Offline Optimization for Online Ad Allocation

[Extended Abstract] *

Jon Feldman† Aranyak Mehta‡

Vahab Mirrokni§ S. Muthukrishnan¶

ABSTRACT

We consider online ad allocation from the perspective of optimizing delivery of a given set of ad reservations. Ad allocation naturally fits into the class of online bipartite matching problems: ad nodes are fixed, and impression (ad slot) nodes arrive one at a time, and must be assigned a qualified ad upon arrival. However most previous work on online matching does not model an important aspect of ad allocation: since past site traffic is known to the the ad serving system, it can make useful forecasts about impression supply, and use those forecasts to make better serving decisions. We model these forecasts as a known distribution over ad slot types.

Inspired by the "optimize-and-dispatch" architecture for expressive ad auctions (Parkes & Sandholm, 2005), we explore the use of offline optimization as way to "guide" online decisions about ad allocation. This general approach not only leads to better allocations, it has the advantage that it can incorporate complex constraints on ad targeting or serving (e.g., fairness, scheduling, budgets) while still maintaining the robustness of an online, dynamic allocation rule. Our technical contribution focuses on maximizing a simple efficiency metric (the number of ads served). We give an algorithm that employs a novel application of the idea of the power of two choices from load balancing (Azar *et al*, 1999; Mitzenmacher, 2001): we compute two disjoint solutions to the expected instance, and use both of them in the online algorithm in a prescribed preference order.

We prove that our algorithm achieves an approximation ratio of 0.67 when compared to the optimal "in hindsight" solution. Since $0.67 > 1 - 1/e$, this result has independent interest from the perspective of online optimization theory, since the quantity $1 - 1/e$ is a notorious "ceiling" that previous algorithms for online matching problems have yet to surpass.

1. INTRODUCTION

Internet websites have multiple pages (e.g., sports, real estate, etc), and several slots where they can display ads (say an image or video or a block of text). Each user who views one of these pages is shown ads, i.e., the ads get what is called an "impression." Advertisers typically pay the website per impression and buy them (usually in lots of one thousand) ahead of time, often specifying a subset of pages on which they would like their ad to appear, or a type of user they wish to target. All such sales are entered into an ad delivery system (ADS).

When a user visits one of the pages, the ADS determines the set of eligible ads for that slot, and selects an ad to be shown. Not all ads are eligible or suitable for each slot (because of targeting or quality criteria), and so the ADS must select carefully. However because of contractual obligations, the ADS must also be aggressive enough to satisfy (to a best effort) the reservations in its system.

This choice faced by the ADS is naturally modeled algorithmically as an online (in two senses of the word) *bipartite graph matching* problem. In such a problem, nodes (ad slots) on one side of a graph arrive one at a time, and upon arrival must be assigned to a neighbor (ad) on the other side of the graph, while satisfying some constraints (for example that each ad is assigned at most a specified number of times). Bipartite matching problems are central in combinatorial optimization with many applications besides online advertising. However in the optimization literature the most common way to analyze matching methods is in a *worst-case* model, where the algorithm knows nothing about the underlying graph. This runs contrary to the ad allocation setting, where the ADS has a plethora of information regarding the expected ad slots, obtained via forecasting models applied to past inventory data. Indeed it is standard industry practice to use traffic estimates in order to judge whether a new sale can be accommodated. Certainly the forecasting model that the ADS uses to make these inventory determinations can also be leveraged at the time of serving the ad in order to make better decisions about which ad to serve.

*A full version of this paper can be found in [14].

†Google, New York, NY. Email: jonfeld@google.com

‡Google, Mountain View, CA. Email: aranyak@google.com

§Google, New York, NY. Email: mirrokni@google.com

¶Google, New York, NY. Email: muthu@google.com

Fifth Workshop on Ad Auctions 2009, Stanford, CA

A natural first attempt at using traffic forecasts for online ad serving is simply to assume that the forecasts will be accurate, solve an optimization problem under this assumption, and commit to that solution when serving ads online. This paradigm is general enough to capture not only complex targeting criteria, but also stylized serving constraints like frequency capping (where each user sees a particular ad no more than, say, seven times) or smoothness of delivery. In addition, the objective function of the optimization can be tuned to explicitly balance goals like efficiency, revenue, fairness, underserving penalties, etc.

