} \quad & \sum_{i,j} v_{ij} x_{ij} \\ \text{s.t.} \quad & \forall j, \sum_{i} v_{ij} x_{ij} \leq g_j, \\ & \forall i, \sum_{j} x_{ij} \leq h_i, \\ & \forall i, j, x_{ij} \geq 0. \end{aligned} \tag{2}$$

The solution presented in Expression (2) is optimal, but it requires knowledge of $h_i$ (number of requests available to the group $i$). On the other hand, $h_i$ is a random variable, i.e. in each run (day, week, month) the number of the requests of impressions is different. However, under the assumption that the probability distribution of the variable $h_i$ is independent of the allocation made at that run, it's possible to estimate that distribution. Then, it is possible to find an off-line solution using the mean value $\bar{h}_i$ of such distribution. Chikering and Heckerman (2000) suggest using the estimation $\bar{h}_i$ to turn the problem in Expression (2) a well-posed problem and to obtain a solution $\bar{x}_{ij}$. Getting back to our simulator, $\bar{h}_i$ can be easily calculated by:

$$\bar{h}_i = \tau P_{request} P_G(i).$$

Then, at any time $t$, after receiving a request from group $i$, a campaign $j$ is allocated to such request with probability:

$$\frac{\bar{x}_{ij}}{\sum_{i'} \bar{x}_{i'j}}.$$

The problem with such solution is that it considers only $\bar{h}_i$, instead of considering the whole probability distribution over $h_i$. In the next section we model the ad network problem as a Markov Decision Process which considers the whole dynamics of requests from groups.

## 5. Markov Decision Process Formulation

We now present our own proposal for ad allocation. We start by reviewing the basics of Markov Decision Processes (MDPs).

MDPs offer a general framework for sequential decision problems. An MDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \tau \rangle$ [Puterman 1994] as follows:

- $\mathcal{S}$ 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 $s_t \in \mathcal{S}$, (*ii*) the action $a_t \in \mathcal{A}$ 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 $P(s_{t+1} = s'|s_t = s, a_t = a) = \mathcal{T}(s, a, s')$.

The solution for an MDP is a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maps each state into an action. An optimal policy maximizes the following value function for all state $s \in \mathcal{S}$ ,

$$V^\pi(s) = \mathrm{E}\left[\sum_{t=0}^{\tau-1} r_t \middle| s_0 = s, \pi\right].$$

$V^\pi(s)$ defines the expected accumulated reward in the horizon $\tau$.

We now model the ad network problem as an MDP by considering states, actions, transitions, and rewards.

## 5.1. Ad Network as an MDP

### States

The state is modelled as $s = [B_1, B_2, \ldots, B_j, I_1, I_2, \ldots, I_{|\mathcal{G}|}]$, where $I_i$ was defined in section 3 and $B_i$ is defined as the number of available ads in a campaign budget. For example, considering 5 campaigns and 3 groups, a possible state is

$$\underbrace{[\ \overbrace{10, 3, 4, 2, 3}^{\text{Campaign Information}}\ ,\ \overbrace{0, 0, 1}^{\text{Group Information}}\ ]}_{\text{State}}.$$

The values on the campaign information tells how many ads that campaign can afford in that state, in the example above, the campaign 1 can afford 10 impressions, campaign 2 can afford 3 impressions and so on. The group information contains the information of which group has generated a request, in the example above group 1 and 2 have not generated any requests, and group 3 has generated one request that should be allocated.

### Actions

In this model an action is defined as the allocation of an ad of the campaign set $\mathcal{C}$ to any impressions of the group set $\mathcal{G}$ in a decision epoch. Given our problem definition our set of actions can be defined by $\mathcal{A} = \{0, 1, \ldots, |\mathcal{C}|\}$ with the following meaning:

$$a = j \tag{3}$$

where $j > 0$ is the campaign index. It is possible to not allocate an impression to any campaign by assigning $a = 0$.

