Personalized Click Shaping through Lagrangian Duality for Online Recommendation*

Deepak Agarwal†, Bee-Chung Chen†, Pradheep Elango‡, Xuanhui Wang‡
†LinkedIn, Mountain View, CA
{dagarwal, bchen}@linkedin.com
‡Facebook, Menlo Park, CA
{pradheep, xuanhui}@fb.com

ABSTRACT

Online content recommendation aims to identify trendy articles in a continuously changing dynamic content pool. Most of existing works rely on online user feedback, notably clicks, as the objective and maximize it by showing articles with highest click-through rates. Recently, click shaping [4] was introduced to incorporate multiple objectives in a constrained optimization framework. The work showed that significant tradeoff among the competing objectives can be observed and thus it is important to consider multiple objectives. However, the proposed click shaping approach is segment-based and can only work with a few non-overlapping user segments. It remains a challenge of how to enable deep personalization in click shaping. In this paper, we tackle the challenge by proposing *personalized click shaping*. The main idea is to work with the Lagrangian duality formulation and explore strong convexity to connect dual and primal solutions. We show that our formulation not only allows efficient conversion from dual to primal for online personalized serving, but also enables us to solve the optimization faster by approximation. We conduct extensive experiments on a large real data set and our experimental results show that the personalized click shaping can significantly outperform the segmented one, while achieving the same ability to balance competing objectives.

Categories and Subject Descriptors: H.3.5 [Information Storage and Retrieval]: Online Information Services

General Terms: Algorithms

Keywords: Personalization, click shaping, dual plan

1. INTRODUCTION

A significant portion of content on the web is ephemeral, new items are continuously published through various channels and obsolete ones are removed quickly. For example, content on Web portals (e.g., Yahoo!, AOL, and MSN) are refreshed frequently to recommend relevant and timely content to users. With overwhelming amount of fast changing information and unique tastes of individual users, it becomes critical to recommend interesting content to users in a personalized and timely manner.

Traditional information filtering provides personalized recommendation based on historical user ratings on items and includes both content-based filtering and collaborative filtering (e.g., [27, 13]). These methods usually work well on a relatively stable item pool, such as movies or products, but they are inferior for online content recommendation which deals with fast changing item pool such as news articles [2]. The main challenge in online content recommendation problems is to quickly identify the most interesting items (e.g., breaking news). In such a setting, it is also hard to collect a data set with explicit user ratings on items. Instead, most of online recommendation works [2, 17] rely on implicit user feedback, notably clicks, and optimize click-rates by showing articles with the highest click-through rates(CTR) that are estimated through explore/exploit techniques [16, 5].

However, using CTR as the only objective is not satisfactory because it cannot fully reflect other important engagement metrics such as number of page views or the total time-spent on a Website, which are indicative of user satisfaction [8] and also reported by third-party metric companies (e.g., comScore). Recently, click shaping [4] was introduced to tackle the problem of recommending items to jointly optimize for multiple objectives: clicks and post-click downstream utilities. The work showed that significant tradeoffs among these competing objectives can be observed and a constrained optimization formulation is an effective implementation. However, the proposed approach in [4] assumes that users are partitioned into a few coarse non-overlapping segments. In many applications a user is characterized by a high-dimensional feature vector (thousands of dimensions with large number of all possible combinations), segmented click-shaping fails to provide recommendations at such granular resolutions (user or fine grained user segments) because decisions are made at coarse user-segment resolutions in epochs — in the current epoch (e.g., 10-minute interval), a linear program (LP) is solved to decide a *serving plan* $\{x_{ij}\}$: for every (user segment i, item j) pair, x_{ij} represents the probability that the system will show item j to user segment i in the next epoch.

While it looks we can apply the LP formulation at granular user resolution by having one variable x_{uj} for each (user u, item j) pair, such a naive extension is infeasible due the

*This work was conducted when all authors were affiliated with Yahoo!

SIGIR'12, August 12–16, 2012, Portland, Oregon, USA.

following reasons: (1) This makes the LP ill-defined in the online setting because there are *unseen* users (user segments) that are observed for the first time in the next epoch. The modified LP has to predict the set of unseen users(unseen user segments) and include corresponding variables in order to serve them in the next epoch; (2) Even if we can accurately predict which users(user segments) will be seen in the next epoch, such an extension increases the size of the LP dramatically since there are usually hundreds of thousands users(user segments) per epoch. The segmented models can bypass these issues by assigning a new user to one of the predefined segments and use the segment serving plan for the new user. Thus, there are inherent limitations to directly utilize segmented click shaping for personalization. On the other hand, when optimizing clicks is the only objective, personalized models [22, 3] usually achieve much better results than segmented models. How to enable personalization in click shaping is a challenging, but important research problem that we address in this paper.

In this paper, we propose *personalized click shaping* which combines deep personalization with click shaping for multi-objective online content recommendation. Instead of working with the *primal* serving plan x_{ij}'s, our key idea is to convert the primal problem through Lagrangian duality and obtain the *dual* serving plan. By exploiting strong convexity of our objective function, we show that the dual plan can be efficiently converted to the primal plan for online personalized serving. Furthermore, our dual formulation can also enable us to solve the optimization faster through approximation. Specifically, we make the following contributions in this paper:

- In Section 3, we propose a novel technique to solve the aforementioned unseen user problem based on Lagrangian duality. By adding some small additional terms to the original linear objective function to achieve strong convexity, we show that the large number of primal variables x_{uj}'s can be obtained as a function of a small number of *user-independent* dual variables. Thus, in each epoch, we obtain the dual plan for the next epoch and, at serving time, we efficiently convert the dual solution on the fly to obtain the primal solution x_{uj} for each user u, seen or unseen.
- In Section 4, we develop a variety of approximation methods for efficient computation. Two clustering-based methods and two types of sampling-based methods are introduced. Our results show that clustering-based methods fail to work with the dual formulation, while simple random sampling is as good as more sophisticated stratified sampling methods.
- In Section 5, we empirically show that personalized click shaping significantly outperforms and uniformly dominates segmented click shaping for all Pareto optimal points based on extensive experiments on a large real data set using a recently proposed unbiased evaluation methodology [18].

2. PRELIMINARIES

In this section, we describe the problem setting and review segmented click shaping [4].

Application setting: Consider a content recommendation module on the front page of a portal such as Yahoo!. In each bucketized time epoch t, there is a set of candidate items available for recommendation, denoted by $\mathcal{A}_t$. Each item $j \in \mathcal{A}_t$ belongs to one of K different properties of the portal (e.g., Yahoo! News, Finance, Sports, etc). Let $\mathcal{P} = \{P_1, ..., P_K\}$ denote the set of properties and $j \in P_k$ means the landing page of item j belongs to property P_k. Let $\mathcal{U}_t$ denote the set of all users in epoch t. A user $u \in \mathcal{U}_t$ visits the front page and is recommended an item from $\mathcal{A}_t$. We only consider single item recommendation in this paper; multi-item problem is more involved and left as future work. When a user clicks the recommended item, it routes her to a page in the property that the clicked item belongs to. User visits to different properties are usually significantly influenced by the clicks on the front page. For the sake of illustration, consider that the portal wants to optimize for two different objectives — (a) total number of clicks on the front page and (b) total time-spent on landing properties by users who click recommended items on the front page. Note that other objectives can be incorporated in our framework.

