Online Ranking with Constraints: A Primal-Dual Algorithm and Applications to Web Traffic-Shaping

Parikshit Shah   Akshay Soni   Troy Chevalier

{parikshit,akshaysoni,troyc}@yahoo-inc.com

Yahoo Research

ABSTRACT

We study the online constrained ranking problem motivated by an application to web-traffic shaping: an online stream of sessions arrive in which, within each session, we are asked to rank items. The challenge involves optimizing the ranking in each session so that local vs. global objectives are controlled: within each session one wishes to maximize a reward (local) while satisfying certain constraints over the entire set of sessions (global). A typical application of this setup is that of page optimization in a web portal. We wish to rank items so that not only is user engagement maximized in each session, but also other business constraints (such as the number of views/clicks delivered to various publishing partners) are satisfied.

We describe an online algorithm for performing this optimization. A novel element of our approach is the use of linear programming duality and connections to the celebrated Hungarian algorithm. This framework enables us to determine a set of *shadow prices* for each traffic-shaping constraint that can then be used directly in the final ranking function to assign near-optimal rankings. The (dual) linear program can be solved off-line periodically to determine the prices. At serving time these prices are used as weights to compute weighted rank-scores for the items, and the simplicity of the approach facilitates scalability to web applications. We provide rigorous theoretical guarantees for the performance of our online algorithm and validate our approach using numerical experiments on real web-traffic data from a prominent internet portal.

1 INTRODUCTION

This paper investigates the online constrained ranking problem — a collection of sessions arrive in an online manner. In each session we wish to make optimal decisions that balance local versus global trade-offs: locally we wish to maximize the engagement of that session, while globally we wish to satisfy certain constraints over the entire set of sessions. Each session involves a user interacting with a collection of items, and the decision-making task at hand is that of selecting a ranking of these items. The rankings chosen across all sessions influence the rewards collected in the sessions as well as whether or not the overall constraints are satisfied.

While our main focus in this paper is to develop algorithms for the general online constrained ranking problem, the key motivating application involves webpage layout optimization and traffic-shaping. In this context, each session corresponds to a user interacting with a webpage (for example a web portal), and the decision-maker wishes to optimize the placement of content from an assortment of content (e.g. news articles, sports articles, ads, blogs, financial news, etc.).

The webpage is assumed to be divided into slots, and the problem of assigning content to slots may be viewed as an assignment or ranking problem. The natural locally optimal approach to optimizing a webpage involves computing an engagement score (e.g., a click probability or a predicted dwell-time) for each content, and displaying the content within slots on the page in descending order of the scores so that the highest (predicted) engaging content is in the most highly clicked slot, and so on.

However, in practice there are often other business constraints in place. For example we may wish to deliver a certain guaranteed number of clicks to different segments of the traffic. Business considerations may dictate that a certain number of clicks be delivered to a premium advertiser, or that a certain number of top-ranked page-views be delivered to certain premium news-agency partners over the entire collection of sessions in the traffic. These additional side constraints introduce trade-offs; performing a simple "locally optimal" ranking may violate these constraints. The challenge thus is to pick assignments in each session, online, in a way that satisfy the traffic-wide business constraints while maximizing user engagement. Since the traffic distribution is quite well-known in advance (and data is available for the same), the natural question to ask is whether we can optimize the pages in each session to maximize user engagement while satisfying the other traffic constraints. Our paper deals with developing a principled approach to deal with this traffic-shaping problem.

Note that our paper does not seek to address the learning-to-rank problem [8, 10], i.e. the problem of learning how to predict user engagement from data. Indeed, in many practical systems, a natural separation of concerns is assumed between learning-to-rank (or machine learning for predicting other user-engagement signals such as click-through rate, dwell-time etc.) and the traffic-shaping problem. The machine learned models are used as input signals to a *federation* layer that performs traffic-shaping, i.e. ingesting the signals and optimizing the ranking to maximize engagement and satisfying the constraints. However, the algorithm for traffic-shaping itself is completely agnostic to the details of how these signals are learned.

Comparison to related work: The traffic-shaping and click-shaping problems were studied by Agarwal et al. [1–3] using an optimization based framework that involved probabilistic sampling.

Several important features distinguish our work. The aforementioned work deals only with the single slot case (i.e. each session involves choosing a single item from a collection and picking the best one that satisfies a constraint), and they defer the multi-slot problem as an open problem. While they also adopt a linear programming approach, their model is substantially different; their optimization variables are probabilities of sampling items that maximize reward and satisfy traffic constraints. Moreover, they do not present theoretical guarantees on the solution quality. In contrast, our approach directly addresses the multi-slot situation (i.e. ranking items), our optimization involves optimizing over the set of permutations (i.e. assignments), and we tackle the online aspect of the problem by incorporating a learning phase followed by an online decision-making phase. We evaluate our method empirically and provide rigorous theoretical guarantees.

Our work draws on ideas and tools from the online optimization literature, specifically the primal-dual method for online linear programming [4] and its applications to online combinatorial optimization problems [7] such as the ad-words problem [12, 14], whole-page optimization problem for ads [13], combinatorial auctions [5], and extensions to the nonlinear case [11]. The aforementioned ideas have been used in online matching of ads to bidders under budget constraints – whereas our approach substantially uses ideas and tools from this literature, a key distinction is the nature of the decision-making problem. Each online session in the aforementioned lines of work involves a binary decision-making problem, i.e. whether or not to assign a unit of an item in the session. In contrast, in our problem, each session involves choosing a permutation (more specifically a perfect matching between the items and the slots).

Moreover, unlike [4] where the decision-maker must satisfy budget (upper bound) constraints, in our paper we have allocation (lower bound) constraints wherein a certain number of units (for e.g. clicks) must be delivered. Finally, we mention that within the literature, different lines of work arise from different distributional assumptions made on the sessions; e.g. [14] analyze the problem in the setting where the ordering of sessions may be adversarial with respect to the online decision maker. Another line of work [4, 12] assumes the *random permutation model*, wherein the distribution order of sessions may be assumed to exchangeable. Different theoretical guarantees are possible under different scenarios, and our work employs the random permutation model also. Our main contributions in this paper are the following:

- **Solution Form:** We present a new algorithm for the online constrained ranking problem. In the initial learning phase, dual prices λ_t corresponding to each traffic constraint t are learned. In the subsequent online phase, in each session, one is provided matrices C and A_t (see Sec. 2 for details) that capture the predictions for engagement and traffic-shaping units contributed for different possible rankings. The online ranking algorithm assumes the simple form:

$$\sigma = \texttt{MaxWeightMatching}\left(C + \sum_{t=1}^{T} \lambda_t A_t\right). \qquad (1)$$

When $\lambda_t = 0$, one recovers the "greedy" solution, i.e. that of optimizing each session individually and disregarding the constraints. In order to satisfy the constraints, the session rankings must be re-optimized (over the space of permutations) as per (1). As a consequence of the connection to maximum-weight matching, we intimately use the Hungarian algorithm [16] and its analysis.

- **Scalability:** In the learning phase, data is collected and a single linear program needs to be solved offline to obtain the prices. Thereafter, the online aspect is extremely scalable as it involves computing a matching using the prices.

- **Evaluation and Guarantees:** We present detailed empirical validation of our algorithm on real data. We also theoretically analyze our algorithm using the primal-dual method. Specifically we show that our algorithm achieves a competitive ratio of $1 - O(\epsilon)$, where ϵ is the fraction of samples that are used in the learning phase.