### 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 variables in vector $I$ do not depend on the previous state and $I$ is described in section 3, the $B$ component of the state depends only on the previous $B$ and in the occurrence of click events, given $s = [B_1, B_2, \ldots, B_j, I_1, I_2, \ldots, I_i]$ and $s' = [B'_1, B'_2, \ldots, B'_j, I'_1, I'_2, \ldots, I'_i]$, the transition function $\mathcal{T}$ is:

$$\mathcal{T}(s, a, s') = P_t(I') \times \prod_{j \in \mathcal{C}} P(B'_j | B_j, a, I) \tag{4}$$

where $P(B'_j | B_j, a, I)$ is equal to:

$$P(B'_j | B_j, a, I) = \begin{cases} 1 & \text{, if } B'_j = B_j \text{ and } (a \neq j \text{ or } I_i = 0 \text{ or } B_j = 0) \\ P_{CTR}(i,j) & \text{, if } B'_j = B_j - 1 \text{ and } (a = j \text{ and } I_i = 1 \text{ and } B_j > 0) \\ 1 - P_{CTR}(i,j) & \text{, if } B'_j = B_j \text{ and } (a = j \text{ and } I_i = 1 \text{ and } B_j > 0) \\ 0 & \text{, otherwise} \end{cases} \tag{5}$$

and

$$P_t(I') = \begin{cases} (1 - P_{request}) & \text{, if } I'_i = 0 \text{ for all } i \in \mathcal{G} \\ P_{request} P_G(i) & \text{, if } I'_i = 1 \text{ and } I'_{i'} = 0 \text{ for all } i' \neq i \in \mathcal{G} \\ 0, & \text{otherwise} \end{cases}$$

**Rewards**

The reward function $\mathcal{R}$ is defined as $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. It attaches a reward to each state given an action, i.e. a value is obtained by performing an action on the current state. In our model it is defined as follows:

$$\begin{aligned} \mathcal{R}(s,a) &= \sum_{j \in \mathcal{C}} \mathrm{E}\left[ q_j \times (B_j - B'_j) \right] \\ &= \begin{cases} q_j P_{CTR}(i,j) & \text{, if } I_i = 1 \text{ and } B_a > 0 \\ 0 & \text{otherwise} \end{cases}, \end{aligned} \tag{6}$$

where $q_j$ and $P_{CTR}(i,j)$ were defined in section 4 and define respectively the CPC for campaign $j$ and CTR between campaign $j$ and group $i$. The intuition behind the reward function is that it represents a local evaluation and in the case of ad networks, this local evaluation happens to be the eCPI.

## 5.2. Generating policies using the MDP

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

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

$$V^{\pi}(s,t) = \mathrm{E}\left[ \sum_{i=t}^{\tau-1} \mathcal{R}(s_i, \pi(s_i, i)) \right]. \tag{7}$$

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:

$$V^*(s,t) = \max_{a \in \mathcal{A}} \left\{ \mathcal{R}(s,a) + \sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') V^*(s', t+1) \right\}, \tag{8}$$

where $V^*(s,\tau) = 0$ for any state $s \in \mathcal{S}$.

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:

$$\pi^*(s,t) = \arg\max_{a \in \mathcal{A}} \left\{ \mathcal{R}(s,a) + \sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') V^*(s', t+1) \right\}. \tag{9}$$

## 6. A Case Study

In this section we present the results of our MDP-based solution and compare its results with a probabilistic approach as presented in section 4 and a greedy approach that allocates ads to impressions with higher eCPI. All graphs that show the results of the experiments were made by calculating the average of 1000 executions.

### 6.1. Experiment 1

In this experiment we want to analyze what would be the effect of using different time limit $\tau$ for each approach: greedy, probabilistic and MDP. For this experiment the following parameters were set: $|G| = 3$, $|C| = 3$ and $B(C_i) = 10$ for all $i \in \mathcal{C}$; $P_{CTR}$ is defined in table 1, $P_G$ is defined in table 2, $P_{request} = 0.4$, and we considered three different $\tau$: 100, 150, and 200.

**Table 1. CTR values**

| | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|
| $G_1$ | 0.05 | 0.50 | .05 |
| $G_2$ | 0.80 | 0.81 | .05 |
| $G_3$ | 0.05 | 0.81 | .80 |

**Table 2. $P_G$ values**

| Group | $P_G$ |
|---|---|
| $G_1$ | .50 |
| $G_2$ | .25 |
| $G_3$ | .25 |

The parameters $P_{CTR}$, $P_{request}$, and $P_g$ were chosen in order to show the effect of a different time limit in each one of the three methods: greedy, probabilistic and MDP.