Explore/exploit: Content optimization is an explore/exploit problem. To optimize for any metric, we need to estimate the performance of each candidate item in terms of that metric. Without displaying an item to any user, it is difficult to know the performance of that item. The explore/exploit problem for a single objective is well studied (e.g., [3, 18]). Developing explore/exploit methods for multiple objectives that ensure certain notion of optimality is nontrivial. 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 [26]. This scheme works as follows: We serve a small fraction of randomly selected user visits (called exploration population) with items selected uniformly at random from the current content pool in order to collect data for every item. For the remaining visits (called exploitation population), we serve the item with the highest estimated click-through rate (CTR) if the goal is to maximize clicks. With multiple objectives, the serving scheme for the exploitation population is different from that of displaying the highest CTR item as we shall see later in this paper.

Segmented models: An important assumption made in the segmented click shaping work [4] is that users are clustered into m segments. Let $\mathcal{S}$ denote the set of segments and $i \in \mathcal{S}$ denote a segment in the rest of the paper. To measure the utility of an item j, statistical models are used to estimate: (a) the probability p_{ijt} that a user in segment i would click item j when it is displayed in epoch t, and (b) the time d_{ijt} that a user in segment i would spend post-click in the landing property of item j. We call p_{ijt} CTR and d_{ijt} time-spent. Note that if the goal is to maximize clicks (or time-spent, but not both), the optimal solution is to recommend the item j with the highest p_{ijt} (or $p_{ijt} \cdot d_{ijt}$) to users in segment i.

Segmented serving scheme: We call an algorithm that recommends items to users a serving scheme. For each epoch t, a segmented serving scheme uses information obtained before the epoch to produce a *segmented serving plan* $\mathbf{x}_t = \{x_{ijt} : i \in \mathcal{S}, j \in \mathcal{A}_t\}$, where x_{ijt} is the probability that the serving scheme will recommend item j to users in segment i in epoch t. When a user visits the front page in epoch t, she is first assigned to an appropriate segment and then an item is served through a multinomial draw according to $\{x_{ijt} :$

$j \in \mathcal{A}_t\}$. It should be clear that $x_{ijt} \geq 0$ and $\sum_j x_{ijt} = 1$. Different optimization methods generate different serving plans according to different criteria. For example, the click maximization method would set x_{ij^*t} to 1, if j^* has the highest CTR item, and 0 for the remaining items. Note that the serving plan for epoch t is made before epoch t, i.e., in epoch $t-1$ in our application setting.

Objectives: Recall that we consider two objectives in this paper, number of clicks and time-spent. Let N_t denote the total number of visits during epoch t, and 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 of segment i. Usually, $\boldsymbol{\pi}_t$ can be estimated based on past user visits [4]. For succinctness, we drop subscript t from the notations since we always consider the current epoch t. For example, $\mathbf{x} = \{x_{ij}\}$ is the serving plan for the epoch t. Given serving plan $\mathbf{x}$, the two objectives are:

- The expected total number of clicks on the front page:

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

- The expected total time-spent on property P_k:

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

  We use $\mathit{TotalTime}(\mathbf{x}) = \mathit{TotalTime}(\mathbf{x}, \mathcal{A})$ to denote the total time-spent on all properties.

Click maximization scheme: The status quo algorithm is to optimize the total number of clicks. We can obtain the serving plan as $\mathbf{z} = \{z_{ij}\}$ in the following,

$$z_{ij} = \begin{cases} 1 & \text{when } j = \arg\max_{j'} p_{ij'} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

We use $\mathit{TotalClicks}^* = \mathit{TotalClicks}(\mathbf{z})$ and $\mathit{TotalTime}^*(P_k) = \mathit{TotalTime}(\mathbf{z}, P_k)$ to denote the values of the two objectives for the click maximization scheme; they are constants since $\mathbf{z}$ is a set of constants defined above.

Linear program: Although a variety of multi-objective programs (MOP) were introduced in [4], we only discuss the most flexible formulation *localized multi-objective program* (ℓ-MOP) in this paper. Formally, the optimization problem is given below.

$$\begin{aligned} \max_{\mathbf{x}} \ & \mathit{TotalTime}(\mathbf{x}) \\ \text{s.t.} \ & \mathit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \mathit{TotalClicks}^* \\ & \mathit{TotalTime}(\mathbf{x}, P_k) \geq \beta \cdot \mathit{TotalTime}^*(P_k), \ \forall k \in \mathcal{I} \end{aligned} \quad (4)$$

where $\mathcal{I}$ is a subset of $\mathcal{P}$, for which we want to ensure certain level of time-spent. This linear program seeks to maximize the total time-spent on all properties, such that the potential loss in the total number of clicks is bounded by α ($0 \leq \alpha \leq 1$) compared to the status quo click maximization scheme. For each of the key properties $P_k \in \mathcal{I}$, the total time-spent on P_k is guaranteed to be at least β ($0 \leq \beta \leq 1$) times that of the click maximization scheme.

Summary: It is important to note that any linear constraint can be added to the linear program. The above choice is useful in our application setting and serves as a running example for illustration purposes. In every epoch, we use statistical models to predict p_{ij}, d_{ij} and π_i, and then solve the linear program to obtain the serving plan $\mathbf{x}$ for the next epoch. Thus, user visits in the next epoch will be served according to the plan made before we actually see the users.

3. PERSONALIZED CLICK SHAPING

As shown above, linear programs can be effectively used to perform segmented click shaping. However, such a segmented serving scheme treats all the users in a segment in the same way — it lacks the ability of serving each individual user differently to satisfy his/her unique personal information need. In this section, we extend click shaping beyond user segments to provide personalized serving plan for individual users.

3.1 Primal Formulation

In this section, we define the primal formulation of personalized click shaping by extending the segment-based linear program in Equation 4 through redefining the form of serving plan $\mathbf{x}$.

Personalized serving plan: For personalization, we have a user-specific serving plan for each user u, instead of segment-specific ones. Thus the personalized serving plan is defined as $\mathbf{x} = \{x_{uj} : u \in \mathcal{U}, j \in \mathcal{A}\}$, where x_{uj} is the probability that user u will be served with item j in the next epoch. Since the x_{uj}'s are probabilities, we have $x_{uj} \geq 0$ and $\sum_j x_{uj} = 1$. A personalized serving scheme serves items to each individual user according to these probabilities.

