Click Shaping to Optimize Multiple Objectives

Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, Xuanhui Wang
Yahoo! Labs
Sunnyvale, CA
{dagarwal,beechun,elango,xhwang}@yahoo-inc.com

ABSTRACT

Recommending interesting content to engage users is important for web portals (e.g. AOL, MSN, Yahoo!, and many others). Existing approaches typically recommend articles to optimize for a single objective, i.e., number of clicks. However a click is only the starting point of a user's journey and subsequent downstream utilities such as time-spent and revenue are important. In this paper, we call the problem of recommending links to jointly optimize for clicks and post-click downstream utilities *click shaping*. We propose a multi-objective programming approach in which multiple objectives are modeled in a constrained optimization framework. Such a formulation can naturally incorporate various application-driven requirements. We study several variants that model different requirements as constraints and discuss some of the subtleties involved. We conduct our experiments on a large dataset from a real system by using a newly proposed unbiased evaluation methodology [17]. Through extensive experiments we quantify the tradeoff between different objectives under various constraints. Our experimental results show interesting characteristics of different formulations and our findings may provide valuable guidance to the design of recommendation engines for web portals.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services

General Terms

Algorithms, Design

Keywords

Multi-objective, click shaping, constrained optimization

1. INTRODUCTION

Searching and browsing are two dominant ways to access information on the Web. While searching to obtain specific content of interest is ubiquitous, a significant amount of time is also spent on browsing the web for content items that are trendy, interesting and eclectic. Examples include visits to a sports site like ESPN to find the latest game updates, browsing the Yahoo! Front Page to read interesting articles, and so on. Hence, it is important to recommend interesting content items that delight and engage users.

Online digital content is easy to create and access. This provides a rich inventory of widely varying quality on the Web, and underscores the need to develop algorithmic methods that can prune low quality content in order to surface the engaging ones. However, such an approach depends on the metrics that we are interested in optimizing for. Prior work typically approach this problem by considering a single metric – the click through rate (CTR) on the recommended content (e.g., [3, 8] and references therein). While CTR is a reasonable proxy for content popularity and simple to measure, such an approach ignores nuances that are involved in measuring other metrics that are often of interest to website owners. For instance, time-spent on the content page is an important engagement metric reported by comScore[1], it is known to be indicative of users' satisfaction in other scenarios such as search [18, 6]. Furthermore, website owners may also consider non-engagement metrics like advertising revenue on the content page. Often, these metrics are in conflict and optimizing for one may deteriorate some others. For example, increasing CTR at the expense of total time-spent is perhaps detrimental and indicative of inferior user experience. Thus, recommending items to obtain an appropriate tradeoff among several metrics is an important research problem.

Consider the Today Module on Yahoo! Front Page in Figure 1 as a concrete example. Articles are selected from a live content pool for display on the four module positions. Of the four positions, the F1 position has the maximum exposure. An article in the content pool links to a page on some other Yahoo! property like News, Finance, and Sports. Thus Yahoo! Front Page is like a distribution channel that routes users to various other Yahoo! pages. Hence a click on the Today Module is associated with downstream metrics like time-spent and revenue that accrues post-click. Although optimizing total clicks on Today Module can boost overall *supply* to different Yahoo! pages, in the presence of significant inter-property variance in downstream metrics, optimizing for clicks alone may lead to an undesirable supply distribution. For instance, maximizing for total clicks may divert more user visits to a low revenue entertainment site at the expense of a higher revenue finance site. It is perhaps desirable to change the content serving scheme to obtain a larger fraction of clicks on finance articles without losing too many clicks overall. This is an example of optimizing for multiple objectives when recommending articles (items) to users. Intuitively when we optimize for different objectives, the clicks on the Today Module that generate visits to different Yahoo! properties

[1]http://www.comscore.com/

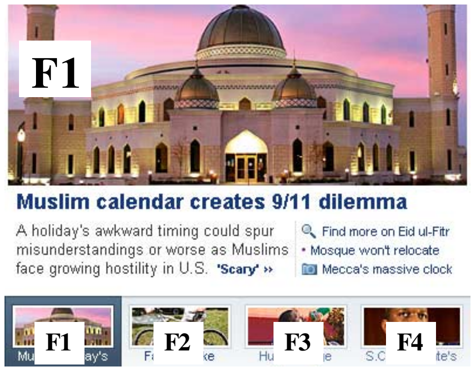
F1
Muslim calendar creates 9/11 dilemma
A holiday's awkward timing could spur misunderstandings or worse as Muslims face growing hostility in U.S. 'Scary' »
Find more on Eid ul-Fitr
• Mosque won't relocate
Mecca's massive clock
F1
F2
F3
F4

Figure 1: Today Module on Yahoo! Front Page

are *shaped* by the recommendation system. Hence we refer to this problem of recommending items to users to jointly optimize for clicks and post-click downstream utilities *click shaping*.

In this paper we propose a multi-objective programming (MOP) approach [20] for click shaping. The goal of such click shaping in our application scenarios is to obtain significant improvements in some downstream metrics (e.g., revenue or time-spent) by forgoing a small fraction of total click volume that is obtained through the status quo scheme that maximizes for CTR alone. We formulate the problem in a constrained optimization framework. Such a formulation can naturally incorporate various application-driven business requirements. A range of solutions that compute the amount of click loss incurred to barter a certain improvement in downstream metrics is given by the multi-objective programming approach. The solution embraced by the website owner depends on the application scenario and business strategy. We believe that quantifying such tradeoffs among multiple competing objectives provide non-trivial insights to the design of recommendation engines and can help with better decision-making. Specifically we make the following contributions in this paper:

- We propose *click shaping*, a novel twist to content recommendation that incorporates both click-throughs and other downstream metrics in a multi-objective programming (MOP) framework. In particular, we formulate MOPs of different flavors that tradeoff total clicks with downstream engagement metrics like time-spent. We show that for recommending content on large web portal pages that distribute traffic to several downstream pages, click shaping provides opportunities to effectively tradeoff multiple downstream metrics.

- We introduce variants that ensure "fairness" when routing visits to different properties (domains). Our fairness objective ensures that the gains in engagement are shared in a fair way among different domains. Such constraints are important in a portal setting where the downstream properties are heterogeneous. Furthermore, our fairness objective also helps in normalizing for scale effects whereby engagement metrics may have sharp differences in scale across different properties. Imposing such constraints presents additional computational challenges. We show several of our constrained MOPs can be efficiently solved through convex optimization techniques [21].