In the top-left graph of Figure 1, $\tau$ was set to 100. In this case the probabilistic and the MDP approaches outperformed the greedy one. The greedy approach starts allocating requests to the campaign with higher eCPI ($C_2$), and after the end of the budget of this campaign its performance begins to degrade; this effect can be seen in the decrease of the curve slope near $t = 40$, while the MDP and the probabilistic approaches maintain almost the same results.

In the top-right graph of Figure 1, $\tau$ was set to 150, and the effect of time can be seen on both MDP and probabilistic methods. The probabilistic maintains almost a straight line until the budgets are almost spent when its performance starts to degrade.

The MDP approach starts to outperform the greedy approach near $t = 40$, for the same reason explained in the above paragraph.

When $\tau$ is set to $200$ as shown on the bottom graph of Figure 1 the MDP agent spent all the budget of the campaigns near $t = 185$, outperforming both greedy and the probabilistic approaches, even though all methods are getting almost the same revenue in the end, for this $\tau$.

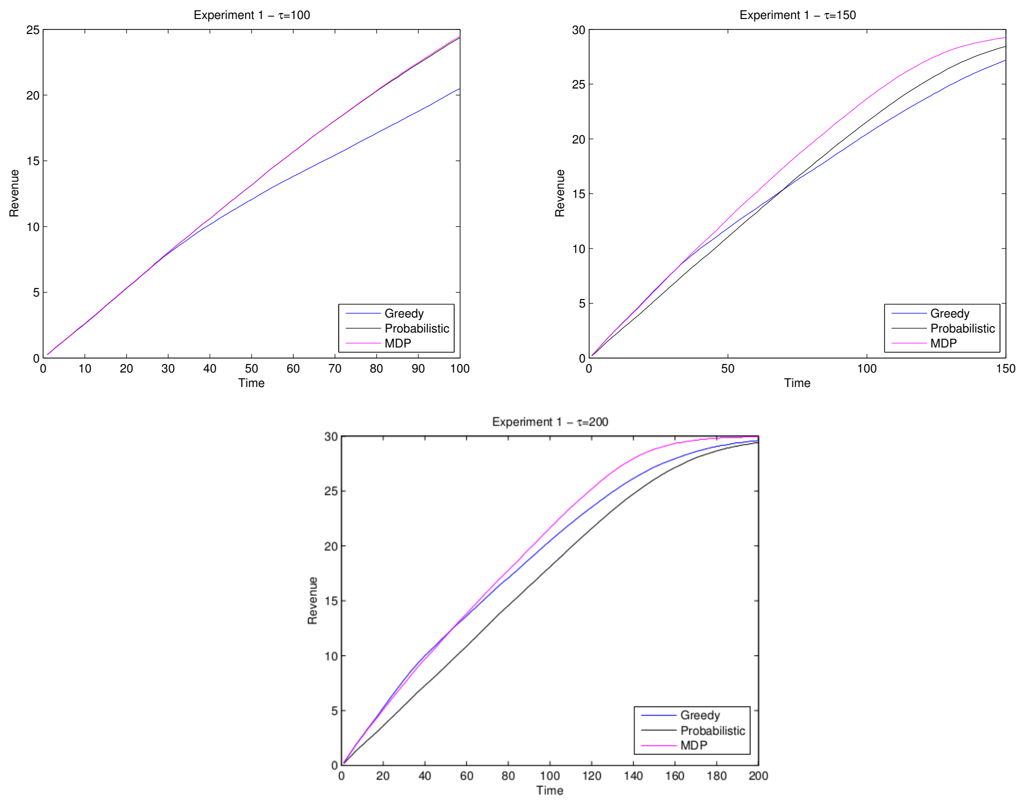

**Figure 1. Revenue over time. Top left: the simulation runs until $\tau = 100$. Top right: the simulations runs until $\tau = 150$. Bottom: the simulation runs until $\tau = 200$.**

## 6.2. Experiment 2

The objective of this experiment is to experience a little more realistic conditions. In the first experiment the probabilities have been chosen so as to force the greedy method to first exhaust campaigns with higher eCPI over the choice of a narrower allocation for a CTR group-campaign a bit more restricted, which is usually harder to be satisfied.

In this experiments the following parameters were set: $|G| = 3$, $|C| = 3$ and $B(C_i) = 10, i \in \mathcal{C}$; $P_{CTR}$ is defined in table 3, $P_G$ is defined in table 4, $P_{request} = 1$, and $\tau$ ranging from $100$ to $300$. The values of $P_{CTR}$ were chosen at random and with low values to simulate a more realistic conditions.