Objectives: Because the form of serving plan $\mathbf{x}$ changed, the definitions of $\mathit{TotalClicks}(\mathbf{x})$ and $\mathit{TotalTime}(\mathbf{x}, P_k)$ also need to be updated.

$$\begin{aligned} \mathit{TotalClicks}(\mathbf{x}) &= \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{A}} x_{uj} p_{uj}, \\ \mathit{TotalTime}(\mathbf{x}, P_k) &= \sum_{u \in \mathcal{U}} \sum_{j \in P_k} x_{uj} p_{uj} d_{uj}, \end{aligned} \quad (5)$$

where p_{uj} is the probability that user u would click item j and d_{uj} is the length of time that user u would spend on pages in the landing property of item j after he/she clicks the item. Both the estimation of p_{uj} and d_{uj} are orthogonal to our problem formulation and any personalized statistical model such as online regression proposed in previous works [3, 2] can be plugged in. Based on CTR estimation p_{uj}, we can compute the click optimization serving scheme similarly to Equation 3 and define the status quo constants $\mathit{TotalClicks}^*$ and $\mathit{TotalTime}^*$ for personalized serving accordingly. Please note that if we instantiate p_{uj} and d_{uj} using a segment-based model (i.e., $p_{uj} = p_{ij}$ and $d_{uj} = d_{ij}$ for all u in segment i), Equation 5 will reduce to Equation 1 and Equation 2.

Challenges: With these new definitions of $\mathbf{x}$, $\mathit{TotalClicks}$ and $\mathit{TotalTime}$, on the surface, Equation 4 can be trivially applied to personalized click shaping: We can solve this linear program in each epoch and use the solution to serve users in the next epoch. However, such a formulation is challenging for the following reasons:

- **Unseen users:** In the linear program, the variables x_{uj}'s are meant for all users who visit in the *next* epoch. For those next epoch users who have not visited the portal before, computing x_{uj}'s is challenging. Although we may be able to estimate the number of users (which

determines the number of variables), it is difficult to predict who exactly will visit in the next epoch. But the LP input p_{uj} and d_{uj} are required to be known for all users, including unseen ones.

- **Scalability:** This linear program requires a set of variables $\{x_{uj}\}_{\forall j \in \mathcal{A}}$ for each user u, which is in fact the serving plan for that individual user. Even if we know p_{uj} and d_{uj} for all the users in the next epoch, when the number of users is large (e.g., millions), the large number of variables results in a very large linear program and thus causes scalability issues.

Key idea: To tackle these challenges, we exploit Lagrangian duality formulation of our constrained optimization problem. In the primal program of Equation 4, x_{uj}'s are the primal variables. Although we have a large number of *user-specific* primal variables, there are only a small number of nontrivial constraints in the primal formulation. Our key idea is to explore Lagrangian duality to capture the primal variables by using a small number of *user-independent* dual variables, one per constraint in the primal program. Now suppose that we can efficiently compute the optimal dual solution for the next epoch (which will be discussed in Section 4). Also suppose that in the next epoch, we can efficiently convert the dual solution on the fly to the primal solution (i.e., serving plan) for each individual user at serving time. Then the challenges are solved.

Unfortunately, linear programs do not allow easy conversion from dual solutions to primal solutions and vice versa because the derivatives of the Lagrangian vanish [7]. To ensure convertibility, we slightly modify the originally linear objective function so that it becomes strongly convex. Intuitively, let $\mathbf{q} = \{q_{uj} : u \in \mathcal{U}, j \in \mathcal{A}\}$ be some baseline serving plan. We add terms to the objective function to penalize serving plans $\mathbf{x}$ that are far away from $\mathbf{q}$. There are a few choices for the baseline $\mathbf{q}$. One is the click maximization plan $\mathbf{z}$. Another is the uniform serving plan $q_{uj} = 1/|\mathcal{A}|$, which will encourage some notion of "fairness" among items. These penalty terms can be the L2 norm or KL divergence. We focus on the L2 norm penalty term and the uniform serving plan $\mathbf{q}$ in this paper.

$$||\mathbf{x} - \mathbf{q}||^2 = \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{A}} (x_{uj} - q_{uj})^2$$

Revised primal program: After adding the penalty term, we obtain the revised primal program.

$$\begin{aligned} \min_{\mathbf{x}} \ & \frac{1}{2}\gamma ||\mathbf{x} - \mathbf{q}||^2 - \mathit{TotalTime}(\mathbf{x}) \\ \text{s.t.} \ & \mathit{TotalClicks}(\mathbf{x}) \geq \alpha \cdot \mathit{TotalClicks}^* \\ & \mathit{TotalTime}(\mathbf{x}, P_k) \geq \beta \cdot \mathit{TotalTime}^*(P_k), \ \forall k \in \mathcal{I}, \end{aligned} \tag{6}$$

where γ specifies the importance of the penalty. Such a modification is generic since it can be applied to any linear program based click shaping. Notice that we also change from maximization to its equivalent minimization to put the primal program in the standard form of quadratic programming problem. In some sense, the added penalty term serves as regularization and can potentially reduce the variance of the solution.

3.2 Lagrangian Duality

We now introduce the dual variables and an algorithm to efficiently convert a dual solution to its corresponding primal solution. For ease of exposition, let

$$g_0 = \alpha \cdot \mathit{TotalClicks}^* \quad \text{and} \quad g_k = \beta \cdot \mathit{TotalTime}^*(P_k).$$

The Lagrangian function of the primal program is

$$\begin{aligned} \Lambda(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\delta}) = & \tfrac{1}{2}\gamma \textstyle\sum_u \sum_j (x_{uj} - q_{uj})^2 - \sum_u \sum_j p_{uj} d_{uj} x_{uj} \\ & - \mu_0 (\textstyle\sum_u \sum_j p_{uj} x_{uj} - g_0) \\ & - \textstyle\sum_{k \in \mathcal{I}} \mu_k (\sum_u \sum_{j \in P_k} p_{uj} d_{uj} x_{uj} - g_k) \\ & - \textstyle\sum_u \nu_u (\sum_j x_{uj} - 1) - \sum_u \sum_j \delta_{uj} x_{uj}, \end{aligned}$$

where $\mu_0 \geq 0$, $\mu_k \geq 0$, for all $k \in \mathcal{I}$ and $\delta_{uj} \geq 0$, for all u and j. This is obtained by expanding *TotalTime* and *TotalClick* according to Equation 5 and applying the Lagrange multipliers μ_0 to ensure the total click constraint, μ_k to ensure the per-property total time constraint, ν_u to ensure $\sum_j x_{uj} = 1$, and δ_{uj} to ensure $x_{uj} \geq 0$. Note that μ_0, μ_k, ν_u and δ_{uj} are also called the *dual variables*.