- Through replay experiments [17] on logs obtained via a randomized serving scheme on the Today Module, we provide unbiased performance comparison for different MOPs. Our data analysis is conducted with two metrics – clicks and time-spent. Solutions obtained for MOPs under different constraints provide interesting insights. We show that significant gains in downstream engagement is possible by losing small fraction of clicks. We also show the behavior of click shaping in the presence of fairness objective. We provide interesting insights obtained through extensive data analysis.

2. PROBLEM SETUP

For the sake of concreteness, we define our problem in the setting of the Today Module on Yahoo! Front Page in Figure 1. Our methodology can be easily applied to other multi-objective recommendation settings. Today Module provides users with interesting, timely, and informative articles every day. At each time point, there is a candidate pool of live articles created through editorial oversight, the pool is frequently refreshed to keep up with timely and trendy articles. For every visit to the Yahoo! Front Page, articles selected from the live content pool are recommended on the Today Module with the goal of optimizing some objective(s).

Notations: Formally, at each epoch t, the set of live articles is denoted by $\mathcal{A}_t = (A_1, \cdots, A_{n_t})$. Each article $j \in \mathcal{A}_t$ belongs to one of K different Yahoo! properties (e.g., News, Finance, Sports, etc). Supply to a given Yahoo! property is significantly influenced by the links clicked in the Today Module and provides a good setup for our study. Let $\mathcal{P} = \{P_1, ..., P_K\}$ denote the set of properties and $j \in P_k$ means the landing page of article j belongs to property P_k. Users are assumed to be clustered into m segments: $\mathcal{S} = \{S_1, \cdots, S_m\}$, this is an important assumption for methods discussed in this paper. We discuss various user segmentation schemes in Section 5. Let p_{ijt} denote the probability (CTR) that a user in segment i would click article j when it is displayed in epoch t, and let d_{ijt} denote the time that the user would spend on the landing property of article j if she clicks the article. Other downstream utilities (single or multiple) can be easily incorporated in our framework, we focus only on time-spent in this paper for the sake of illustration. Note that if the goal is to optimize for a single objective (maximize clicks or time-spent, but not both), the optimal solution is to recommend the article with the highest p_{ijt} (or $p_{ijt} \cdot d_{ijt}$) to users in segment i.

Explore/exploit: Content optimization is an explore/exploit problem. To optimize for any metric, we need to estimate the performance of each candidate article in terms of that metric. Without displaying an article to any user, it is difficult to know the performance of that article. Models may be built to predict article performance based on article features; however they may be worse than a simple method that directly computes the metric from online data (e.g., computing click-through rate using the clicks and views of the article) [3]. The explore/exploit problem for a single objective is well studied (e.g., [3]). Developing explore/exploit methods for multiple objectives that ensure certain notion of optimality is non-trivial. In this paper we assume some explore/exploit scheme is running in the system. In particular, we use a simple scheme — ϵ-greedy, which has been empirically shown to achieve good performance [24]. This scheme works as follows: We serve a small fraction of randomly selected user visits with articles selected uniformly at random from the current content pool in order to collect data for every article (explore population). For the remaining visits, we serve the article with the highest estimated CTR if the goal is to maximize clicks (exploit population). With multiple objectives, the serving scheme for the exploit population is different from that of displaying the highest CTR article as we shall see later in this paper.

Serving scheme: Given multiple objectives and constraints, our goal is to construct a serving scheme that can be used to serve articles to users in the exploit population. For each epoch t, a serving scheme uses information obtained before the epoch to decide an allocation plan $\mathbf{x}_t = \{x_{ijt} : i \in \mathcal{S}, j \in \mathcal{A}_t\}$, where x_{ijt} is the fraction of visits in user segment i to be served with article j in epoch t. Obviously the fractions are non-negative and lie on the article simplex for each user segment, i.e., $x_{ijt} \geq 0$ and $\sum_j x_{ijt} = 1$. Given $\mathbf{x}_t$, an incoming user in epoch t is assigned to the appropriate user segment and an article is served through a multinomial draw from the article simplex of the segment. Different optimization methods generate different allocation plans according to different criteria. For example, the click maximization method would set x_{ij^*t} to 1 if j^* has the highest estimated CTR among all the articles and 0 for the remaining articles. Soft clustering of users also works with our formulation and in this case the allocation vector of a user is a weighted sum of cluster-specific allocation vectors. User clusters can also change over time, for ease of exposition we assume they are fixed.

Offline evaluation: The best way to evaluate a content optimization method is to run controlled live experiments on small fractions of randomly selected user visits on the Yahoo! Front Page. However, such a method is expensive and not feasible in practice for a battery of schemes that one may want to test. Hence, we take recourse to an offline evaluation methodology that works on logged data collected retrospectively [17, 14]. This offline evaluation method can give us *unbiased* and *replicable* comparison. We provide a brief description of the scheme below.

The logged data must be collected from a serving scheme that selects each article from the pool with some *known, non-zero* probability. In our case, we collect view and click events from an experiment where articles are displayed uniformly at random to a small randomly selected user population. We shall refer to this as the *random bucket*. The selection probability of each article is $1/n_t$ at epoch t; this avoids the need to estimate selection probabilities from a non-randomized scheme and also ensures each article is shown a large number of times.

Each view event in our logged data consists of the user segment ID and article ID displayed at F1. For each click event, in addition to user segment ID and article ID, we also record the time-spent (or any other downstream metric of interest) on the landing property. All the events are ordered temporally. The replay evaluation works as follows. It goes through each epoch sequentially. At epoch t, we perform the following steps:

1. Compute the expected CTR and time-spent for each (user segment i, article j) pair. The estimates are computed using all data before epoch t.

2. Compute the allocation plan $\mathbf{x}_t$ for the multiple objective optimization (MOP) under consideration.

3. For each event in t, if the user segment = i, we draw an article j^* from the current pool according to probability x_{ijt}. If the article served in the logged data matches j^*, we record this matched event; otherwise we ignore it.