- **Flexibility:** Another desirable feature of our algorithm is that it is flexible; indeed constraints can change, performance of the machine learned input signals can vary, and we would like a solution that is robust. Robustness is achieved by simply repeating the learning phase (with the new constraints in place, for instance), and obtaining the new prices. Thereafter, the implementation of the online algorithm remains the same, only the new prices need be propagated.

Notation: Throughout the paper, we will use the notation $[m] := \{1, \ldots, m\}$; this quantity will refer to the size of the list (of documents) to be ranked in each session. A permutation (or ranking), denoted by σ refers to a bijection between sets $\sigma : [m] \to [m]$. Usually the sets of interest are the set of documents (in a fixed session), and the different slots, and a permutation σ specifies the assignment of documents to slots. We will use the terms permutation, assignment, and ranking interchangeably throughout the paper. It will often be convenient to represent the ranking σ by a permutation matrix $P \in \mathbb{R}^{m \times m}$ where $(P)_{ij} = 1$ if $\sigma(i) = j$ and 0 otherwise. Such a matrix has a single entry equal to one in each row and column, with the remaining entries equal to zero. We will refer to the set of $m!$ permutation matrices as $\mathcal{P}$. The convex hull of $\mathcal{P}$, denoted by $co(\mathcal{P})$ is the Birkhoff polytope [16], which will play an important role.

Throughout the paper, the index k is reserved for sessions, i.e. $k \in [n]$, the index t is reserved for the T different traffic-shaping constraints, i.e. $t \in [T]$. In a session k, if document d is shown in position p, a user engagement of $(C_k)_{d,p}$ is assumed to be accrued. Given a permutation σ, the overall engagement reward is assumed to be additive, i.e. given by $\sum_{i=1}^{m} (C_k)_{i,\sigma(i)}$. The matrix C_k is conveniently represented by a bipartite graph with m nodes on the left and right and the edges having weights $(C_k)_{d,p}$. A matching σ achieves a weight of $\sum_{i=1}^{m} (C_k)_{i,\sigma(i)}$. A maximum-weight perfect matching is a permutation $\sigma \in \mathcal{P}$ with maximum possible weight. We remind the reader that a maximum-weight perfect matching in a bipartite graph is obtained via the Hungarian algorithm [16]. We will use the notation $P \geq 0$ to specify that the matrix P is entry-wise non-negative, and P' denotes the matrix transpose. We will use $\langle P, Q \rangle := \sum_{i=1}^{m} \sum_{j=1}^{m} P_{ij} Q_{ij}$ to denote the inner-product between the matrices $P, Q \in \mathbb{R}^{m \times m}$. When a ranking P is chosen,

the engagement corresponding to that ranking is given by $\langle C_k, P \rangle$ due to the additive nature of the reward.

2 FORMULATION

Let $k \in [n]$ index a collection of sessions where different users interact with the content-delivery system of interest. In each session, a collection of m documents are made available. The task of the ranking system is, in each session k, to rank the documents, i.e. produce a permutation $\sigma_k : [m] \rightarrow [m]$.

For each session, we assume access to predictions of how each document would perform with respect to user engagement (e.g. clicks or dwell time) if document d was shown in the pth position. Let $C \in \mathbb{R}^{m \times m}$ be the matrix such that $C_{d,p}$ models the engagement when document d is shown in the pth position. (Typically, this type of information is available from the output of a machine-learned model that predicts the engagement for each document, personalized for each user corresponding to the session in question). For each session, the engagement is given by $\sum_{p=1}^{m} C_{p,\sigma(p)} = \langle C, P \rangle$, where P is the permutation matrix corresponding to σ. The total engagement across all sessions is therefore $\sum_{k=1}^{n} \langle C_k, P_k \rangle$, where k is the user engagement for the kth session and P_k is the corresponding permutation chosen.

Simultaneously, we assume that a number of traffic-shaping constraints are present. As an example, we may have constraints on the number of clicks that must be delivered to various publishing partners (in a fixed number of sessions). These can be captured as:

$$\sum_{k=1}^{n} \langle A_{kt}, P_k \rangle \geq b_t,$$

where the index k refers to the session, and t refers to t^{th} traffic-shaping constraint. The matrix $A_{kt} \in \mathbb{R}^{m \times m}$ is the matrix whose (d,p) entry captures the number of engagement units (e.g. clicks) delivered for the tth constraint in the kth session when document d is shown in position p (such estimates are also typically obtained as the output of a machine-learned click model)[1]. The constant b_t is the number of clicks that are committed to the publishing partner in n sessions contractually. The objective of the ranking system is to maximize the overall user-engagement while satisfying certain constraints.

The above description suggests a natural optimization formulation:

$$\begin{aligned} \underset{P_1, \ldots, P_n}{\text{maximize}} \quad & \sum_{k=1}^{n} \langle C_k, P_k \rangle \\ \text{subject to} \quad & \sum_{k=1}^{n} \langle A_{kt}, P_k \rangle \geq b_t \quad t = 1, \ldots, T \\ & P_k \in \mathcal{P} \quad k = 1, \ldots, n. \end{aligned} \quad (2)$$

---

[1]As an aside, we note that in practice, C_k and A_{kt} are predictions obtained from a machine learned model, and the engagement (i.e., dwell-time and clicks) realized will likely be different from the predictions. We ignore this distinction in this paper, and make the simplifying assumption that the predictions are "perfect". When the models are consistent and a sufficiently large number of sessions are involved, the expected performance (being optimized here) will be close to that of the realized one.

While conceptually clean, working directly with this formulation has several disadvantages:

- Solving this formulation requires knowledge of the C_k, A_{kt} beforehand, whereas the problem is online in nature (i.e. the sessions arrive in an online manner).
- The above optimization formulation is combinatorial and therefore intractable, i.e. it involves optimizing over the set of permutations.

To address the last point, we perform a *convex relaxation* [6, Chapter 6] of the problem, i.e. relax the optimization constraints $P_k \in \mathcal{P}$ to $P_k \in co(\mathcal{P})$, where the $co(\mathcal{P})$ represents the convex hull of $\mathcal{P}$ [16, Chapter 18]. The resulting relaxation converts the optimization formulation (2) into a convex optimization problem (indeed, it is a linear programming problem), which can therefore be solved (in principle) in polynomial time. There are a number of difficulties pertaining to the relaxation:

- Linear programming is computationally intensive and solving one at serving-time is often infeasible due to the strict latency requirements. Hence we desire to avoid solving them at run-time (although solving offline on a periodic basis is acceptable).
- A problem associated with convex relaxations of combinatorial problems is that the resulting solution may be fractional (i.e. the solutions P_k may not be permutation matrices). We then have to devise a scheme to convert the optimal solution to a permutation matrix via a rounding scheme. As we will show in our subsequent analysis, we will not need to round solutions. The special structure of our problem and well-known results from graph matching and the analysis of the Hungarian algorithm [16, Chapter 17] actually guarantee that we can always produce a valid extremal solution, rather than a fractional one.
- In order to solve the convex relaxation, one needs a tractable representation of the set $co(\mathcal{P})$ in terms of a small number of equations and inequalities. It turns out that the convex hull of the set of permutation matrices has a compact description via the Birkhoff-von Neumann Theorem [16, Chapter 18], it is simply the set of so-called doubly stochastic matrices (sometimes also called the Birkhoff polytope) given by:

$$\mathcal{B} := co(\mathcal{P}) = \left\{ P \in \mathbb{R}^{m \times m} \mid P\mathbf{1} = \mathbf{1} \; P'\mathbf{1} = \mathbf{1} \; P \geq 0 \right\}. \quad (3)$$

We will be able to resolve the above difficulties by computing and analyzing the dual; indeed the analysis will reveal a natural and simple online algorithm which we will describe below.