By setting $\frac{\partial}{\partial x_{uj}} \Lambda(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\delta}) = 0$, we obtain

$$x_{uj} = \frac{c_{uj} + \nu_u + \delta_{uj}}{\gamma}, \tag{7}$$

where $c_{uj} = p_{uj} d_{uj} + \mu_0 p_{uj} + \mathbf{1}_{\{j \in P_k \wedge k \in \mathcal{I}\}} \mu_k p_{uj} d_{uj} + \gamma q_{uj}$. Note that $\mathbf{1}_{\{\text{True}\}} = 1$ and $\mathbf{1}_{\{\text{False}\}} = 0$. Thus, if we have the dual solution for μ_0, μ_k, ν_u and δ_{uj}, we can reconstruct the primal solution x_{uj}. However, this does not help to solve the challenges since ν_u and δ_{uj} still depend on the users u in the next epoch. In the following, we provide an efficient algorithm that reconstructs x_{uj} from $\boldsymbol{\mu} = \{\mu_0, \mu_k\}_{\forall k \in \mathcal{I}}$ alone without the need for ν_u and δ_{uj}.

Dual serving plan: We call $\boldsymbol{\mu}$ the *dual serving plan*, which does not include any user-specific variables. The algorithm that converts a dual plan to its corresponding primal serving plan is based on the following proposition.

Proposition 1 *In the optimal solution, given user u and two items j_1 and j_2, if $c_{uj_1} \geq c_{uj_2}$ and $x_{uj_2} > 0$, then $x_{uj_1} > 0$.*

Proof sketch: According to the KKT conditions, at the optimum point, $\delta_{uj_2} = 0$ since $x_{uj_2} > 0$. Then, because $c_{uj_1} \geq c_{uj_2}$ and $\delta_{uj_1} \geq 0$, we have

$$x_{uj_1} = \frac{c_{uj_1} + \nu_u + \delta_{uj_1}}{\gamma} \geq \frac{c_{uj_2} + \nu_u}{\gamma} = x_{uj_2} > 0. \ \square$$

Without loss of generality, for each user u, we reindex items such that $c_{u1} \geq c_{u2} \cdots \geq c_{un}$, where n is the number of items. Notice that this ordering is user-specific. Based on Proposition 1, there exists a number $1 \leq t \leq n$ such that, in the optimal solution, $x_{uj} > 0$ for $j \leq t$ and $x_{uj} = 0$ for $j > t$. To find the value of t, we check from $t = 1$ to n whether the following linear system has a feasible solution:

$$\begin{aligned} & x_{uj} = \frac{c_{uj} + \nu_u}{\gamma} \ \text{ and } \ x_{uj} > 0, \ \text{for } 1 \leq j \leq t \\ & \textstyle\sum_{j=1}^{t} x_{uj} = 1. \end{aligned}$$

Notice that $\delta_{uj} = 0$ if $x_{uj} > 0$. The largest t value that still gives a feasible solution is what we are looking for. By some algebra, given t, we have $\nu_u = (\gamma - \sum_{j=1}^{t} c_{uj})/t$. The above system is feasible if the smallest $x_{ut} > 0$; i.e.,

$$x_{ut} \propto c_{ut} + \tfrac{\gamma - \sum_{j=1}^{t} c_{uj}}{t} > 0. \tag{8}$$

Algorithm 1 Conversion algorithm

Input: dual plan $\boldsymbol{\mu}$ and an incoming user u
Output: primal plan $\{x_{uj}\}$

```
Predict p_uj and d_uj for each item j
Compute c_uj based on μ, p_uj and d_uj
Order items by c_uj so that c_u1 ≥ c_u2 ≥ ...
Set a = γ and t = 1
repeat
    if c_ut + (a − c_ut)/t ≤ 0 then
        t = t − 1 and break
    else
        a = a − c_ut and continue
    end if
    t = t + 1
until t ≥ |A|
ν_u = a/t
for j = 1 to t do
    x_uj = (c_uj + ν_u)/γ
end for
return {x_uj}
```

Conversion algorithm: Given dual plan $\boldsymbol{\mu}$ and an incoming user u, the primal serving plan $\{x_{uj}\}$ for u is obtained by the conversion algorithm which is summarized in Algorithm 1.

Proposition 2 *If the input dual plan $\boldsymbol{\mu}$ is optimal for a set of users, then the output serving plan from the conversion algorithm is also optimal for the same set of users.*

Proof sketch: Based on dual variable $\boldsymbol{\mu}$, the conversion algorithm gives us $\boldsymbol{\nu}$ and $\mathbf{x}$. We can further compute δ_{uj} for all $x_{uj} = 0$ based on Equation 7. It can be verified that all these values satisfy all the KKT conditions. □

The complexity of the conversion algorithm is dominated by the prediction of p_{uj} and d_{uj} and the sorting of c_{uj}'s. Since the number of items is usually small (from hundreds to thousands), the conversion algorithm is very efficient. Furthermore, the computation can be easily paralleled by users because each user can be computed independently from others.

Discussion: By comparing the above formulation with a simple scalarization that linearly combines different objectives, their similarities lie in the weights μ_k's which can be thought as the importance for different objectives/constraints. However, scalarization is limited in its ability to obtain all the meaningful Pareto optimal points because it does not allow for fractional serving [4, 7]. The existence of ν_u in our formulation can achieve fractional serving and thus can move the Pareto optimal solutions in a more controlled way than scalarization.

4. APPROXIMATION METHODS

Since we do not observe all the users who will visit in the next epoch, we can only solve the QP approximately. The main idea is to exploit the observation that the distribution of users in the next epoch is similar to that in the current epoch. This is a mild requirement because our epoch is usually of short duration (10 minutes in our paper) and user population usually does not change significantly. It is important to note that, if we use the primal solution directly for serving, we can only serve the set of seen users because we do not have primal variables for unseen users. However, if we use the dual plan for serving, we can convert it to the per-user serving plan on the fly for all users. Our dual plan only requires that the sets of user features in the two adjacent epochs are statistically similar(we do not require users themselves to be similar). For this reason, our dual formulation can even work with a sample of users. In this section, due to the potentially large number of users in an epoch, we explore two techniques to reduce computational cost: clustering and sampling.

4.1 Clustering

Our goal is to obtain the dual solution for the QP defined in Equation 6. To reduce the QP problem size, an intuitive choice is to cluster users and then we have a small number of primal variables. Specifically, we define a set of primal variables $\{x_{ij}\}_{\forall j \in \mathcal{A}}$ for each cluster i, instead of each user u. An off-the-shelf QP solver is then used to find the dual solution $\boldsymbol{\mu}$ of the small QP problem. Finally, at serving time, $\boldsymbol{\mu}$ is used to compute the personalized serving plan $\{x_{uj}\}_{\forall j \in \mathcal{A}}$ for each individual user u using Algorithm 1. Two clustering methods are considered:

- **k-Means:** The standard k-means algorithm is applied to the set of users in the current epoch to create m clusters based on the similarity between users.
- **Top1Item:** This method puts all the users having the same highest CTR item into the same cluster. Specifically, we name clusters by item IDs. The set of users in cluster j is

$$S_j = \{u \in \mathcal{U} : j = \arg\max_{j' \in \mathcal{A}} \ p_{uj'}\}.$$

  For consistency, we will still use index i to refer to such a cluster in general. Please note that the number of clusters in this method does not need to be specified.