At the end of this process, we compute CTR and time-spent metrics based on the recorded events and it can be shown these estimates are unbiased [17, 14]. Note that all the matched events for a serving scheme give us a sample of user activities for that scheme. Different serving schemes have different sets of matched events. However, because each article in the random bucket has an equal probability to be displayed to users, the number of matched view events for any serving scheme is expected to be the same. A better serving scheme to optimize CTR can match more click events. We can thus compute the overall CTR and time-spent from these matched events and use these metrics to compare different serving schemes. As emphasized before, under mild assumptions such an evaluation provides an unbiased estimate of performance with small variance when we use large amounts of data.

3. OBJECTIVE DEFINITIONS

In this section we introduce various objectives to be optimized in our multi-objective programming (MOPs). We only consider content serving on a single position (the F1 position on Today Module); optimizing for multiple positions is more involved and part of future work.

We consider two different objectives – (a) total clicks and (b) total time-spent on landing properties. These objectives are standard engagement metrics and commonly reported by companies like comScore; they play a crucial role in forming advertiser perception and measuring website performance. Also from a user satisfaction perspective, it is important to encourage downstream engagement with content in addition to clicks. For instance, the presentation of a link (captions, images, etc) may sometimes boost click-rates without generating high engagement; our MOP approach helps avoid such artifacts by simultaneously optimizing for both downstream engagement and CTR. We work with the two metrics mentioned above to illustrate the main ideas on a real world application.

Let N_t denote the total number of visits during epoch t. Also let $\boldsymbol{\pi}_t = (\pi_{1t}, \cdots, \pi_{mt})$ denote the fraction of visits in different user segments. Obviously, $\sum_{i \in \mathcal{S}} \pi_{it} = 1$ and $N_t \pi_{it}$ is the total number of visits to segment i. In a MOP, $\boldsymbol{\pi}_t$ can be estimated easily based on user visits in past epochs. We now describe the objectives below. Since we solve MOP separately for each epoch, we drop suffix t from the notations. For example, $\mathbf{x} = \{x_{ij}\}$ is the allocation plan for the current epoch.

- **Total clicks**: Note that $N\pi_i x_{ij}$ denotes the number of displays of article j in segment i; thus the total number of clicks in the epoch is

$$\mathit{TotalClicks}(\mathbf{x}) = N \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{A}} \pi_i x_{ij} p_{ij} \qquad (1)$$

- **Total time-spent**: Recall that d_{ij} is the mean of the time-spent by a user in segment i on the landing property of article j after a click; hence the total time-spent is

$$\mathit{TotalTime}(\mathbf{x}) = N \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{A}} \pi_i x_{ij} p_{ij} d_{ij} \qquad (2)$$

Before defining our MOPs, we first consider the **click optimization scheme** with the serving algorithm that solely maximizes total clicks (overall CTR); this algorithm serves as the baseline that our MOPs will be compared to. Let $\{z_{ij}\}$ denote a serving scheme that maximizes only for CTR. Then,

$$z_{ij} = \begin{cases} 1 & \text{when } j = \arg\max_J p_{iJ} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

We use *TotalClicks** and *TotalTime** to denote the value of objectives for the click maximization scheme; they are constants in our MOPs.

4. OPTIMIZATION MODELS

In this section we discuss different ways of formulating multi-objective optimization problems. In general multi-objective problems can be solved by a number of multi-criteria programming techniques [20]. These include solving a single objective that is a weighted average of multiple objectives or using goal programming where the objectives are optimized sequentially with constraint(s) that previous objectives retain a certain fraction of their optimal value [12].

4.1 Scalarization

Assume *TotalClicks* and *Downstream* are the two objectives of interest where *Downstream* can be *TotalTime* or other downstream metrics of interest. To construct the allocation plan, we find the $\mathbf{x}$ that maximizes

$$\lambda \cdot \textit{TotalClicks}(\mathbf{x}) + (1-\lambda) \cdot \textit{Downstream}(\mathbf{x}),$$

where $\lambda \in [0,1]$ represents the tradeoff between total clicks and downstream engagement; with a smaller λ we lose more clicks and obtain better downstream engagement. It is easy to see that the solution is given by

$$x_{ij} = \begin{cases} 1, & \text{if } j = \arg\max_J \lambda \cdot p_{iJ} + (1-\lambda) \cdot p_{iJ} d_{iJ} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The tradeoff between objectives for a constant λ may vary significantly across epochs. In fact in some epochs the loss in clicks could be significant, this maybe a problem in some application scenario. However, the ability to lose a significant number of clicks in some epochs to obtain substantial gains in engagement may lead to better results over a long time horizon. We shall refer to this scheme as a scalarized multi-objective program, or **s-MOP**. Such an approach is attractive when the website owner is interested in a weighted combination of multiple objectives and it does not hurt if any one of them deteriorates significantly. Since the objectives are both linear, one drawback of this approach is the inability to explore all possible points on the Pareto optimal curve ([21] Chapter 7), it may miss out some interesting solutions. Another drawback of this method is that it is not easy to introduce application-driven business constraints.

4.2 Constrained Optimization

We now describe the constrained optimization approach to our multi-objective problem. This is important when website owners are interested in optimizing for multiple objectives but under certain constraints. This is the case in our problem. For instance, too much variation in click loss changes the downstream supply and could have an adverse impact on advertising. It is also not desirable to get overall lifts in downstream metrics by significantly changing the supply to different properties from the status quo. Hence, constrained optimization provides a reasonable approach in our application scenario.

4.2.1 g-MOP: Global Constrained Optimization

We begin with the simplest version of MOP that involves constrained optimization of objectives at the global level. We will call them global multi-objective programs or **g-MOP**s since they involve objectives that are "global". For our two-objective problem, we obtain the design variable $\mathbf{x}$ as the solution to the following optimization problem. We note that $\{x_{ij}\}$s are always non-negative and on the simplex for each i; thus, we do not mention it explicitly in describing our MOPs.

$$\begin{aligned} &\text{maximize } \textit{Downstream}(\mathbf{x}) \\ &\quad\text{s.t. } \textit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \textit{TotalClicks}^* \end{aligned} \tag{5}$$

where $\alpha \in [0,1]$, and *Downstream* in this paper is *TotalTime*. Here we want to obtain maximum improvement in downstream engagement ensuring the percentage loss in total click volume in no more than $(1-\alpha)$. Since maximizing total clicks is the normative serving scheme in general, it is reasonable for website owners to control this loss. We also note that solutions to g-MOPs will show lower variation in number of clicks lost per epoch relative to s-MOP described earlier. In fact, one can interpret the constraint imposed per epoch in g-MOP as inducing a different tradeoff parameter λ_t in s-MOP for epoch t. Hence constrained g-MOPs can find solutions on the Pareto curve that are missed by s-MOP. With *Downstream* $=$ *TotalTime* in Equation 5, it is easy to see that the optimal $\mathbf{x}$ to this problem can be obtained by a standard linear program solver.