We first explicitly state the primal and dual linear programs that result from the convex relaxation of (2).

$$\begin{aligned} \underset{P_1, \ldots, P_n}{\text{maximize}} \quad & \sum_{k=1}^{n} \langle C_k, P_k \rangle \\ \text{subject to} \quad & \sum_{k=1}^{n} \langle A_{kt}, P_k \rangle \geq b_t \quad \forall \, t = 1, \ldots, T \\ & P_k \in \mathcal{B} \quad k = 1, \ldots, n \end{aligned} \quad (4)$$

$$\text{maximize}_{\lambda,\alpha_k,\beta_k} \quad \sum_{t=1}^{T} \lambda_t b_t + \sum_{k=1}^{n} \alpha'_k \mathbf{1} + \sum_{k=1}^{n} \beta'_k \mathbf{1}$$

$$\text{subject to} \quad C_k + \sum_{t=1}^{T} \lambda_t A_{kt} + \mathbf{1}\alpha'_k + \beta_k \mathbf{1}' \leq 0 \ \ \forall k = 1, \ldots, n$$

$$\lambda \geq 0$$

$$(5)$$

The dual linear program is of key interest in the paper — our algorithm will consist of solving (5) on historical data, and computing the optimal dual variables λ which have a natural price interpretation. We describe this in the next section.

3 THE ALGORITHM

Recall that in our setup, the n sessions arrive in a sequential manner online, and for each session k we have access to the matrices C_k (the engagement), and A_{tk} (corresponding to the traffic-shaping constraint). We denote the full set of sessions by N. Our approach involves using a subset of the sessions, S, initially to learn the dual prices. The corresponding primal/dual linear programs are called the *sample linear programs*. In these learning-phase sessions, we make arbitrary decisions — concretely we let $P_k = I$, the identity permutation. Let $\hat{n} := |S|$ be chosen and denote $\epsilon = \frac{\hat{n}}{n}$. Since the constraints require that a budget of b_t clicks are to be delivered over n sessions, it follows that when ϵn sessions are present, the corresponding commitment is only ϵb_t clicks for each traffic-constraint b_t. In this way, the b_t is rescaled. In order to be conservative, we instead require that for each constraint: $\sum_{k=1}^{\hat{n}} \langle A_{kt}, P_k \rangle \geq \nu \epsilon b_t$. The additional ν multiplicative factor ensures that the solution obtained is robust to random variations (we will discuss this in Section 5), and also to counter the conservative estimate that since the first ϵ fraction of sessions are used for learning, zero (or negligible) contributions to the constraints are made from it. Accordingly, the primal/dual sample linear programs are:

$$\text{maximize}_{P_1, \ldots, P_{\hat{n}}} \quad \sum_{k=1}^{\hat{n}} \langle C_k, P_k \rangle$$

$$\text{subject to} \quad \sum_{k=1}^{\hat{n}} \langle A_{kt}, P_k \rangle \geq \nu \epsilon b_t \ \ \forall t = 1, \ldots, T$$

$$P_k \in \mathcal{B} \ \ k = 1, \ldots, \hat{n}$$

$$(6)$$

$$\text{maximize}_{\lambda,\alpha_k,\beta_k} \quad \sum_{t=1}^{T} \nu \epsilon \lambda_t b_t + \sum_{k=1}^{\hat{n}} \alpha'_k \mathbf{1} + \sum_{k=1}^{\hat{n}} \beta'_k \mathbf{1}$$

$$\text{subject to} \quad C_k + \sum_{t=1}^{T} \lambda_t A_{kt} + \mathbf{1}\alpha'_k + \beta_k \mathbf{1}' \leq 0 \ \ \forall k = 1, \ldots, \hat{n}$$

$$\lambda \geq 0$$

$$(7)$$

In the theoretical anaylsis section we prove that when $\nu = 1 + 4\epsilon$, with high probability a feasible solution is returned. We also prove that when $\nu = 1 - \epsilon$, an almost feasible solution is returned, and that the objective is almost as large as the hindsight optimal feasible solution.

Algorithm 1 Algorithm for optimal traffic-shaping

1: **Input:** Data $\{(C_k, A_{k1}, \ldots, A_{kT})\}$ for sessions $k = 1, \ldots, n$ presented online for ranking, sample linear program parameters ν, ϵ.
2: For the first $\hat{n}$ sessions, let $P_k = I$ (arbitrary decisions) and log the session data $\{(C_k, A_{k1}, \ldots, A_{kT})\}$ corresponding to this sample set $k \in S$.
3: Solve the sample linear program (7) on S to obtain the dual prices $\hat{\lambda}_1, \ldots, \hat{\lambda}_m$.
4: **while** $\hat{n} < k \leq n$ **do**
5:     Compute document score matrix $S_k = C_k + \sum_{t=1}^{T} \hat{\lambda}_t A_{kt}$ where $S_{d,p}$ represents the score when document d is shown in position p.
6:     Set $\sigma_k := \texttt{MaxWeightMatching}(S_k)$.
7:     **Output:** Ranking σ_k for session k.
8:     Increment session counter $k := k + 1$
9: **end while**

In words, the algorithm works as follows. First, using historical data, we offline solve the linear program (7) to obtain the optimal dual prices $\hat{\lambda}_t$ for each traffic shaping constraint. These prices are then made available to the online serving system for ranking. Once the learning phase is over, upon the arrival of new sessions, the matrices C_k, A_{kt} are computed, and the score matrix S_k is computed as per step 5. As mentioned before, the (d,p) entry of the matrix S_k represents the score when document d is shown in position p.

Once the score matrix is computed, we call a subroutine to compute the maximum-weight bipartite perfect matching with respect the weight matrix S_k, this perfect matching assigns documents to positions, and thereby yields the desired ranking σ_k for that session.

The run-time complexity is essentially the complexity of computing a perfect matching, for which a number of options are available. In order to compute a max-weight perfect matching exactly, one may use the celebrated Hungarian algorithm. The computational complexity of implementing the same is $O(m^3 \log m)$ [16]. In scenarios of interest to us, we are typically only interested in ranking of the order of $20 - 200$ webpages in each session, and hence the algorithm is feasible to implement at run-time. When the scale is larger and the Hungarian algorithm is not a viable option, one can instead implement the naive greedy algorithm[2] as an alternative. The naive greedy algorithm is known to be a $\frac{1}{2}$-optimal algorithm for maximum matching, with a worst-case running time of $O(m^2)$. In our experiments, we use a position effects model called the "reference CTR" model, in which the greedy algorithm is actually optimal. The offline computational complexity is polynomial in the size of the problem — it involves the solution of the linear program (7), which scales with the sample size of the sampled linear program and is hence high. However, since offline computation is not a bottleneck, our approach is scalable to web applications.