After we get the user clusters, we approximate the QP as follows: for each cluster i, we estimate p_{ij} and d_{ij} by averaging the per-user p_{uj} and d_{uj} over users u in the cluster. Also, to ensure feasibility, the baseline performance *TotalClicks** and *TotalTime** are defined based on the click maximization scheme in Equation 3. We then solve the QP to get the dual optimal values $\boldsymbol{\mu}$.

4.2 Sampling

Another way to reduce computational cost is to downsample users. We consider two types of sampling methods:

- **Random sampling:** Given a sample rate r, we randomly select r% users uniformly in the current epoch.
- **Stratified sampling:** Given a set of user clusters, we randomly sample r% users for each individual user cluster. The clusters can be created by k-means or Top1Item.

After sampling, we obtain a subset of users in the current epoch. When we solve the QP, we define a set of primal variables $\{x_{uj}\}_{\forall j \in \mathcal{A}}$ only for sampled users u. Again, we update the baseline performance *TotalClicks** and *TotalTime** by only considering the sampled users. Then, the small QP is solved to obtain the dual solution $\boldsymbol{\mu}$ which is used by the conversion algorithm for online personalized serving.

Name	Method Description
Segment-LP	Segment-based linear program, same as [4].
Clustering-based Methods (Section 4.1)	
kMeans-QP-Dual	k-means clustering and personalized serving using the dual solution
Top1Item-QP-Dual	Top1Item clustering and personalized serving using the dual solution
Sampling-based Methods (Section 4.2)	
Random-Sampling	Random sampling and personalized serving using the dual solution
Stratified-kMeans	Stratified sampling using k-means, and personalized serving using dual
Stratified-Top1Item	Stratified sampling using Top1Item, and personalized serving using dual

Table 1: Summary of the methods.

5. EXPERIMENTS

In this section, we compare different methods for click shaping as summarized in Table 1. Note that Segement-LP relies on the primal optima directly for *segmented* serving, while kMeans-QP-Dual and Top1Item-QP-Dual rely on dual optima for *personalized* serving. We report our extensive empirical results using an unbiased evaluation methodology [18] based on Yahoo! Front Page log data. We will mainly compare the personalized click shaping with the most effective segmented ones in prior work [4]. We also compare the proposed methods in terms of their abilities to enforce constraints and achieve desired tradeoff between clicks and time-spent.

5.1 Experimental Setup

Data: Our data set is derived from Yahoo! Web server logs, which record users' clicks and views of items displayed in the Today module on the Yahoo! Front Page. The data is collected from a "random bucket" of users in August 2010 for the purpose of evaluating different serving schemes. A random sample of users was assigned to the random bucket, in which each user visit was served with an item uniformly randomly picked from an editor-selected item pool. It is important to note that, based on the replay methodology described in [18], this random bucket data allows us to perform *provably unbiased* comparisons among different serving schemes. Around 2 million click and view events were collected per day. 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 item. Each user is identified by an anonymized browser-cookie and has a profile consisting of demographics (age and gender) and his/her affinities to different categories (such as sports, finance and entertainment) based on his/her activities throughout the Yahoo! network. No personally identifiable information is used in our experiments. To create user segments or clusters, we use the best performed method proposed in [4]. Specifically, we collect a history data of 10 days in April 2010 and create an activity vector for each user. Each entry in the vector corresponds to an item in the history data and the value is the predicted CTR of the item based on the user profile. Users are then clustered using the k-means algorithm based on the activity vectors and new users are assigned to a cluster based on cosine similarity. This applies to all k-means related methods.

Metrics: We define the time-spent d_{uj} of a user u 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 $\rho_{\text{TS}} = \frac{q_{\mathsf{A}}}{q_{\mathsf{B}}}$.

Age-gender model		Linear regression	
MAE	RMSE	MAE	RMSE
86.11	**119.44**	87.75	122.02

Table 2: Comparison of time-spent prediction.

Estimation of p_{ujt} and d_{ujt}: Predicted CTR p_{ujt} and time-spent d_{ujt} are necessary input to our quadratic program. The choice of statistical models for such prediction is orthogonal to the methods presented in this paper and any model can be used here. In our experiments, we use the online logistic regression (OLR) model proposed in [3] to predict CTR p_{ujt} based on the feature vector (i.e., the profile consisting of demographics and category affinities) of user u. One OLR model is trained for each item and updated per epoch, in order to quickly capture the unique behavior of each item that does not generalize well through item features. Time-spent d_{ujt} is predicted by an age-gender model. We discretize ages into 10 groups and have 3 gender groups (male, female, and unknown); this gives 30 groups in total. For each group, we use a dynamic Gamma-Poisson model to track the mean of time-spent d_{ujt} on item j by a random user u in the group. Note that because time-spent is very noisy, this simple age-gender model cuts down the variance and provides good prediction performance. More fine-grained models such as building an linear regression model on user features for each item were tested, but did not provide improvement: Table 2 compares the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) of the two methods using 2-fold cross validation. Age-gender model performs just slightly better. A better time-spent model needs further research and is not the focus of this paper.

5.2 Experimental Results

Advantage of the personalized click shaping: We first show the that personalized click shaping significantly outperforms segmented click shaping in Figure 1. We start with a simplified special case that only considers the tradeoff between total clicks and total time-spent and does not have any per-property constraint. We show the results of Random-Sampling method with sampling rate $r = 20\%$ and $\beta = 0$. The tradeoff curve is generated by varying α from 1 to 0 (where each point is produced by averaging over 5 sampling runs). In this simplified setting without per-property constraints, *scalarization* is also applicable: serving each user u with the item j that has the highest weighted sum of the two objectives: $\lambda \cdot p_{uj} + (1-\lambda) \cdot p_{uj} d_{uj}$. By varying λ from 1 to 0, we obtain a tradeoff curve for scalarization in Figure 1. Each specific value of λ or α gives us a serving scheme that,

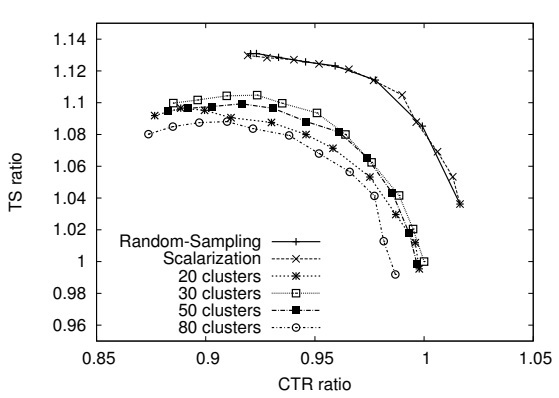

Figure 1: Personalized vs the segmented methods.