4.2.2 ℓ-MOP: Localized Constrained Optimization

As mentioned before, the g-MOP formulation is reasonable when landing pages in different properties are homogeneous. This is hardly true in practice; the layout of the landing page on a property may have a significant influence on the distribution of d_{ij}s. For instance, suppose we have two pages — an entertainment page with several video links and a finance stock ticker page. The distribution of time-spent for the video page is stochastically larger than the finance page. Maximizing downstream engagement may simply take away clicks from finance and route it to video page. In fact the situation may get worse — we take away highly engaged users on finance and route them to video where they engage less than a typical video page user. To avoid such effects, we solve g-MOPs under additional constraints. We refer to these as **ℓ-MOP**s and not surprisingly, the ℓ-MOPs come in various flavors depending on the objective being maximized and the constraints being imposed. We motivate the constraints that define the ℓ-MOPs below.

Let us assume the articles have been grouped based on landing page characteristics. We use the property grouping $\{P_1, \cdots, P_K\}$ described earlier to illustrate in this paper; some other grouping may be important in other applications. Our first formulation to address the inter-property heterogeneity maximizes the minimum value of total engagement across properties subject to constraint on click loss. By maximizing the minimum, we ensure "fairness" since all properties receive the benefit of losing clicks on the portal page. Before defining our constrained optimization problems, we introduce some new notations.

Let *Obj*$(P, \mathbf{x})$ denote the value of the objective restricted to property P, and let *Obj*$^*(P)$ denote the objective value on property P under the click maximization scheme where *Obj* can be *TotalClicks*, *TotalTime*, or some other objective of interest. To normalize engagement metrics across properties and create a single "currency", we measure engagement improvement for a property P_k by the ratio $\frac{\textit{Obj}(P_k, \mathbf{x})}{\textit{Obj}^*(P_k)}$. We are now ready to mathematically state our ℓ-MOP to ensure "fairness".

MaxMin ℓ-MOP:

$$\begin{aligned} &\text{maximize}_{\mathbf{x}} \ \min_k \Big\{ \frac{\textit{Downstream}(P_k, \mathbf{x})}{\textit{Downstream}^*(P_k)} \Big\} \\ &\quad\text{s.t. } \textit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \textit{TotalClicks}^* \end{aligned} \tag{6}$$

When *Downstream* $=$ *TotalTime*, the objective function is concave and hence can be solved by convex programming techniques. However it is not differentiable, so it is difficult to solve it directly. We use a standard trick in convex programming that transforms the problem to the epigraph form [21]. The epigraph for this program is given by

$$\begin{aligned} &\text{maximize } \beta \\ &\text{s.t. } \frac{\textit{Downstream}(P_k, \mathbf{x})}{\textit{Downstream}^*(P_k)} \geq \beta, k = 1, ..., K \\ &\quad \textit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \textit{TotalClicks}^* \end{aligned}$$

It is easy to see that the above is a linear program.

Relaxed ℓ-MOP: Although the MaxMin formulation encourages the benefits of click loss to be shared among properties, the socialistic nature may impose constraints that are too strict. In fact, it may not be possible to get significant gains in downstream engagement under such strict constraints in some applications. Furthermore, portal owners may not care to improve engagement metrics of some properties; only a handful may be more important than others. They may be satisfied with increasing total engagement by ensuring engagement of important properties does not deteriorate compared to the status quo. We can encourage such behavior by introducing property level constraints but still maximizing aggregate engagement. The number of properties subject to such constraints can also vary. Formally our optimization problem is described below.

$$\begin{aligned} &\text{maximize}_{\mathbf{x}}\ \textit{Downstream}_1(\mathbf{x}) \\ &\quad \text{s.t. } \textit{Downstream}_2(P_k, \mathbf{x}) \geq \textit{Downstream}_2^*(P_k), k \in \mathcal{I} \\ &\quad\quad \textit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \textit{TotalClicks}^* \end{aligned} \tag{7}$$

where *Downstream*$_1$ and *Downstream*$_2$ can be the same or different (in this paper they are both *TotalTime*), and $\mathcal{I}$ is the set of properties that are constrained. When this set is empty we fall back to g-MOP and when it includes all properties, we have tight constraints but these are still more relaxed than MaxMin ℓ-MOP. Instead of constraining the properties to improve, we can also relax and provide a lower bound to the maximum possible deterioration (it is easy to incorporate such constraints). The solution techniques for this class of problem again depends on the combination of downstream engagement objective *Downstream*$_1$ and downstream engagement constraint *Downstream*$_2$. For total time-spent constraints, all variants of constrained ℓ-MOPs can be solved using linear programming!

5. STATISTICAL MODELS

In previous sections, we assume that π_{it}, p_{ijt} and d_{ijt} are given. In our application, we solve MOPs by "plugging-in" the expected values estimated from a statistical model. Note that our formulation requires clustering the users into segments, hence we focus on segmented CTR (and time-spent) model in this paper. Since our optimization problem involves constraining click-loss, we create user segments that are most predictive for estimating click-rates and use these to estimate time-spent. Gains are significant if there is wide variation in average time-spent across articles for a given user segment.

5.1 User Segmentation

Each user is characterized by a feature vector consisting of demographics (age and gender) and affinity to different behavioral categories (such as sports, finance or entertainment) based on cookie activities throughout the Yahoo! network. In this paper, we illustrate with user segments constructed using a few strategies. We consider three segmentation schemes:

- Age-gender: segments are obtained based on user age and gender.
- Attribute-based: unsupervised k-means clustering on the attribute vectors based on age, gender and behavioral affinities.
- Activity-based: unsupervised k-means that cluster users based on their estimated click activities on different articles.