While Algorithm 1 advocates choosing the first ϵ fraction of the traffic for solving the sampled linear program once (in advance)

---

[2]The greedy algorithm is simply the following: sort the edges of the bipartite graph in decreasing order. We maintain a set of edges M (initialized to be empty), and loop over the sorted edges. In each step, add the corresponding edge to M if it is possible to do so while maintaining the requirement that M is a (possibly partial) matching.

on historical data, this is mostly to simplify the analysis of the algorithm (in our experiments in the next section where data for 10^6 sessions were collected, we found $\epsilon = 0.001$ to work quite well for learning the correct prices). In practice, we recommend solving the dual linear program periodically (e.g. once daily) on all the historical data collected in that time period to refresh the prices to capture the random as well as seasonal variations in traffic. The analysis of periodically recomputing the solution is also possible (in the spirit of [4]), but the analysis is more complicated with only marginally better bounds — we leave this for future work.

4 NUMERICAL EXPERIMENTS

4.1 Experimental Setup

In this section we validate our algorithm via numerical experiments. Our numerical experiments are conducted on real data collected from session history on a major web-portal. We describe the setup for our experiment below.

Our data consists of 2000 sessions of web-traffic data. In each session, we are presented with 20 documents to rank on the web-page. Each document has associated with it the following attributes:

(1) The predicted dwell-time: This is a score generated by a machine-learned model on a large corpus of data that incorporates document features, user features, and document popularity signals. The predicted score is to be interpreted as a surrogate for the expected dwell-time when the user is shown that document.
(2) The click-through rate: This is again generated via a machine learned model, and is an estimate for the click-probability for a particular user to click on the document in question when that document is shown in the first position.
(3) Newsiness score: This is a normalized score that captures how newsy a particular document is. It is also obtained as the output of a machine-learned model. When a newsy article is shown in a highly clickable slot on the page, the overall page is deemed more newsy.
(4) Publisher A: This is a binary $\{0, 1\}$ score that captures whether a document is curated by Publisher A or not. Showing a Publisher A-curated document in a more clickable slot delivers more clicks to that publisher, and is hence more desirable to it.
(5) Publisher B: This is also a binary score (as above) corresponding to whether document is curated by Publisher B.

In our experiments, we implement Algorithm 1, i.e. use a subset of the data for learning the dual prices (making arbitrary decisions in the process), and in the remaining sessions making the optimal decisions with respect to the learned dual prices. In our experiments, the online optimization problem involves: (a) maximizing the dwell-time in each session, (b) delivering a fixed number of clicks to Publishers A and B, (c) achieving at least a certain average newsiness score over the entire set of sessions.

One of the key requirements of our framework is that we need the dwell-time, click probability, and newsiness score for each document when it is shown in any one of the 20 slots. The machine learned models generate these scores with respect to the first slot only. In order to address this, we utilize the (somewhat standard) Reference CTR model, which is explained below.

4.2 The Reference CTR

When users are presented with a list of content on a web-page, it is well-known [9] that there is a strong *position effect* for clicks, dwell-time, and other engagement metrics (such as the newsiness score of interest to us in this paper). It is commonly assumed [9], that a document d, when shown in slot p, has a CTR that depends on its CTR in position 1 (denoted by $c_1(d)$) and a factor that depends only on the position (denoted by ref_p). It is normalized so that $\text{ref}_1 = 1$. Hence the predicted CTR of document d in position p is given by:

$$c_p(d) = c_1(d) \times \text{ref}_p.$$

The quantity ref_p is computed by computing the aggregate CTR for each position, and normalizing by the aggregate CTR for position 1 [9]. We will also assume (for this experiment) that the dwell-time and newsiness have a similar position effect as above.

For a list of items presented to a user according to a permutation σ, the total number of (predicted) clicks generated by the session, and the clicks generated for publisher t (where $t \in \{A, B\}$) are respectively given by:
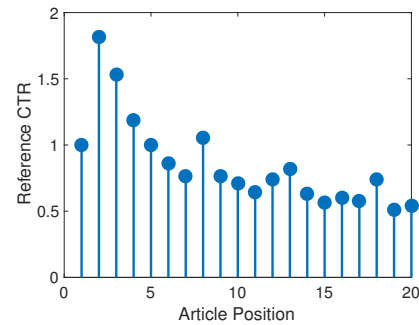
$$C(\sigma) = \sum_{p=1}^{m} c_p(\sigma(p)) \times \text{ref}_p$$

$$C_t(\sigma) = \sum_{p=1}^{m} c_p(\sigma(p)) \times \text{ref}_p \times \mathbf{1}\left(\sigma(p) \text{ of traffic type } t\right).$$

Similarly, we assume that the total dwell-time and newsiness respectively are:

$$D(\sigma) = \sum_{p=1}^{m} \text{dwell}(\sigma(p)) \times \text{ref}_p, \qquad N(\sigma) = \sum_{p=1}^{m} \text{news}(\sigma(p)) \times \text{ref}_p.$$

Figure 1 shows the reference CTR distribution.

Figure 1: The reference CTR distribution for the slots on the page. Interestingly, the reference CTR distribution is not monotonically decreasing with the slot position due to various page presentation effects.

4.3 Observations

We first investigate the performance of sampled linear program when the size of the sample set increases to 2000 sessions (entire dataset). Figure 2a shows how the average (normalized) dwell-time per session varies as we increase the size of the sampled linear program. Note that when all 2000 sessions are used, we essentially

obtain the hindsight-optimal solution. As Fig. 2a shows, this level of performance is approximately achieved by the sampled linear program within just 800 sessions, justifying our intuition that this quantity is distributionally stable with respect to the traffic.

Figure 2b shows how the dual prices change as a function of the number of sessions over which the sampled linear program is solved. Again, we see that the prices stabilize at approximately 800 sessions.

In Fig. 2c we investigate the trade-offs that are inevitable as a consequence of traffic-shaping. We fix the number of sessions to be 2000 and solve the corresponding linear program (i.e. the hindsight optimal solution). We fix the values of the constraints for clicks delivered to publisher B and the newsiness constraint. We vary the number of clicks deliverable to publisher A by changing a parameter θ (as θ increases, the corresponding click constraint commitment b_t increases multiplicatively). We study how the dwell-time per session changes as θ increases — the corresponding trade-off curve is shown in Fig. 2c.

In Fig. 2d we study the performance of the online algorithm. On the x-axis is the fraction of the 2000 sessions which were used for the sampled linear program. On the y-axis is the performance ratio for different metrics. A value of $\nu = 1.05$ (as required in (7)) was chosen for this experiment. The solid-blue curve is the competitive ratio (i.e. the ratio of the performance by the online algorithm to that of the hindsight optimal solution, see Sec. 5). Note that in the first ϵ fraction of the sessions (the price-learning phase), arbitrary suboptimal decisions are made, and hence the competitive ratio decreases as a function of ϵ (see Theorem 5.1). The yellow, purple, and green plots are the ratio of the clicks/newsiness delivered versus the value committed by the constraints (i.e. the b_t), and values under one indicate constraint violation. Note that for small values of ϵ, some of the constraints are actually violated — hence while the competitive ratio is high, training on that small subset of sessions yields infeasible solutions. The red dotted line indicates the online performance ratio as a function of ϵ. The online performance ratio is the ratio of the reward of the online algorithm to that of the hindsight optimal sessions, *restricted to the segment of the sessions where online decisions are made.* While the online performance ratio is close to 1 throughout, it attains this ratio by violating constraints at small values of ϵ.

Another interesting feature that we found was that, under the RefCTR model, the greedy algorithm (i.e. sorting documents by the price-weighted scores, and then matching them to positions in sorted order of their RefCTR value) achieve the *same ranking* compared to the Hungarian algorithm. This is simply a structural consequence of the RefCTR model (we omit the proof here), however the scalability implications are substantial; instead of implementing the Hungarian algorithm, the serving implementation only needs a sorting procedure.