Method	CTR ratio		TS ratio	
	mean	stdev	mean	stdev
Scalarization (λ=0.50)	0.9601	0.0394	1.0796	0.0403
Random-Sampling (α=0.97)	0.9605	0.0306	1.0778	0.0322

Table 3: Temporal variance comparison.

after running through the data set, generates a pair of CTR ratio and TS ratio. To compare with segmented click shaping in this setting, we also plot the tradeoff curve using the scalarization method with different number of clusters. All the implementation details are the same as [4]. These methods are labeled as *k clusters* in the figure. From this figure, we can see that both versions of personalized click shaping outperform all the segmented click shaping on the tradeoff curves. For example, when $\lambda = 1$ and $\alpha = 1$, the personalized models achieve 2% CTR lift (statistically significant at level 0.1) and 4% TS lift (statistically significant at level 0.05). Thus working with personalized models is beneficial. Segmented click shaping achieves the best performance when the number of cluster is 30. When the number of clusters become large, the performance drops mainly because we have few users in each cluster and the component estimation of p_{ijt} and d_{ijt} (based on Gamma-Poisson) has high variance due to smaller sample size.

Figure 1 shows that both Random-Sampling and scalarization achieves similar tradeoff. This means that our dual formulation combined with sampling-based approximation is very effective to trade off competing objectives. In Table 3, we show the temporal variances of both methods on the TS ratio and CTR ratio over all the epochs. As can be seen, the variances of scalarization are much larger than those of Random-Sampling, meaning that Random-Sampling is a more stable method. This observation is similar to [4] where primal plans are used. Thus our dual formulation enjoys the same desirable property of constrained optimization, which makes sure no significant deviation from status quo performance in most epochs. It is important to note that scalarization cannot provide detailed control over per-property performance constraints (as shown in [4]). Thus, we do not further discuss scalarization in the following.

Constraint satisfaction: We have proposed a variety of approximation methods for personalized click shaping. A main question is whether they can satisfy the per-property constraints. An approximation would not be useful if it cannot satisfy the constraints well. In this set of experiments, we set $\beta = 1$. Given an α value and a property P_k, we look

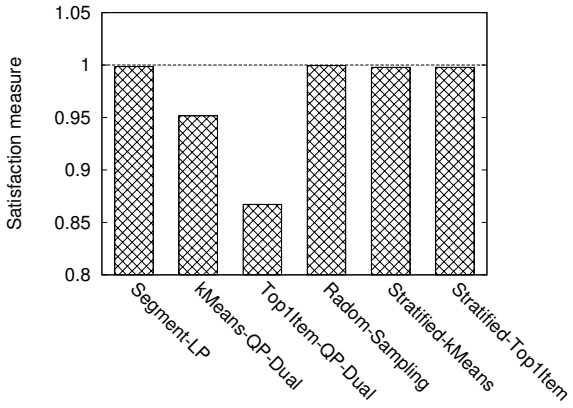

Figure 2: Satisfaction of constraints.

at the relative difference in time-spent on that property between an approximation method and the baseline click maximization scheme

$$\phi_\alpha(P_k) = \frac{TotalTime_\alpha(P_k) - TotalTime^*(P_k)}{TotalTime^*(P_k)}.$$

$\phi_\alpha(P_k) < 0$ means that the constraint on property P_k is violated and $|\phi_\alpha(P_k)|$ represents the degree of violation. We then define a satisfaction measure over all the per-property constraints as

$$\Phi_\alpha = 1 + \frac{1}{|\mathcal{V}|}\sum_{k\in\mathcal{V}} \phi_\alpha(P_k),$$

where $\mathcal{V} = \{k \,|\, \phi_\alpha(P_k) < 0, k \in \mathcal{I}\}$ is the set of properties which constraints are violated.

In Figure 2, we set $\alpha = 0.95$ and plot the satisfaction measure for each method. We set the sampling rate $r = 20\%$ for all sampling-based methods. As can be seen, all the methods except for clustering-based dual methods have $\Phi_\alpha \approx 1$ and thus satisfy the per-property constraints very well. In particular, all sampling-based dual methods satisfy the constraints well. However, clustering-based dual methods (kMeans-QP-Dual and Top1Item-QP-Dual) significantly violate the constraints. To further understand this behavior, we plot kMeans-QP-Dual and Top1Item-QP-Dual in Figure 3 for each individual property using $\phi_\alpha(P_k)$ with $\alpha = 0.95$. It can be seen that both do not respect the constraints and gives some properties more traffic than sufficient, while gives others less than necessary. This means that the clustering we tried is not a good approximation for personalized click shaping.

Tradeoff comparison: In Figure 4, we compare the tradeoff curves of different methods with per-property constraints by setting $\beta = 1$. Each curve is generated by varying α from 1 to 0. We will show that stratified sampling methods have very similar performance to Random-Sampling. Thus, for plot clarity, we only show the tradeoff curve for Random-Sampling (for which, we ran the experiments 5 times and show the average curve).

From this figure, we can see that the Random-Sampling method performs much better than the Segmented-LP method, under the per-property constraints. We can also see that Random-Sampling have the same ability to trade off the two competing objectives as Segmented-LP. Our Random-Sampling is personalized click shaping based on a dual plan,

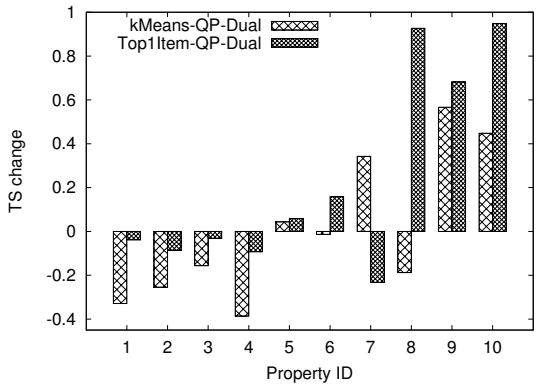

Figure 3: Per-property satisfaction for clustering.

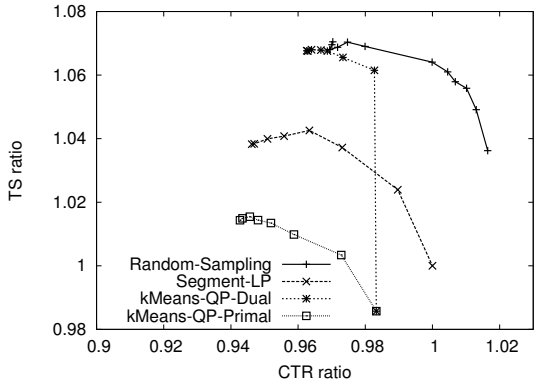

Figure 4: Tradeoff curves of different methods.