**Table 3. CTR values**

| | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|
| $G_1$ | 0.08 | 0.02 | .20 |
| $G_2$ | 0.16 | 0.10 | .04 |
| $G_3$ | 0.09 | 0.09 | .05 |

**Table 4. $P_G$ values**

| Group | $P_G$ |
|---|---|
| $G_1$ | .35 |
| $G_2$ | .20 |
| $G_3$ | .45 |

In this experiment for $\tau = 100$ the three methods perform almost equally as shown in the top-left graph of Figure 2. This effect came from the fact that there is still campaigns with high probability of clicks in the requests stream with not all budget spent.

When $\tau$ is set to $200$ as shown in the top-right graph of the same Figure, both probabilistic and MDP aproaches outperform the greedy method, because it starts to decrease its slope near $t = 120$. This effect can be explained because the greedy approach burns out the higher eCPI campaign faster, not taking advantage of requests from groups that have a more specific CTR for some campaign.

In the bottom graph of Figure 2, when $\tau = 300$ almost all budget is spent, and the probabilistic and greedy methods perform nearly the same. However, we can see that the MDP approach is still better than the other two approaches.

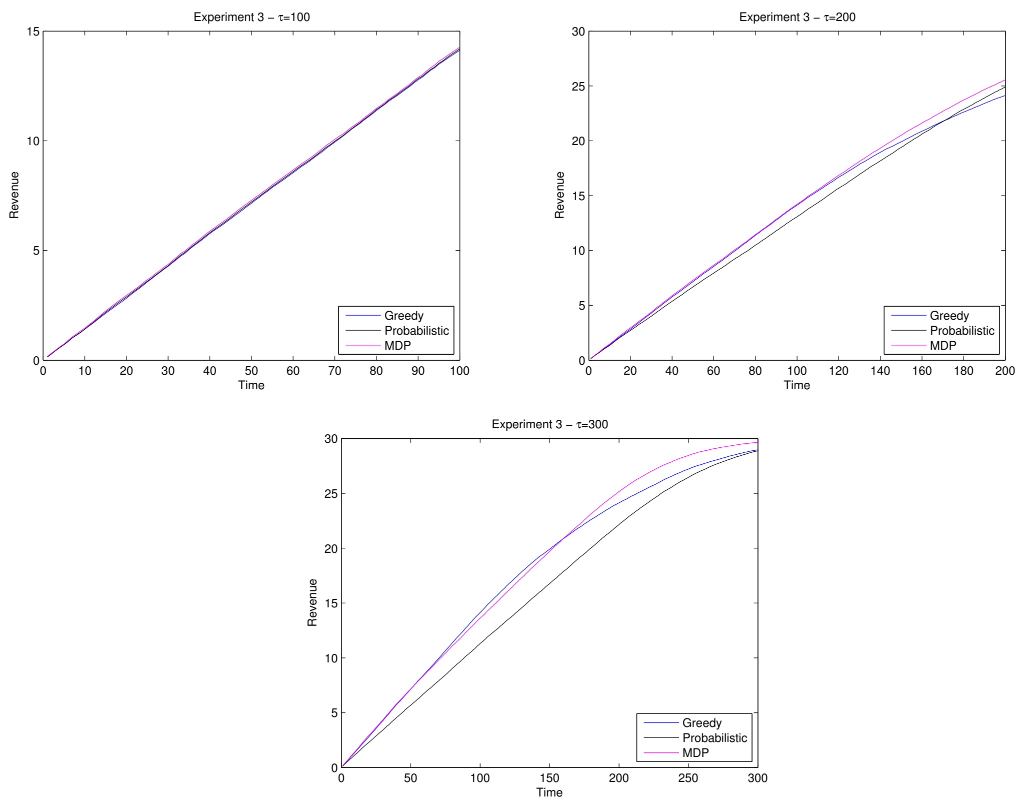

**Figure 2. Revenue over time. Top left: the simulation runs until $\tau = 100$. Top right: the simulations runs until $\tau = 200$. Bottom: the simulation runs until $\tau = 300$.**

## 7. Conclusion and Discussion

In this paper we proposed a new method to ad allocation subjected to budget constraints as an alternative to solve the ad allocation problem using planning techniques in an MDP framework.