We finally make a comment concerning comparing our approach to baseline approaches. To the best of our knowledge, there is no other work addressing the multi-slot traffic-shaping problem in the literature that would form a natural baseline to compare our approach with. Certain naive baselines that can be considered are:

- No traffic-shaping: we can simply apply the greedy approach and sort the sessions based on engagement scores only. Of course this is naive, and will likely violate the shaping constraints.

- A natural other baseline is to sort based on a weighted sum of the engagement scores and the constraint quantities. In the most general setting, this will be suboptimal compared to our approach (this approach is tantamount to the "greedy" approach for weighted matching, whereas the Hungarian algorithm yields the optimal solution). In the case of the RefCTR model considered in this section, the greedy approach actually yields optimal solutions when the weights are chosen judiciously. A typical heuristic to choose the weights is to perform a grid search over the weights (in which case the optimal weights will be learned), the main disadvantage being that this approach is not scalable when there are many competing constraints. Indeed, one interpretation of our work is that it provides a theoretical foundation and efficient approach for applying this popularly used weight-based heuristic.

5 THEORETICAL GUARANTEES

In order to describe the performance guarantees, we need to define a critical (and well-known) solution concept related to online algorithms, namely the notion of the competitive ratio of the algorithm.

Given online events $\{(C_k, A_{k1}, \ldots, A_{km})\}$ for sessions $k \in [n]$, let OPT denote the solution obtained by solving the linear program (4) on this data — we call the solution thus obtained the *optimal hindsight solution.* A hypothetical decision-maker with advance knowledge of the "future" sessions would make these optimal decisions, and this is the best performance achievable by any algorithm. Suppose Algorithm 1 (which is online) achieves a performance of $\widehat{\text{OPT}}$. We define the *competitive ratio* of the algorithm to be $\frac{\widehat{\text{OPT}}}{\text{OPT}}$. The competitive ratio measures how the performance of the online algorithm compares to the optimal hindsight solution. Our main result shows that, under certain assumptions, our algorithm achieves an $1 - O(\epsilon)$ competitive ratio. We state our assumptions next:

(1) **Assumption 1:** The sessions $\{(C_k, A_{k1}, \ldots, A_{kT})\}$ arrive in a random order, i.e. the matrices $\{(C_k, A_{k1}, \ldots, A_{kT})\}$ can be arbitrary, but every permutation of the indices k have an equal probability.

(2) **Assumption 2:** The optimization problems (4) and (6) are both strictly feasible.

(3) **Assumption 3:** The total number of sessions n is known a priori.

(4) **Assumption 4:** For every permutation P we have $\langle C_k, P \rangle \in [0, 1]$ and $\langle A_{kt}, P \rangle \in [0, 1]$ for all $k = 1, \ldots, n$ and $t = 1, \ldots, T$.

THEOREM 5.1. *Consider the online traffic-shaping problem when the above assumptions hold true, and let $\epsilon \in (0, \frac{1}{3})$ be fixed. Define $B := \min_{t=1,\ldots,T} b_t$. Suppose the constraints satisfy the requirement that*

$$\epsilon \geq \left(\frac{\log\left(\frac{2C_0 T}{\epsilon}\right) + (T+1)\log\left(m^2 n\right)}{B} \right)^{\frac{1}{3}}. \quad (8)$$

Let the dual prices computed by the sampled linear program be denoted by $\hat{\lambda}$. Then the solution constructed by Algorithm 1, denoted by $P_k(\hat{\lambda})$, has the following properties:
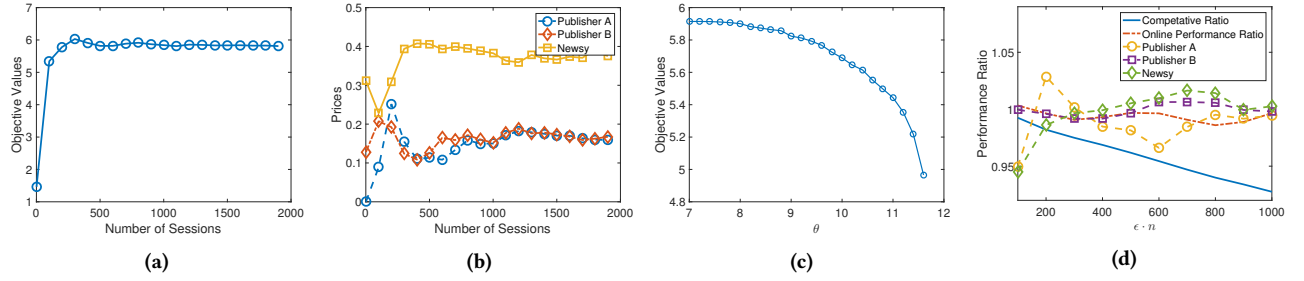
Figure 2: (a) Average Dwell-time per session as a function of the sample size for the sampled linear program. The dwell-time per session stabilizes at approximately 800 **sessions and is constant thereafter. (b) Dual prices for each of the constraints, as a function of the sample size for the sampled linear program. The dual prices also stabilize at approximately** 800 **sessions and are constant thereafter. (c) Trade-off between dwell-time per session and clicks delivered to Publisher A. (d) Different performance ratios as a function of** ϵ**.**

(1) *In each session the optimal hindsight solution and the online approach via Algorithm 1 produce valid assignments P_k^* and $P_k^{online}(\hat{\lambda})$. Moreover, if the optimal hindsight prices, denoted λ^*, are given 1, then $P_k^* = P_k^{online}(\lambda^*)$.*

(2) *When $\nu = 1 + 4\epsilon$, With probability exceeding $1 - \epsilon$, the assignments satisfy the traffic-shaping constraints, i.e.*

$$\sum_{k=1}^{n} \langle A_{kt}, P_k(\hat{\lambda}) \rangle \geq b_t \quad \forall t = 1, \dots, T$$

(3) *When $\nu = 1 - \epsilon$, with probability exceeding $1 - \epsilon$ the online solution satisfies*

$$\sum_{k=1}^{n} \langle A_{kt}, P_k(\hat{\lambda}) \rangle \geq (1 - 2\epsilon)b_t \quad \forall t = 1, \dots, T$$

and objective attained by the online assignments $P_k(\hat{\lambda})$ satisfies:

$$\sum_{k=1}^{n} \langle C_k, P_k(\hat{\lambda}) \rangle \geq (1 - \epsilon)OPT,$$

(i.e. the competitive ratio is $1 - O(\epsilon)$ with high probability).

We give a detailed proof in Sec 7. We make the following remarks about the main result:

Remarks

(1) Note that the first part states that if the optimal prices are available to the online algorithm, then optimal decisions will be made in each session. Hence the prices λ^* are sufficient to make good decisions online. The second and third parts of the theorem quantify the intuition that if we use a small set of samples initially to learn the prices using the sampled linear program, the online algorithm will then perform almost optimally.

(2) Note that Algorithm 1 advocates using the Hungarian algorithm to compute the best matching in each session. Depending on the number of items to be ranked, this may or may not be feasible at run time — in the latter case a greedy algorithm may be used. The greedy algorithm, being a $\frac{1}{2}$-approximation in the worst case, can also be analyzed and bounds on the competitive ratio may be derived. In our experiments, we find that using the greedy algorithm does not

lead to any loss of performance as a consequence of the RefCTR assumption (see Sec. 4).