For age-gender segmentation, we discretize age into 10 groups and have 3 gender groups (male, female, and unknown), and thus we have 30 segments. Obtaining clusters using attribute-based k-means is straightforward. For the activity-based k-means, we collected large amounts of historic click data. A simple approach is to construct a vector for each user based on the articles the user clicked. However, this does not work so well since the distribution of user visits is heavy tailed - a small number of users have many clicks and a large fraction of users have few clicks. This imbalance makes it difficult to construct user profiles directly from clicks. We build a separate logistic regression model for each article based on user attributes and use the *predicted* CTR values on each article to compose the activity vectors. Specifically, let $\mathbf{a}_u$ denotes attribute vector for user u, then the click probability on article j for user u is $\theta_{uj} = \frac{1}{1+\exp(-\mathbf{a}_u \cdot \eta_j)}$, where η_j for article j is estimated through a logistic regression for article j based on all the users who clicked (positive examples) or viewed j without clicks (negative examples). This gives us an activity-based profile for user u based on θ_{uj} for a set of articles in the historic data. We then cluster the users into segments by applying k-means on these profiles. A similar approach was used in [7] with article categories instead of articles which implicitly assumed a constant affinity for a given user for all articles in a given category. We found wide variations in user affinities for articles within a category, hence the profiles used in this paper are at the article level.

For online serving, each user will be classified to a user segments based on their corresponding article profile vectors. We note that creating such a profile for historic articles that are no longer in the pool is not a problem for us since we can always predict θ_{uj} for any user on any article j (old or new). For all k-means based clustering, a centroid is kept for each cluster and a user is assigned to the nearest one. In this paper, we use the cosine similarity and the k-means algorithm proposed in [10, 9]

5.2 Estimation of π_{it}, p_{ijt} and d_{ijt}

Given user segments, both user population π_{it} and time-spent d_{ijt} are estimated based on moving average over historical data. For example, given all the epochs before t, we collect all click events for article j from user segment i. Each click event has a time-spent value and d_{ijt} is estimated as the average time-spent over all the clicks.

For CTR estimation, recall that p_{ijt} denotes the CTR of user segment i on article j at epoch t. In [2], we showed that the Gamma-Poisson model performs well for estimating CTR on such segmented models. Let n_{ijt} denote the number of times we show article j to user segment i at time t. Let c_{ijt} denote the number of clicks that resulted from these n_{ijt} page views. For simplicity, assume CTR does not change much over time in a user segment; thus we drop index t in p_{ijt} (for the extension that captures temporal dynamics, see [2]). The Gamma-Poisson model assumes

$$\begin{aligned} c_{ijt} &\sim \text{Poisson}(p_{ij}\, n_{ijt}) \\ p_{ij} &\sim \text{Gamma}(\text{mean=}\mu_i, \text{size=}\gamma_i), \end{aligned}$$

where μ_i is the CTR of user segment i on a random article according to our prior belief, and γ_i is the equivalent sample size of the prior belief (intuitively, this prior belief represents information equivalent to γ_i page views and $\gamma_i \mu_i$ clicks). Effectively, the Gamma-Poisson model is a "smoothed" counting model which estimates p_{ijt} as

$$\hat{p}_{ijt} = \left(\gamma_i \mu_i + \sum_{\tau<t} c_{ij\tau}\right) / \left(\gamma_i + \sum_{\tau<t} n_{ij\tau}\right).$$

If we have little data the estimate will be close to the prior μ_i; with more data the estimate will be close to maximum likelihood.

6. EXPERIMENTS

In this section we report experimental results on Yahoo! Front Page Today Module data.

Data: Our data set is derived from Yahoo! web server logs which record the view and click information of users who interact with the Today Module on the Yahoo! Front Page. We collected such click and view data from a random bucket (defined in Section 2) of the Today Module during August 2010. Around 2 million view events were collected on average on a daily basis. To compute downstream time-spent, we also collected post-click information on all the pages that a user visited within Yahoo! after clicking on a Today Module article.[2] We use the first $1/3$ of the data to learn the user segmentation. The remaining $2/3$ of the data is used to compare different algorithms based the replay methodology described in Section 2 that provides unbiased estimates.

Metrics: We define the time-spent d_{ij} of a user in segment i after clicking article $j \in P_k$ as the length (in second) of the *session* of the user's events that starts from the click and ends at the last page view inside property P_k, before the user either leaves the property or has no activity for more than 30 minutes. For confidentiality reasons, we cannot reveal the total number of clicks or total time-spent. Thus, we only report the relative CTR and relative time-spent as defined below. After running a replay experiment using serving scheme A, we compute the average number of clicks per view p_{A} (i.e., CTR) and the average time-spent per view q_{A}. Fixing one baseline algorithm B, we report the performance of algorithm A by two ratios: CTR ratio $\rho_{\text{CTR}} = \frac{p_{\mathsf{A}}}{p_{\mathsf{B}}}$ and TS ratio $\rho_{\text{TS}} = \frac{q_{\mathsf{A}}}{q_{\mathsf{B}}}$.

6.1 Results on Segmentation

Comparison of segmentation methods: In this set of experiments, we report the performance of different algorithms in terms of CTR and TS ratios by using the click optimization model with age-gender segmentation as the baseline algorithm B. We first compare different segmentation methods using both the s-MOP and g-MOP algorithms in Figure 2. In this figure, the α (click-constraint parameter) values used are {1, 0.99, 0.97, 0.95, 0.93, 0.9, 0.85, 0.8, 0} for g-MOP and we vary λ (tradeoff in scalarization approach) from 1 to 0 with decrements of 0.1. Each specific parameter value gives us a pair of CTR ratio and TS ratio. The tradeoff curves are thus obtained by varying these two parameters respectively for s-MOP and g-MOP. We have 30 segments in age-gender model. For all k-means-based clusters (feature and activity based), we also set the number of clusters m to 30. Each method is represented by a curve; the higher and more to the right the curve is, the better the performance. From this figure we make the following observations:

(1) The activity-based segmentation significantly outperforms the other two methods, while the age-gender segmentation is slightly better than the attribute-based k-means method which clusters users based only on their features. This shows that a carefully designed user segmentation is crucial component of our method. Our activity-based method can segment users much better because it uses click information (one of our objective) to construct clusters. On the contrary, the attribute-based method that uses only user profile features is less effective. The age-gender model though simple, still achieves better results than attribute-based k-means method.