However, committing to a particular solution, while sound in theory (if the distribution of the arriving ad slots concentrates tightly around its mean), is highly non-adaptive and thus highly sensitive to inaccuracies in the forecasts. This effect is exacerbated by the fact that often "ad slot" refers not to a banner on a webpage that gets a million visitors per day, but rather an obscure combination of targeting criteria or quality features that is expected to occur only a handful of times per day (e.g., a banner ad on a sports page about baseball that is viewed between 5pm and 6pm by someone in New York who has never seen the ad before.) In this "long tail" setting, committing beforehand to a particular ad matching can cause severe loss of efficiency or underdelivery. Parkes and Sandholm [23], and others [1, 7, 24] suggest a more dynamic "optimize-and-dispatch" strategy: instead of solving an optimization problem that assumes forecasts will be accurate, solve an optimization problem (or problems) whose solution(s) can serve as a "guide" to a particular robust online selection heuristic.

The technical contribution of this paper is to extend the theory of online bipartite matching to this "optimize-and-dispatch" setting. We prove that this approach has guarantees even in the most basic ad allocation setting where the goal is to maximize efficiency, defined as simply the number of ads served. For this simple setting, we provide an efficient offline planning algorithm together with its partner online allocation heuristic. The planning algorithm involves the careful selection of *two* independent solutions to the optimization, and the heuristic is to simply apply these solutions in a prescribed preference order. This simple approach is not only amenable to real application, we also prove analytically that the efficiency achieved by our algorithm is higher than both the best-known matching methods in the worst-case model, as well as the natural non-adaptive algorithm. Specifically, we prove that our algorithm achieves an approximation ratio (when compared to the optimal "in hindsight" solution) better than known methods and analyses.

Our results prove that the most effective ad serving techniques must combine the power of offline optimization with the adaptiveness of simple online heuristics. Applying this framework to more realistic ad serving settings is important. For example the ADS may want to maximize the value of the contracts fulfilled, rather than the total number of impressions, or may want to maximize some notion of quality of ads served. One extension that we address is frequency capping, which we discuss in Section 5, but there is more work to be done in this area, and indeed a range of possible algorithmic techniques to explore beyond the ones we provide here.

1.1 Online Stochastic Matching

We first define the general online bipartite matching problem, and then motivate our particular stochastic variant:

(Online Bipartite Matching) There is a bipartite graph $G(A, I, E)$ with advertisers A and impressions I, and a set E of edges between them. Advertisers in A are fixed and known. Impressions (or requests) in I (along with their incident edges) arrive online. Upon the arrival of an impression $i \in I$, we must assign i to any advertiser $a \in A$ where $(i, a) \in E(G)$. At all times, the set of assigned edges must form a matching (that is, no end points coincide). □

Each $i \in I$ is an "impression type," which may represent a particular web page, or even a cross product of targeting criteria (location, demographic, etc.). Edges (a, i) then capture the fact that advertiser a was interested in an impression of type i.

If the ADS knows nothing about I or E beforehand, and the impressions arrive in an arbitrary order, we have an *worst-case model*. Then, Karp, Vazirani and Vazirani [18] solved this problem by presenting an online algorithm with an "approximation ratio" of $1 - 1/e \simeq 0.632$; in our setting that means that the efficiency (the number of assigned impressions) is at least 63% of the number achieved by an optimal offline optimization. They further showed that no algorithm can achieve a better ratio.

However, since the ADS serves ads on the same web pages from day to day, they have an idea of the traffic that occurs on these websites. While there are inaccuracies and indeed it is nearly impossible to forecast the number of viewers of a webpage in the future, we would expect this information to be useful for planning purposes. Thus the worst-case model for bipartite matching does not capture the true problem facing the ADS—i.e., these forecasts should allow the ADS to produce better matchings than those generated by a myopic algorithm. A more suitable model is one that uses prior information about the impressions, but still has some uncertainly about when and exactly how many of each will arrive.