(3) Assumption 4 requires that the for each permutation, the rewards and the influence to each traffic shaping is non-negative and bounded by 1. The non-negativity assumption is natural in our setting (since we are interested in quantities such as views and clicks). The upper bound of 1 is somewhat arbitrary, we merely need the rewards and influences to be bounded by some uniform quantity (say U). Our results would be more or less unaffected (modulo certain logarithmic factors in the statement of Theorem 5.1) if $U > 1$).

(4) We now comment on the qualitative relationship between ϵ, B and the competitive ratio. In order to make sense of the statement, let us study the setting when B increases linearly with the number of sessions n (note that this is the setting of most interest in practical applications; when traffic-size doubles, we hope to be able to commit twice as many clicks to a publishing partner). In this regime, $B = \rho n$ and as n increases, the right hand side of (8) decreases to zero and we may make ϵ arbitrarily small, thus yielding (a) a small fraction of the samples used for learning the prices (b) a better competitive ratio, and (c) a high probability of success simultaneously. Indeed (8) may be viewed as a "sample-complexity" type result.

Note that B growing super-linearly with respect to the number of sessions n would be problematic from a feasibility point of view – if we over-commit to a traffic-shaping constraint the optimization problem becomes infeasible violating Assumption 2).

The situation when B is small (e.g. $o(n)$) is a little more subtle. In this situation, the optimization problem itself is "easier" in the sense of finding a feasible solution. However, to get *multiplicative* bounds on the competitive ratio our proof techniques do need B to be large enough so that (8) is satisfied. The fact that we ask for multiplicative bounds (as opposed to additive ones) together with the fact that we make no distributional assumptions on the traffic beyond random arrival requires us to use a particular concentration inequality in the proof which in turn requires B to satisfy this condition. It is an interesting open problem to see whether this condition can be weakened.

(5) The competitive ratio decreases as the fraction of sessions ϵ used for learning increases. While on the one hand the prices are

learned more accurately as ϵ increases, on the other hand suboptimal decisions are made by the online algorithm in the learning phase. Indeed, we assume (somewhat pessimistically) that arbitrary decision are made in the learning phase with zero reward accrued, thus an ϵ-fraction of the possible reward is lost in this phase.

(6) While our formulation (4) only considers lower bound (satisfaction) constraints, our approach is amenable to upper bound (budget) constraints as well. As an example, if we wanted to optimize page-placement jointly for content and ads, with ads having budget constraints, we would encounter formulations with both upper and lower bound constraints. The corresponding linear inequalities can be incorporated into the optimization formulation of the sampled linear program, and the dual prices can be used as weights in weighted bipartite matching problem in the same manner. While we don't analyze this situation in this paper, we anticipate that an analogous analysis can be carried out.

6 CONCLUSION

In this paper we investigated the problem of online constrained ranking and its application to the web traffic-shaping problem. We developed a linear-programming based primal-dual algorithm for online ranking and demonstrated it's efficacy on real data. We also proved rigorous guarantees about the algorithm in terms of the achieved competitive ratio.

While the paper focuses on traffic-shaping as the motivating application, we believe this algorithm to have broad applicability to whole-page optimization and beyond. For instance, when optimizing the simultaneous placement of ads and organic content, additional constraints (or modifications to the objective) can be made to factor in the revenue considerations traffic-wide. More generally, this approach can be used for solving a variety of constrained online assignment problems such as matching customers to servers (e.g. matching drivers to passengers) subject to overall service-level constraints. We believe this approach will scale well to a variety of such application domains.

A number of further avenues for future work are worth mentioning:

- Our model involves optimizing (in the objective) subject to constraints the predicted dwell-time and the predicted clicks over a period of sessions. These predictions are obtained as the output of a machine-learned model. The number of actual clicks obtained, and the actual dwell-time realized will of course be different. As a practical strategy, we advocate re-solving the linear program periodically (and suitably altering the constraints) to capture these variations. However, there is an unavoidable feedback loop between the rankings and the machine learned models — the decisions made in the ranking phase affect the learning of these online models. Consequently, there is a natural exploration-exploitation trade-off that is unavoidable in this setup. Ranking unknown items in a highly clickable slot will lead to more exploration, whereas ranking more certain items will lead to exploitation. How one navigates this will likely be of key practical importance.
- Our approach assumes that the objective and the constraints can be expressed as linear functions of the ranking;

the expected number of clicks and expected dwell time are assumed to be a weighted sum of the components derived from each slot. Other types of constraints may not be linearly expressible (e.g. suppose one wishes to enforce a certain notion of diversity in each ranking and enforce a certain minimum amount of average diversity across sessions). Working with non-linear constraints will likely enable us to enrich the quality of ranking in our traffic.

- Our approach makes weak assumptions about the traffic in order to achieve competitive ratio bounds, i.e. that the events arrive in a random order. If stronger independence assumptions are made, it is possible that stronger results might hold – for example it may be possible to prove that the optimal prices are learned and a sample complexity for the same might be characterizable.

7 PROOFS

We begin by stating two lemmas (their proofs are omitted due to space constraints) which follow as a standard consequence of the analysis of the Hungarian algorithm (we refer the reader to [16] for details).

LEMMA 7.1. *Let $C \in \mathbb{R}^{m \times m}$ be an entry-wise non-negative matrix. Consider the following optimization problems:*

$$\begin{array}{ll} \underset{P}{maximize} & \langle C, P \rangle \\ subject\ to & P \in \mathcal{B} \end{array} \tag{9}$$

$$\begin{array}{ll} \underset{\alpha, \beta}{maximize} & \alpha' \mathbf{1} + \beta' \mathbf{1} \\ subject\ to & C + \mathbf{1}\alpha' + \beta \mathbf{1}' \le 0 \end{array} \tag{10}$$

Then the following hold:

(1) *The pair* (9)*,* (10) *is a primal-dual pair, and strong duality holds.*

(2) *An optimal solution P^* to* (9) *is attained at a permutation matrix, and this solution can be found using the Hungarian algorithm.*

LEMMA 7.2. *Let $\lambda = \bar{\lambda} \in \mathbb{R}^T$ be fixed such that $\bar{\lambda} \ge 0$. Consider the solution to the optimization problem*

$$\underset{\alpha_k, \beta_k}{maximize} \quad \sum_{t=1}^{T} \bar{\lambda}_t b_t + \sum_{k=1}^{n} \alpha_k' \mathbf{1} + \sum_{k=1}^{n} \beta_k' \mathbf{1}$$

$$subject\ to \quad C_k + \sum_{t=1}^{T} \lambda_t A_{kt} + \mathbf{1}\alpha_k' + \beta_k \mathbf{1}' \le 0 \quad \forall k = 1, \dots, n \tag{11}$$

A set of optimal dual solutions (w.r.t. the constraints of (11)*) $P_k(\bar{\lambda})$ for $k = 1, \dots, n$ corresponding to* (4) *are achieved at extreme points of the Birkhoff polytope, (i.e. permutation matrices). Moreover each of the $P_k(\bar{\lambda})$ may be computed by computing a maximum-weight perfect matching with respect to the bipartite graph with weights $C_k + \sum_{t=1}^{T} \bar{\lambda}_t A_{kt}$ (using the Hungarian algorithm).*

We will also need the following lemma concerning concentration of random variables. For a proof we refer the reader to [4, Lemma A.1], [12, Lemma 3] and the references therein.