[2]By users we mean an anonymized browser-cookie. There is no personally identifiable information in the data used in our experiments.

Method	CTR ratio		TS ratio	
	mean	stdev	mean	stdev
s-MOP (λ=0.60)	0.9580	0.0316	1.0850	0.0480
g-MOP (α=0.97)	0.9604	0.0242	1.0830	0.0376

Table 1: Temporal variance of s-MOP and g-MOP.

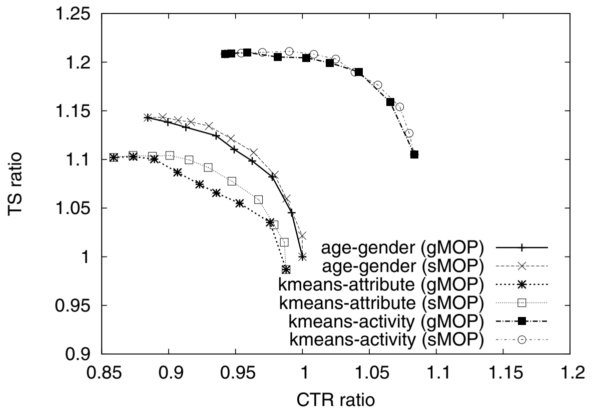

Figure 2: Comparison of user segmentation methods.

(2) The bottom right end point of each curve corresponds to $\alpha = 1$ or $\lambda = 1$, i.e., the performance of the click optimization model. As we decrease the α values, we move up and to the left; the CTR ratio becomes smaller but the TS ratio becomes larger. For example, for the activity-based segmentation, comparing $\alpha = 1$ and $\alpha = 0.97$, we increase the TS ratio from 1.10 to 1.20 (8.1% lift) and decrease the CTR ratio from 1.08 to 1.04 (3.8% drop). This means both s-MOP and g-MOP effectively tradeoff clicks for time-spent. For all the 3 curves, we observe similar trends and the curves become flat towards the end as expected.

(3) We observe that s-MOP achieves better tradeoff than g-MOP, especially with a less effective user segmentation. As described earlier, the α constraint imposed per epoch in g-MOP can be interpreted as introducing a different tradeoff parameter λ_t in s-MOP for each epoch t. The parameter α uniformly bounds the CTR loss for the worst case over all the epochs while λ bounds the CTR loss on average. Thus, given the same CTR loss, we anticipate that the temporal variance in CTR loss and TS gain will be large. To verify this, for each epoch, we compute the CTR ratio of s-MOP and g-MOP: $\frac{p_{\lambda=0.6}}{p_{\lambda=1}}$, $\frac{p_{\alpha=0.97}}{p_{\alpha=1}}$ and their TS ratios $\frac{q_{\lambda=0.6}}{q_{\lambda=1}}$, $\frac{q_{\alpha=0.97}}{q_{\alpha=1}}$ based on the activity-based segmentation. We compute the mean and standard deviation of these four sets of values over all epochs and summarize them in Table 1. From the table, we clearly observe that the variances of both CTR ratio and TS ratio in s-MOP are significantly larger than g-MOP. In some epochs, s-MOP loses more clicks to obtain better gains in time-spent, g-MOP on the other hand is more stable in its tradeoff behavior over epochs. This shows the desirable properties of constrained optimization, if significant deviation from the status quo CTR is not desirable during most time periods, one should take recourse to constrained optimization.

Number of clusters: We study the impact of varying the number of clusters m in the activity-based segmentation scheme on the tradeoffs between clicks and time-spent. In Figure 3, we plot tradeoff curves for different values of m using g-MOP. As evident, choosing number of clusters has a significant impact on performance. If m is small, the clusters might get coarse and lead to biased CTR and time-spent estimates for items within clusters. Increasing the

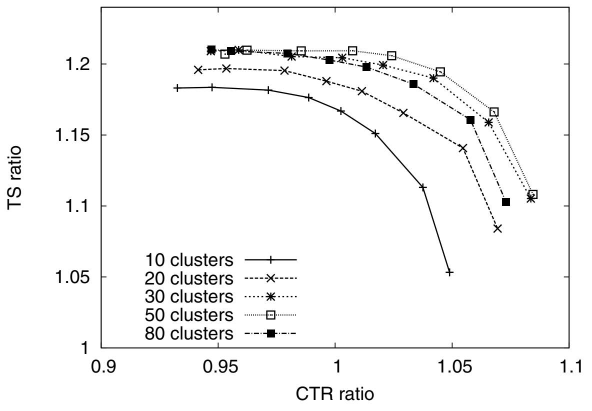

Figure 3: The impact of the number of user segments using activity-based k-means.

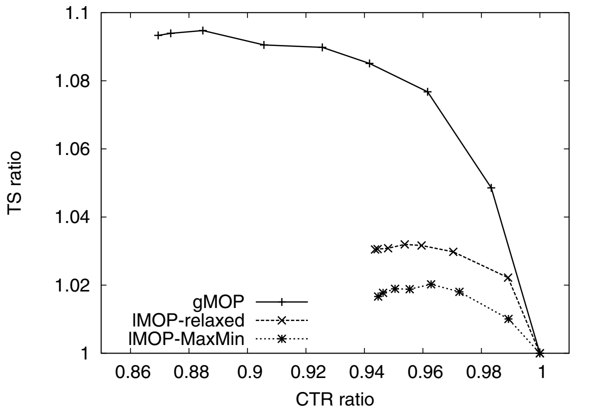

Figure 4: Comparison of different constrained optimization formulation.

number of clusters leads to high variance and deteriorates performance. As can be seen from Figure 3, in our experiments, the optimal m is between 30 and 50.

6.2 Results on Different Formulations

In this section, we compare three different formulations: g-MOP, ℓ-MOP-relaxed, and ℓ-MOP-MaxMin. For segmentation, we report results only from activity-based k-means with $m = 30$ in the following experiments. The CTR ratio and TS ratio are computed using the click optimization model with the activity-based segmentation as the baseline B.