In this paper we study the *iid* model, where impressions $i \in I$ arrive online according some *known* probability distribution (with repetition). In other words, in addition to G, we are given a probability distribution $\mathcal{D}$ over the elements of I. Our goal is then to compute an efficient solution on $\hat{G} = (A, \hat{I}, \hat{E})$, where $\hat{I}$ is drawn from $\mathcal{D}$.[1] In this *iid* model, a simple greedy algorithm achieves an approximation ratio of $1 - 1/e$ [16, 2]. Nothing better is known.

An alternative stochastic model is the *random order model* where we assume that I is unknown, but impressions in I arrive in a random order. While not as appropriate for the ad allocation setting (since it does not capture uncertainty in the *number* of impressions of each type), this model has proved be an important analytical construct for other problems such as secretary-type problems where worst cases are inherently difficult. It is known that in this case even the

[1]We give more details on this model in Section 2, including a discussion of different ways to characterize an approximation ratio in this context.

greedy algorithm has a (tight) competitive ratio of $1-\frac{1}{e}$ [16]. Further, no deterministic algorithm can achieve approximation ratio better than 0.75 and no randomized algorithm better than 0.83 [16]. Currently the best known approximation ratio remains $1-1/e$.

Can one beat the $1-1/e$ bound? This is the main technical question addressed in this work. We answer this question positively by giving an algorithm whose guarantee in the *iid* model is $\simeq 0.67$, breaking past the $1-1/e$ bottleneck. The core idea of the algorithm is to compute several offline solutions to a matching problem on G, representing a "typical realization" of the distribution $\mathcal{D}$. Then, we use these solutions online as "advice." We give a novel analysis of this algorithm by bounding the value of the offline optimal solution in terms of the structure of this "typical realization."

1.2 Our Results and Techniques.

We present two results for the online stochastic bipartite matching problem under the *iid* model.

- We present an algorithm with an approximation factor of $\frac{1-\frac{2}{e^2}}{\frac{4}{3}-\frac{2}{3e}} \simeq 0.67$. We also show that our analysis is tight, by providing an example for which our algorithm achieves exactly this factor.
- We show that there is no $1-o(1)$-approximation algorithm for this problem. Specifically, we show that any online algorithm will be off by at least 26/27 (or $\approx .99$ if one requires a family of instances that grows with n).

Our algorithms are based on computing an optimal offline solution, and using it to guide online allocation. An intuitive approach under this paradigm is to compute a matching M_{OFF} on the "expected graph"—that is, the one that would result if all impressions occurred exactly as many times as expected. Thereafter, one can use this matching online, that is, when node $i \in I$ arrives, match it with $a \in A$ iff $(i,a) \in M_{\text{OFF}}$. One expects this to perform well if the empirical probability of occurrence of each node $i \in I$ is very close to its value in the distribution. This can be shown if all $i \in I$ occur very frequently using for example the Chernoff bound. However in general, many $i \in I$ will have very low frequency. In this paper, we show that this first attempt achieves (you guessed it) $1-1/e$, and this is tight.

To get our main result and beat $1-1/e$, we compute *two* disjoint offline solutions and use them as follows: when a request arrives, we try to assign it based on the first offline solution, and if that assignment fails, we try the second. In order to identify these two disjoint offline solutions, we solve a max flow problem in a boosted flow graph, and then carefully decompose this maximum flow to two edge-disjoint (near)-matchings. Other than guiding the online decision making, these offline solutions are used to characterize an upper bound for the optimum in each scenario. This bound is determined by identifying an appropriate cut in each scenario that is guided by a cut in the offline solution. This is the main technical part of the analysis, and we hope this technique proves useful for analyzing heuristic algorithms for other stochastic optimization problems.[2]

The idea of using two solutions is inspired by the idea of power of two choices in online load balancing [4, 21]. Power of two choices has traditionally meant choosing between two *random* choices for online allocation; in contrast, we use two *deterministic* choices, carefully computed offline to guide online allocation.[3]

Our results are somewhat more general, and the full version of this work can be found in [14].

1.3 Other Related Work.

Our online stochastic matching problem is an example of online decision making problems studied in the Operations Research literature as stochastic approximate dynamic programming problems [6, 5, 10, 13]. Several heuristic methods have been proposed for such problems (e.g., see Rollout algorithms for stochastic dynamic programming in [5]), but we are not aware of any rigorous analysis of the performance of the heuristics. Recently other online stochastic combinatorial optimization problems like Steiner tree and set cover problems have been studied in the *iid* model (e.g., [17, 15, 11]); one can achieve an approximation factor better than the best bound for the adversarial online variant.