LEMMA 7.3. *Let $u_1, \ldots, u_r$ be a uniformly random sample without replacement from real numbers*

$$\mathbb{P}\left\{ |\sum_{i=1}^{r} u_i - r\bar{c}| \geq t \right\} \leq \exp\left(-\frac{t^2}{2r\sigma_R^2 + t\Delta_R} \right),$$

where $\Delta_R = \max_i c_i - \min_i c_i$, $\bar{c} = \frac{1}{R}\sum_{i=1}^{R} c_i$, and $\sigma_R^2 = \frac{1}{R}\sum_{i=1}^{R}(c_i - \bar{c})^2$.

We now provide a detailed proof of Theorem 1. Note that both our main result for online ranking, and our assumptions are of a similar flavor as existing literature in online algorithms [4, 12]. Indeed, it is known [4, 12], that in these related results, these assumptions are also tight, i.e. it is not possible to get $1 - O(\epsilon)$ competitive ratio if a single one of these assumptions is removed. While we do not show the tightness of the assumptions, we believe it likely that they are needed to achieve the stated competitive ratio.

PROOF OF THEOREM 5.1. The main idea behind the proof, which mimics [4] with suitable adaptations, will be to use the complementary slackness conditions for linear programming. Let $\hat{\lambda}$ be a dual price vector obtained as the solution of the sampled linear program. We denote by $P_k(\hat{\lambda})$ the permutation that is chosen as per steps 5, 6 of the algorithm.

Part 1. The online assignments $P_k^{\text{online}}(\hat{\lambda})$ (and similarly $P_k^{\text{online}}(\lambda^*)$) are valid assignments since they are constructed via the Hungarian algorithm. Note that they are both trivially primal-feasible (since assignments are extreme points of the Birkhoff polytope).

Let λ^* be an optimal dual solution of (5). Note that when $\bar{\lambda} = \lambda^*$, the corresponding optimal solutions of (11) coincide with those of (5). By strong duality, a set of corresponding primal solutions also coincide. By Lemma 7.2, a dual variable corresponding to the constraints of (11) achieves optimality at permutation matrices $P_k(\lambda^*)$. Hence, $P_k(\lambda^*) = P_k^*$.

Part 2. Our proof strategy will be as follows. We will take a hypothetical optimal solution $\hat{\lambda}$ to the sampled linear program. It will be assumed to satisfy that the complementary slackness conditions of the primal dual pair (9),(10) - but otherwise arbitrary. We will show that as a consequence of complementary slackness and the assumptions, with high probability the solution yielded by $P_k(\hat{\lambda})$ are feasible for this hypothetical $\hat{\lambda}$. We will then take a union bound over all possible $\hat{\lambda}$ to obtain that the solution is feasible unconditionally.

Let us fix a constraint t and note that for the first ϵn sessions, arbitrary decisions are made and we thus assume that no contributions to the constraints are made from these. As a result we required that the sampled linear program satisfy:

$$\sum_{k \in S} \langle A_{kt}, P_k \rangle \geq (1 + 4\epsilon)\epsilon b_t,$$

where the additional $(1+4\epsilon)$ factor ensures that the solution robustly satisfies the constraints.

The bad event of interest to us is the event that a constraint is violated. We will call the set S bad when the solution λ leads us to a situation of such a constraint violation. Concretely, we consider the following set relations corresponding to a set S being bad:

$$\left\{ \sum_{k \in N\backslash S} \langle A_{kt}, P_k(\lambda)\rangle < b_t, \sum_{k \in S} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+4\epsilon)\epsilon b_t \right\} \subseteq A_1 \cup A_2,$$

$$A_1 = \left\{ \sum_{k \in N\backslash S} \langle A_{kt}, P_k(\lambda)\rangle < b_t, \sum_{t \in S} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+4\epsilon)\epsilon b_t, \right.$$

$$\left. \sum_{k \in N} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+3\epsilon)b_t \right\}$$

$$A_2 = \left\{ \sum_{k \in S} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+4\epsilon)\epsilon b_t, \sum_{k \in N} \langle A_{kt}, P_k(\lambda)\rangle < (1+3\epsilon)b_t \right\}$$

Next we note that:

$$\mathbb{P}(A_1) \leq \mathbb{P}\left(\sum_{k \in N\backslash S} \langle A_{kt}, P_k(\lambda)\rangle < b_t, \sum_{k \in N} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+3\epsilon)b_t \right)$$

$$\leq \mathbb{P}\left(\sum_{k \in N\backslash S} \langle A_{kt}, P_k(\lambda)\rangle < b_t, \sum_{k \in N} \langle A_{kt}, P_k(\lambda)\rangle \geq \frac{1+\epsilon}{1-\epsilon}b_t \right)$$

$$= \mathbb{P}\left(\sum_{k \in N\backslash S} \langle A_{kt}, P_k(\lambda)\rangle < b_t, (1-\epsilon)\sum_{k \in N} \langle A_{kt}, P_k(\lambda)\rangle \geq (1+\epsilon)b_t \right)$$

$$\leq 2\exp\left(-\frac{\epsilon^2 b_t}{2-\epsilon} \right)$$

where the second inequality follows since $\frac{1+\epsilon}{1-\epsilon} < 1 + 3\epsilon$ for all $\epsilon \in (0, \frac{1}{3})$, and the last inequality follows from Lemma 7.3.

Defining $Y_k := \langle A_{kt}, P_k(\lambda)\rangle$, we also note that

$$\mathbb{P}(A_2) \leq \mathbb{P}\left(\sum_{t \in S}\langle A_{kt}, P_k(\lambda)\rangle \geq (1+4\epsilon)\epsilon b_t, \sum_{t \in N}\langle A_{kt}, P_k(\lambda)\rangle < (1+3\epsilon)b_t \right)$$

$$\leq \mathbb{P}\left(|\sum_{t \in S} Y_t - \epsilon \sum_{t \in N} Y_t| \geq \epsilon^2 b_t \right)$$

$$\leq 2\exp\left(-\frac{\epsilon^3 b_t}{2+\epsilon} \right)$$

where the last inequality again follows from an application of Lemma 7.3.

Hence, the probability of a bad sample is bounded above by $4\exp\left(-\frac{\epsilon^3 b_t}{2+\epsilon} \right) \leq \delta$, where $\delta := \frac{\epsilon}{C_0(m^2 n)^{T+1}}$

Since the solution λ obtained from the sampled linear program can be arbitrary, we take a union bound over all the possible "distinct" λ. We call two solutions λ_1 and λ_2 as distinct if $P_k(\lambda_1) \neq P_k(\lambda_2)$ for some session $k \in N$. Consider a dual constraint with complementary slackness:

$$\left(C_k + \sum_{t=1}^{T} \lambda_t A_{kt} + \mathbf{1}\alpha_k' + \beta_k \mathbf{1}' \right)_{ij} (P_k)_{ij} = 0.$$

and note that $(P_k)_{ij} > 0$ is only possible when

$$\left(C_k + \sum_{t=1}^{T} \lambda_t A_{kt} + \mathbf{1}\alpha_k' + \beta_k \mathbf{1}' \right)_{ij} \geq 0.$$

(Indeed, the set of possible solutions yielded by the Hungarian algorithm will be a subset of such possible P_k - by design the

Hungarian algorithm maintains the dual inequality as an invariant [16]). The set of possible P_k is thus bounded above by the number of unique separations of the points

$$\left\{ \left((C_k)_{ij}, (A_{k1},)_{ij}, \ldots, (A_{kT},)_{ij}, 1 \right) \right\}_{i,j,k \in [m] \times [m] \times [n]}$$

in $(T+1)$-dimensional space. By results from computational geometry [15], the number of such distinct hyperplanes, is bounded above by $C_0 (m^2 n)^{T+1}$ for some constant C_0. Applying a union bound over the possible different prices, and $t = 1, \ldots, T$ we obtain the required result.