Tradeoff comparison: We first compare the tradeoff curves of the three formulations in Figure 4. In this figure, we can see that g-MOP improves time-spent significantly with a small loss in CTR. However, when we impose the per-property constraints in ℓ-MOP-relaxed, the time-spent gains become much smaller. ℓ-MOP-MaxMin has the least time-spent gain among the 3 formulations. For example, when we set $\alpha = 0.97$, the TS ratio of these 3 methods are 1.08, 1.03 and 1.02. This is because after imposing per-property constraints, we have less freedom to move clicks from one property to another even when there is a property that has higher average time-spent. The ℓ-MOP-MaxMin model is more constrained since it aims to improve every property simultaneously.

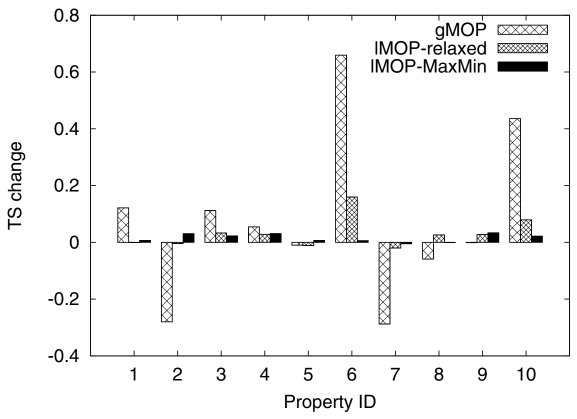

Figure 5: Time-spent difference before and after click shaping given the same amount of CTR loss.

	g-MOP	ℓ-MOP-relaxed	ℓ-MOP-MaxMin
click	0.2750	0.0637	0.0361
time-spent	0.2764	0.0504	0.0134

Table 2: Comparison of the standard deviation over the most popular 10 properties.

Both ℓ-MOP-relaxed and ℓ-MOP-MaxMin impose property-level constraints. We show the impact of these constraints in Table 2. We select the top 10 popular properties, and for each of these properties we compute per-property total click and time-spent relative difference between $\alpha = 0.97$ and $\alpha = 1$ for all the 3 formulations using $\frac{t_{\alpha=0.97}-t_{\alpha=1}}{t_{\alpha=1}}$. We then compute the standard deviations for each formulation across all 10 properties. From this table, we can see that the variance for g-MOP is much higher than ℓ-MOP-relaxed and ℓ-MOP-MaxMin. To further illustrate the variance, we plot the relative difference of time-spent for the 10 properties in Figure 5. From this figure, we can see that the difference across properties for g-MOP vary quite a bit. For example, for a given amount of click loss, some properties can gain more than 60% in time-spent whereas some properties lose around 30% in time spent. On the other hand, ℓ-MOP-relaxed and ℓ-MOP-MaxMin have relatively low variation across properties; no property loses total time-spent significantly. This underlines the effectiveness of including per-property level constraints in our MOPs. The figure also illustrates the fairness of ℓ-MOP-MaxMin - all properties experience similar time-spent gains.

Fewer property constraints: We showed two extreme cases where we either do not have any per-property constraints or we apply the constraints on all properties. In reality, some constraints are more important for certain business purpose. We show the results of selectively applying the constraints on a subset of properties. In Figure 6 and Table 3, we show the results of ℓ-MOP-relaxed with constraints applied to the top 3, 5, 7 properties that have the largest number of views in our data set. Clearly when we reduce the number of constraints, we can achieve much better gains in time-spent with the same loss in clicks. On the other hand as shown in Table 3, the variance in time-spent gain across top 10 properties becomes larger when we enforce fewer constraints.

Example: To understand how clicks are shaped, we compare the time-spent distribution based on ℓ-MOP-relaxed in Figure 7 before and after click shaping on two properties. We discretize the

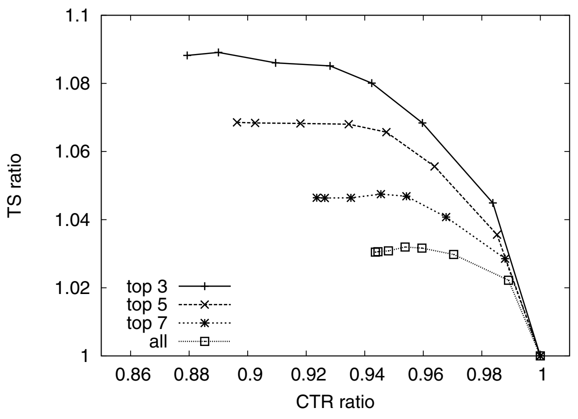

Figure 6: Impact of number of constraints on ℓ-MOP-relaxed.

	Top 3	Top 5	Top 7	All
click	0.2372	0.1792	0.1501	0.0637
time-spent	0.2424	0.1818	0.1506	0.0504

Table 3: Comparison of the standard deviation over properties for ℓ-MOP-relaxed with different number of per-property constraints.

time-spent values into multiple bins and compute the percentage of clicks that fall into each bin. Let $\mathrm{P}_{\alpha=1}(b)$ and $\mathrm{P}_{\alpha=0.97}(b)$ be proportional to the number of clicks in bin b before ($\alpha = 1$) and after ($\alpha = 0.97$) click shaping, we plot the ratio $\frac{\mathrm{P}_{\alpha=0.97}(b)}{\mathrm{P}_{\alpha=1}(b)}$ against the time-spent bins in Figure 7. We also fit a regression line to each property. It can be seen that both lines have positive slope (0.004 for Property B and 0.002 for Property A) and this shows the percentage ratio increases with increase in time-spent. This means that after click shaping, we reduce the number of clicks that result in lower time-spent but increase the number of clicks with more time-spent for each property. This is expected behavior from our formulation and more desirable because more engaged articles (measured by time-spent) are recommended by our algorithms. The fact that this behavior is empirically supported in real world application provides a good validation of our theoretical formulation.

7. RELATED WORK

It is commonplace to measure user interaction with websites through multiple metrics such as number of page views, number of unique user visits, and total time-spent. Such metrics are routinely reported by companies like comScore and have profound impact on advertiser perception. Most of the previous methods on Web content optimization (e.g., [3, 8, 2, 1, 16]) focus on a single metric: number of clicks. The problem is often formulated as a multi-armed bandit problem [5, 15, 4]. However, to the best of our knowledge, no prior work has attempted to provide methods that simultaneously optimize a combination of various engagement metrics.