We believe that this approach has a major improvement over the state of art solutions because the MDP can capture the dynamics of the problem that other methods cannot tackle. In our experiments neither the greedy nor the probabilistic methods could outperform our method, and the experiments indicate that modeling as MDPs is better than other approaches, even though there is still many scenarios and other methods to be investigated.

Although our proposal can overcome the other methods, the MDP has a serious flaw: the state space is usually huge, even for small scenarios, making the transition probability function sometimes intractable, even using sparse matrices to represent it. However, we believe that the hugeness of the state space is an opportunity to explore new formalisms, alternative techniques and best representations of states.

One way to tackle the hugeness of the state space is to use a factored representation of the MDP [Boutilier et al. 1999, Guestrin et al. 2003], which can represent large structured MDPs compactly. In this representation, the state is implicitly described by using state variables and the transition model is made using a dynamic Bayesian network, allowing an exponential reduction in the representation size of the MDPs. Moreover, there are algorithms that can automatically discover and exploit the hidden structure of factored MDPs [Kolobov et al. 2012] in order to solve MDPs faster and with less memory.

The next step of this work is to start adding more constraints in the problem, such as: minimum number of impression per campaign, category restrictions and a better reward function that is not only based on revenue in favor of creating a more homogeneous distribution of ads from different campaigns.

The presented simulator may have its rules relaxed to better capture the dynamics of the system being modeled. One may want that the system could handle receiving multiple requests from different groups. In order to do this the $P_{request}$ probability should be dropped, and the $P_G$ changed to $\lambda$ of a Poisson distribution [Haight 1967], and the events of requests would be drawn from such distributions. This change would modify the $I$ vector as follows, $I = [I_1, I_2, \ldots, I_{|G|}]$, where $I_i = X$ if the group $i$ has issued $X$ requests, and any group $i \in \mathcal{G}$ would be able to generate requests in the same simulation time.

The planning solution that its proposed in this paper used a fixed CTR to create the plan, as did the other methods in the Literature, despite the fact that in real world the CTR is always changing. Then it is natural to investigate the possibility of adding reinforcement learning techniques [Sutton and Barto 1998] with the purpose of exploring more realistic scenarios and increasing the performance when the CTR is changing. We believe that the fusion of real time CTR estimation and MDP planning working together is the holy grail to solve the problem of ad allocation.

There is many opened research opportunities in this area, motivated not only because its intrinsic business application, but also for the sake of the improvement that these researches can make on planning, MDP, reinforcement learning and other branches of artificial intelligence.

## Acknowledgements

Anna Helena Reali Costa and Fábio Gagliardi Cozman are partially supported by CNPq. Flávio Sales Truzzi is supported by CAPES. The work reported here has received substantial support through FAPESP grant 2008/03995-5 and FAPESP grant 2011/19280-8.

## References

Boutilier, C., Dearden, R., and Goldszmidt, M. (1999). Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121:2000.

Chen, Y., Berkhin, P., Anderson, B., and Devanur, N. R. (2011). Real-time bidding algorithms for performance-based display ad allocation. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 1307–1315, New York, NY, USA. ACM.

Chickering, D. M. and Heckerman, D. (2000). Targeted advertising with inventory management. In *Proceedings of the 2nd ACM conference on Electronic commerce*, EC '00, pages 145–149, New York, NY, USA. ACM.

Devanur, N. R. and Kakade, S. M. (2009). The price of truthfulness for pay-per-click auctions. *Proceedings of the tenth ACM conference on Electronic commerce - EC '09*, page 99.

Guestrin, C., Koller, D., Parr, R., and Venkataraman, S. (2003). Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, 19:399–468.

Haight, F. A. (1967). *Handbook of the Poisson distribution [by] Frank A. Haight*. Wiley New York,.

Kiang, M. Y., Raghu, T. S., and Shang, K. H.-m. (2000). Marketing on the Internet — who can benefit from an online marketing approach ? *Decision Support Systems*.

Kolobov, A., Mausam, and Weld, D. S. (2012). Discovering hidden structure in factored MDPs. *Artificial Intelligence*, 189:19–47.

Muthukrishnan, S. (2009). Ad Exchanges : Research Issues. pages 1–12.

Nagurney, A. (2010). An Integrated Framework for the Design of Optimal Web Banners. pages 1–20.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience.

Pwc, U. S. (2012). IAB Internet Advertising Revenue Report - Full Year 2011. (April).

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1st edition.