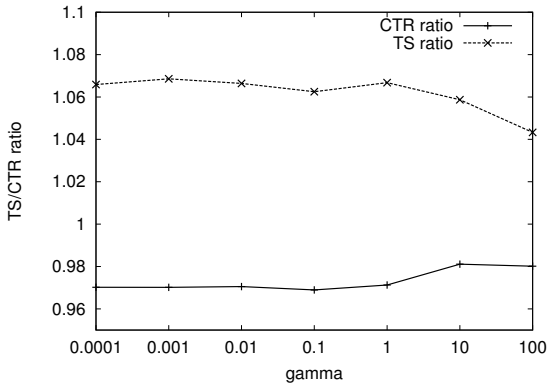

Figure 5: The impact of γ.

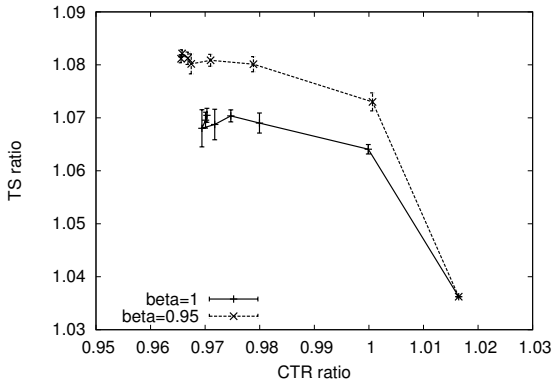

Figure 6: The tradeoff curve of different β values.

while Segmented-LP is based on a primal plan. This result confirms that our Lagrangian duality formulation is not only mathematically sound, but practically effective to incorporate personalization into click shaping.

In this figure, we also show the results of kMeans-QP-Dual for contrast. It can be seen that clustering-based approximation is not able to trade off the two objectives appropriately. The sharp jump in the tradeoff curve is a by-product of violating the per-property constraints as shown in the previous section. To confirm the correctness of our results, we also show the primal variation of k-means based approximation: kMeans-QP-Primal. This method shows a reasonable tradeoff curve. The difference between kMeans-QP-Dual and kMeans-QP-Primal is that the former uses dual plan but the latter uses the primal plan. The latter method is segmented because the primal serving plan is only for segments and each user needs to be assigned to a segment before served. This result again confirms that our clustering-based approximation is not an effective approach to approximate the Lagrangian duality.

The impact of γ: In Figure 5, we set $\alpha = 0.95$ and use the Random-Sampling method to show the impact of the parameter γ. It can be seen that when γ is less than 1, the results are not sensitive to the choice of γ. When γ is too large, the penalty term has high weight and thus the TS lift gets smaller. In all the experiments we set $\gamma = 0.001$.

Relaxation of the per-property constraints: In Figure 6, we show the tradeoff curves for $\beta = 1$ and $\beta = 0.95$. For each parameter setting, we have 5 runs of experiments and we show the mean of these 5 runs. We also plot the error bars of TS ratio. It can be see that we can achieve a better tradeoff when β is smaller because the constraints are less restrictive. We also show the $\phi_{\alpha=0.95}$ for both methods in Figure 7. It can be seen that the Random-Sampling method can roughly guarantee the per-property constraints. For $\beta = 1$, ϕ_α are almost all non-negative. For $\beta = 0.95$, ϕ_α are almost all ≥ -0.05.

Comparing different sampling methods: In Figure 8, we study the sampling ratios for Random-Sampling. As expected, when we have a larger number of samples, our approximation becomes better, thus a better tradeoff curve. In Figure 9, we show the (relative) running time needed for one run of our replay and we compare different sampling ratios. It can be seen that the running time is super-linear and this shows that sampling is necessary for our method. As shown in Figure 8, 20% sampling rate can achieve similar results as 50% sampling rate and thus sampling is an effective way of reducing the time complexity while retaining the effectiveness of our methods.

In Figure 10 and Table 4, we compare different sampling methods based on 5 sampling runs with 20% sample rate. Figure 10 shows the tradeoff curves averaged over the 5 runs and we can see that different sampling methods perform comparably with each other. In Table 4, we compute the click and TS variance as follows: For each α, we compute the standard deviation over the 5 runs. The reported re-

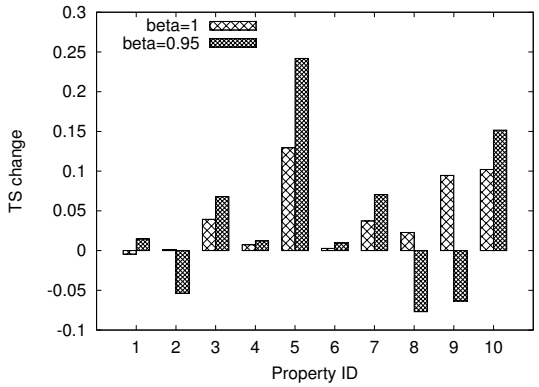

Figure 7: The impact of β on time-spent difference.

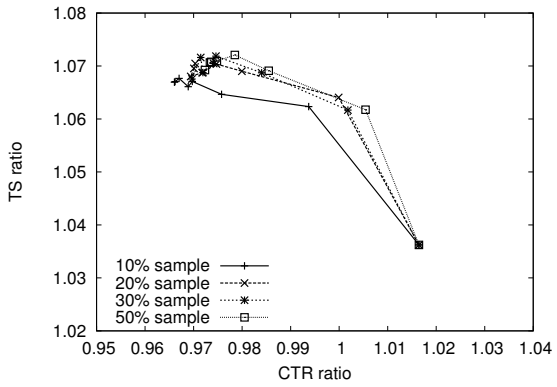

Figure 8: The impact of sample ratios.

sults are the average values of the standard deviation over all the α's. From this table, we can see that stratified methods have lower variance than random sampling. Among the two stratified methods, Stratified-kMeans has lower variance than Stratified-Top1Item. This means that k-means creates more homogeneous clusters than the Top1Item method.

Top N Properties: In Figure 11, we relax the number of per-property constraints in the Random-Sampling method and show the results when we only have the per-property constraints for the top 3, 5, and 7 properties. It can be seen that when the number of per-property constraints is reduced, we can have a better tradeoff curve. This again shows the effectiveness of dual formulation for personalized click shaping.

6. RELATED WORK

Our work is related to the broad category of Web content recommendation, which is to recommend users with *personalized* and *timely* content from a *dynamic* candidate pool such as news stories. For example, traditional collaborative filtering techniques [13, 15] have been extended to predict user clicks for news personalization [10, 20, 19]. To quickly identify interesting content, existing work such as [3, 2, 17] leverages explore/exploit methods [6, 16, 5] to actively collect user feedback (e.g., clicks) on new content in order to quickly estimate the performance of new content and maximize the number of clicks in the long run. The focus of

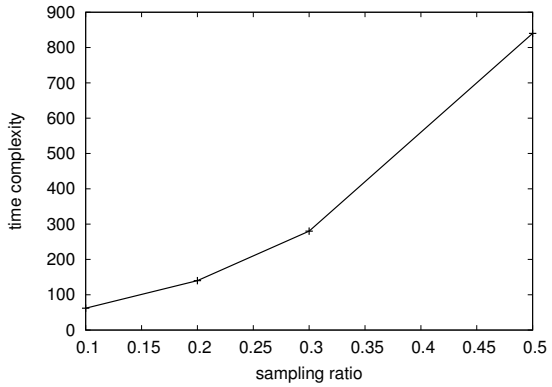

Figure 9: Time complexity vs sampling ratios.