Some instances of tradeoffs among multiple objectives are discussed in online advertising. For instance, auctions in sponsored search incorporate both revenue (bid) and ad quality (measured through CTR) in ranking ads [19]. The multi-objective programming pursued in this area is simple – rank ads by product of bid and CTR. In fact, this rule is a special case of s-MOP with $\lambda = 0$ in our formulation. Recent works on display advertising also consider multi-objective programming to simultaneously optimize for revenue of remnant inventory and overall ad quality delivered to advertisers participating in the guaranteed display ads marketplace [11]. Another work that is somewhat related incorporates both CTR and ad quality in ranking ads for sponsored search [22]. The authors define a new measure called *bounce rate* to capture ad quality by inferring abandonment rate on ad landing pages. However, the study is more focused on ways to predict bounce rates and multi-objective optimization is not the main focus. Other papers such as [13] considered constraints such as limited supply of the items in a collaborative filtering setting and [23] studied several objectives in learning to rank. Both of these are in a static setting and thus the whole problem setup is different from our online recommendation.

We note that there is a rich and mature literature on multi-objective programming [20]. We are not contributing new techniques to this body of work. Instead, we have shown the utility of such approaches in simultaneously optimizing for several engagement metrics when recommending content on large Web portals in an online manner.

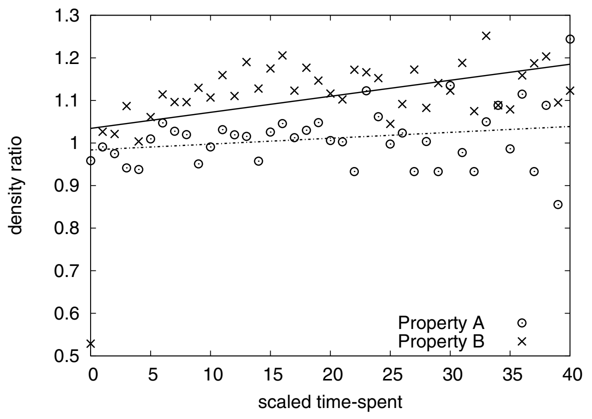

Figure 7: Time-spent distribution ratio between click shaping and status quo.

8. CONCLUSIONS AND FUTURE WORK

In this paper we proposed click shaping, an effective way to recommend content on large portal pages to optimize simultaneously for multiple engagement metrics. We formulated our problem in a constrained optimization framework which can naturally incorporate application-driven business requirement. In particular, we introduced a novel fairness objectives that aims to distribute gains in engagement in a fair way among heterogeneous domains and showed how this can be modeled in our framework. Through extensive data analysis on logged events, we provided unbiased estimates of performance for various flavors of our MOP under different constraints. Our data analysis reveals interesting characteristics of different formulations with tradeoffs among multiple metrics. We provide intuitive explanations of how clicks get shaped that in turn provides valuable guidance to the design of online recommendation engines.

Large variations in downstream engagement metrics provide more opportunities for lucrative tradeoffs; we are currently exploring methods to create fine grained user segments directly to maximize multiple objectives. Our MOP shows that imposing per epoch constraints is stringent when the goal is to obtain good performance over a larger time window. How to formulate solutions that ensure such long-term constraints but provide a per-epoch plan is a challenging problem.

9. REFERENCES

[1] D. Agarwal, B.-C. Chen, and P. Elango. Explore/exploit schemes for web content optimization. In *Proceedings of the Ninth International Conference on Data Mining*, 2009.

[2] D. Agarwal, B.-C. Chen, and P. Elango. Spatio-temporal models for estimating click-through rate. In *WWW*, pages 21–30, 2009.

[3] D. Agarwal, B.-C. Chen, P. Elango, and et al. Online models for content optimization. In *NIPS*, 2008.

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

[5] D. A. Berry and B. Fristedt. *Bandit Problems: Sequential Allocation of Experiments*. Monographs on Statistics and Applied Probability. Chapman and Hall, 1985.

[6] G. Buscher, L. van Elst, and A. Dengel. Segment-level display time as implicit feedback: a comparison to eye tracking. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 67–74, 2009.

[7] W. Chu, S.-T. Park, T. Beaupre, N. Motgi, A. Phadke, S. Chakraborty, and J. Zachariah. A case study of behavior-driven conjoint analysis on yahoo!: front page today module. In *KDD*, pages 1097–1104, 2009.

[8] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[9] I. S. Dhillon, J. Fan, and Y. Guan. Efficient clustering of very large document collections. In V. K. R. Grossman, C. Kamath and R. Namburu, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, 2001. Invited book chapter.

[10] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, Jan 2001.

[11] E.Vee, S.Vassilvitski, and J.Shanmugasundaran. Optimal online assignment with forecasts. In *EC*, 2010.

[12] J. P. Ignizio. *Goal programming and extensions*. Lexington Books, 1976.

[13] T. Jambor and J. Wang. Optimizing multiple objectives in collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 55–62, 2010.

[14] J. Langford, A. Strehl, and J. Wortman. Exploration scavenging. In *ICML*, pages 528–535, 2008.

[15] J. Langford and T. Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In *Advances in Neural Information Processing Systems 20*, 2008.

[16] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the Nineteenth International Conference on World Wide Web*, 2010.

[17] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *WSDM*, 2011.

[18] C. Liu, R. W. White, and S. T. Dumais. Understanding web browsing behaviors through weibull analysis of dwell time. In *SIGIR*, pages 379–386, 2010.

[19] R.Fain and J.Pederson. Sponsored search: A brief history. *Bulletin of American Society of Information and Technology*, 32:12–13, 2006.

[20] R.Steuer. *Multi-criteria optimization: theory, computation, and application*. Wiley, 1986.

[21] S.Boyd and L.Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[22] D. Sculley, R. G. Malkin, S. Basu, and R. J. Bayardo. Predicting bounce rates in sponsored search advertisements. In *KDD*, pages 1325–1334, 2009.

[23] K. M. Svore, M. N. Volkovs, and C. J. C. Burges. Learning to rank with multiple objective functions. In *WWW*, 2011.

[24] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. In *In European Conference on Machine Learning*, pages 437–448. Springer, 2005.