Part 3. Given $v = 1 - \epsilon$, one can prove approximate feasibility in a similar manner as above by observing that

$$\mathbb{P}\left(\sum_{k \in N \setminus S} \langle A_{kt}, P_k(\lambda) \rangle < (1 - 2\epsilon) b_t, \sum_{k \in N} \langle A_{kt}, P_k(\lambda) \rangle \geq (1 - \epsilon) b_t \right) \leq \delta,$$

where δ is the quantity used in the preceding part.

To bound the competitive ratio, our proof strategy will be to construct an auxiliary linear program for which the online solutions are optimal. We will show that a point P_k^* (the solution to (4)) is feasible with respect to this auxiliary linear program, and hence the objective function values between the online solution and that achieved by P_k^* cannot be too different.

Let $\hat{\lambda}$ be the solution of the sampled linear program (7) and $P_k(\hat{\lambda})$ the corresponding assignment. Consider the linear program:

$$
\begin{aligned}
\underset{P_1, \ldots, P_n}{\text{maximize}} \quad & \sum_{k=1}^{n} \langle C_k, P_k \rangle \\
\text{subject to} \quad & \sum_{k=1}^{n} \langle A_{kt}, P_k \rangle \geq \hat{b}_t \quad \forall\, t = 1, \ldots, T \\
& P_k \in \mathcal{B} \quad k = 1, \ldots, n
\end{aligned}
\tag{12}
$$

where the quantity $\hat{b}_t$ is defined as $\hat{b}_t = \sum_{t \in N} \langle A_{kt}, P_k(\hat{\lambda}) \rangle$ if $\hat{\lambda}_t > 0$ and $b_t = \min \left\{ \sum_{t \in N} \langle A_{kt}, P_k(\hat{\lambda}) \rangle, b_t \right\}$ if $\lambda_t = 0$. Since $\hat{\lambda}, P_k(\hat{\lambda})$ satisfy the complementary slackness conditions for (12) by construction, they are optimal solutions to it. Note that in general the solution of the sampled linear program could be fractional (i.e. $\hat{P}_k$ for some members of the Birkhoff polytope). However, as a consequence of Lemma 7.2, it is in fact the case that $\hat{P}_k = P_k(\hat{\lambda})$ are optimal.

Suppose $\hat{\lambda}_t > 0$, then the tth primal constraint is tight in the sampled linear program, i.e. $\sum_{k \in S} \langle A_{kt}, P_k(\hat{\lambda}) \rangle = (1 - \epsilon) \epsilon b_t$. By applying an argument along similar lines to part 2 above and applying concentration, it follows that with probability at least $1 - \epsilon$,

$$\hat{b}_t = \sum_{t \in N} \langle A_{kt}, P_k(\hat{\lambda}) \rangle \leq b_t.$$

Let P_k^* be the optimal solution to (6). It follows that P_k^* are also feasible solutions to (12). As a consequence, $\sum_{k \in N} \langle C_k, P_k(\hat{\lambda}) \rangle \geq$ OPT, where OPT refers to the objective function value of the optimal solution to (6).

By again applying a concentration argument as above, we can argue that with probability exceeding $1 - \epsilon$,

$$\sum_{k \in N \setminus S} \langle C_k, P_k(\hat{\lambda}) \rangle \geq (1 - \epsilon) \sum_{k \in N} \langle C_k, P_k(\hat{\lambda}) \rangle \geq (1 - \epsilon) \text{OPT}.$$

Since the returns from the first $\hat{n}$ sessions are non-negative (under an arbitrary assignment), the online algorithm returns a reward of at least $(1 - \epsilon)$OPT. $\qquad\square$

REFERENCES

[1] Deepak Agarwal, Shaunak Chatterjee, Yang Yang, and Liang Zhang. 2015. Constrained Optimization for Homepage Relevance. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. ACM, New York, NY, USA, 375–384. DOI : http://dx.doi.org/10.1145/2740908.2745398

[2] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click Shaping to Optimize Multiple Objectives. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. ACM, New York, NY, USA, 132–140. DOI : http://dx.doi.org/10.1145/2020408.2020435

[3] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2012. Personalized Click Shaping Through Lagrangian Duality for Online Recommendation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. ACM, New York, NY, USA, 485–494. DOI : http://dx.doi.org/10.1145/2348283.2348350

[4] Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. 2014. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Operations Research* 62, 4 (2014), 876–890. DOI : http://dx.doi.org/10.1287/opre.2014.1289

[5] M. F. Balcan, A. Blum, J. D. Hartline, and Y. Mansour. 2005. Mechanism design via machine learning. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. 605–614. DOI : http://dx.doi.org/10.1109/SFCS.2005.50

[6] Sébastien Bubeck. 2015. Convex Optimization: Algorithms and Complexity. *Foundations and Trends in Machine Learning* 8, 3-4 (2015), 231–357. DOI : http://dx.doi.org/10.1561/2200000050

[7] Niv Buchbinder and Joseph (Seffi) Naor. 2009. The Design of Competitive Online Algorithms via a Primal: Dual Approach. *Found. Trends Theor. Comput. Sci.* 3, 2–3 (Feb. 2009), 93–263. DOI : http://dx.doi.org/10.1561/0400000024

[8] Zhe Cao, Tao Qin, Tie yan Liu, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *In Proc. ICMLfi07*. 129–136.

[9] Olivier Chapelle and Ya Zhang. 2009. A Dynamic Bayesian Network Click Model for Web Search Ranking. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 1–10. DOI : http://dx.doi.org/10.1145/1526709.1526711

[10] Sougata Chaudhuri and Ambuj Tewari. 2016. Online Learning to Rank with Top-k Feedback. *CoRR* abs/1608.06408 (2016).

[11] Xiao Alison Chen and Zizhuo Wang. 2015. A dynamic learning algorithm for online matching problems with concave returns. *European Journal of Operational Research* 247, 2 (2015), 379 – 388. DOI : http://dx.doi.org/10.1016/j.ejor.2015.06.029

[12] Nikhil R. Devanur and Thomas P. Hayes. 2009. The adwords problem: Online keyword matching with budgeted bidders under random permutations. In *In Proc. 10th Annual ACM Conference on Electronic Commerge (EC*.

[13] Nikhil R. Devanur, Zhiyi Huang, Nitish Korula, Vahab S. Mirrokni, and Qiqi Yan. 2013. Whole-page Optimization and Submodular Welfare Maximization with Online Bidders. In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce (EC '13)*. ACM, New York, NY, USA, 305–322. DOI : http://dx.doi.org/10.1145/2482540.2482603

[14] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. 2005. Adwords and generalized on-line matching. In *In FOCS fi05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, 264–273.

[15] Peter Orlik and Hiroaki Terao. 1992. *Arrangements of Hyperplanes*. Springer-Verlag.

[16] A. Schrijver. 2003. *Combinatorial Optimization - Polyhedra and Efficiency*.