	Random Sampling	Stratified-kMeans	Stratified-Top1Item
CTR stdev	0.00139	**0.00081**	0.00094
TS stdev	0.00187	**0.00129**	0.00156

Table 4: Average variance of sampling methods.

our paper is not on accurate prediction of user clicks, but on how to jointly optimize multiple objectives. Other work such as [1, 22] uses latent factor model to combine both features and interaction activities to handle both warm-start and cold-start scenarios. Although all these methods provide *personalized* recommendation, they only consider a single objective in their problem formulations (e.g., clicks). To the best of our knowledge, there is little prior work that attempts to recommend personalized content to simultaneously optimize multiple objectives. The previous work [4] introduces the problem of click shaping. However, the proposed methods only work with user segments, instead of individual users. A main contribution of our paper is to close the gap between personalization and multi-objective optimization in Web content recommendation.

Our constrained optimization problem is similar to the guaranteed delivery problem [25, 9] in display advertising, where incoming users are allocated to see different ads in order to optimize ad-related utilities. For example, Vee et al. [25] consider multi-objective programming and provide techniques to simultaneously optimize for revenue of remnant inventory and overall ad quality delivered to advertisers. Our use of duality is similar to that of [25], but our content recommendation setting is very different from their advertising setting. Also, the focus of [25] is on theoretical properties, while the focus of this paper is to provide extensive experiments on real data.

Multi-objective optimization is also discussed in a number of other problem settings. For instance, auctions in sponsored search incorporate both revenue (measured by bids) and ad quality (measured by CTR) for ads ranking [11]. A common approach is simply to rank ads by the product of bid value and CTR. Sculley et al. [21] also study ad CTR and quality in sponsored search and define a new measure called *bounce rate* to capture ad quality by inferring abandonment rate on ad landing page. However, this study is exploratory and focuses on bounce rate prediction, instead of multi-objective optimization. Besides online advertising, Jambor and Wang [14] consider constraints like limited supply of the items in a collaborative filtering setting; Svore et

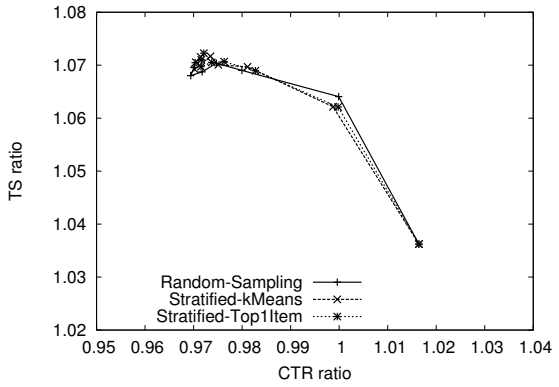

Figure 10: Comparison of different sampling methods (sampling ratio=0.2).

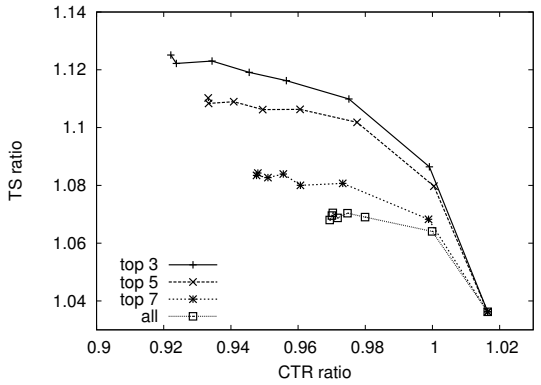

Figure 11: Impact of number of constraints.

al. [24] consider several objectives in learning to rank. These two studies are in a static setting, different from our online recommendation setting.

Finally, we note that there is a rich literature in multi-objective programming [23], convex optimization [7], and stochastic optimization [12]. Our contribution is in the application of optimization techniques to effective online personalized content recommendation.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we study the problem of personalized click shaping which combines deep personalization with click shaping. We have shown that Lagrangian duality can be effectively used to solve the two challenges in applying constrained optimization to personalized serving — unseen users and scalability. By slightly modifying the objective function to achieve strong convexity, we are able to efficiently convert a *dual plan*, which consists of a small number of *user-independent* dual variables, to its corresponding primal serving plan at the time when a new user comes. Based on extensive experiments on a large real dataset, we show the set of Pareto optimal points for personalized click shaping significantly outperforms and uniformly dominates the ones for segmented click shaping.

Interesting problems for future work include how to extend our method to the scenario where multiple items can be simultaneously recommended at multiple positions on a portal page, and how to extend the current per-epoch constraints (e.g., requiring click loss to be bounded for every epoch) to long-term constraints (e.g., only requiring the total click loss in any n-epoch period to be bounded).

8. REFERENCES

[1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD*, 2009.

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

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

[4] D. Agarwal, B.-C. Chen, P. Elango, and X. Wang. Click shaping to optimize multiple objectives. In *KDD*, 2011.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 2002.

[6] D. A. Berry and B. Fristedt. *Bandit Problems: Sequential Allocation of Experiments*. Chapman and Hall, 1985.

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

[8] G. Buscher, L. van Elst, and A. Dengel. Segment-level display time as implicit feedback: a comparison to eye tracking. In *SIGIR*, 2009.

[9] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur. Real-time bidding algorithms for performance-based display ad allocation. In *KDD*, 2011.

[10] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, 2007.

[11] R. Fain and J. Pederson. Sponsored search: A brief history. *Bulletin of American Society of Information and Technology*, 2006.

[12] P. V. Hentenryck and R. Bent. *Online Stochastic Combinatorial Optimization*. The MIT Press, 2006.

[13] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR*, 1999.

[14] T. Jambor and J. Wang. Optimizing multiple objectives in collaborative filtering. In *RecSys*, 2010.

[15] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, 2009.

[16] J. Langford and T. Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In *NIPS*, 2008.

[17] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, 2010.

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

[19] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan. Scene: a scalable two-stage personalized news recommendation system. In *SIGIR*, 2011.

[20] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *IUI*, 2010.

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

[22] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *WWW*, 2009.

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

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

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

[26] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. In *ECML*, 2005.

[27] Y. Zhang, J. P. Callan, and T. P. Minka. Novelty and redundancy detection in adaptive filtering. In *SIGIR*, 2002.