A related ad allocation problem is the *Adwords assignment* problem [20] that was motivated by sponsored search auctions. When modeled as an online bipartite assignment problem, here, each edge has a *weight*, and there is a *budget* on each ad representing the upper bound on the total weight of edges that may be assigned to it. In the offline setting, this problem is NP-Hard, and several approximations have been designed [9, 26, 3]. For the online setting, it is typically assumed that every weight is very small compared to the corresponding budget, in which case there exist $1-1/e$ factor online algorithms [20, 8, 16, 2]. Recently, it has been brought to our attention that an online algorithm [12] gives a $1-\epsilon$-approximation, for any ϵ, for Adwords assignment when *opt* is larger than $O(\frac{n^2}{\epsilon^3})$ times each bid in the iid and random permutation models. Thus, technically, our problem is different from their problem in two ways: the edges are unweighted (making it easier), but OPT is not necessarily much larger than each bid (making it harder – in the bipartite graph case, OPT can be $O(n)$). Moreover, our offline problem is solvable in polynomial time, and we show that no $1-\epsilon$-approximation can be achieved for our problem for some fixed ϵ. In fact, their algorithm, along with other previously studied algorithms (e.g, algorithms based on greedy, greedy bid-scaling, and primal-dual techniques) does not achieve a factor better than $1-\frac{1}{e}$ for our problem, and we beat $1-\frac{1}{e}$ factor using a different technique. An interesting related model for combining stochastic-based and online solutions for the Adwords problem is considered in

[2]For example, this technique might be applicable for proving performance guarantees for heuristics for approximate dynamic programming problems studied in the OR literature [6, 5, 10, 13].

[3]Previously, power of two choices has been used in various congestion control and load balancing settings. Our work is a novel adaptation of this idea to a stochastic bipartite matching setting.

[19], but their approach does not give an improved approximation algorithm for the *iid* model.

We note that the *online stochastic bipartite matching* problem does admit a $1-o(1)$-approximation in two special settings: one is where the arrival rate of each impression type is large, and the other is where the demand of each advertiser is large (in the model above, this corresponds to the extension where we have demands d_j per advertiser, and we are limited to matching d_j impressions to advertiser j during the run of the algorithm). In the former case, it is not hard to show that computing one offline optimal solution for the expected instance and implementing this solution gives a $1-o(1)$ approximation. This can be proved using a Chernoff bound on the number of impressions of each type in each scenario. In the latter case, a simple randomized assignment strategy gives a PTAS [22]. Both these settings, however, fail to capture fully the "long tail" aspect of the ad allocation application. Certainly if an "impression type" represents a particular rare combination of features, we cannot expect it to appear a number of times multiplicatively close to its mean. On the advertiser side, the notion of a "long tail" is admittedly less convincing, as most advertisers reserve bundles of impressions in the thousands. However, if an advertiser imposes restrictions or targeting criteria on their reservations, a publisher may wish to deploy our abstraction of the problem in a different way: when the impressions are sold, the "advertiser" is divided up into multiple individual reservations, each with their own impression quota and targeting criteria. These individual reservations then take on the role of "advertisers" in our model. Of course then our algorithm would decouple these reservations, whereas in reality one would want to make advertiser-level decisions (e.g., to prevent overall underdelivery). However, there are ways to extend our general framework to address this; we give one such example in terms of "frequency capping" in Section 5.

2. PRELIMINARIES

Consider the following online stochastic matching problem in the i.i.d model: We are given a bipartite graph $G = (A, I, E)$ over advertisers A and impression types I. Let $k = |A|$ and $m = |I|$. We are also given, for each impression type $i \in I$, an integer number e_i of impressions we expect to see. Let $n = \sum_{i \in I} e_i$. We use $\mathcal{D}$ to denote the distribution over I defined by $\Pr[i] = e_i/n$.

An instance $\Gamma = (G, \mathcal{D}, n)$ of the *online stochastic matching* problem is as follows: We are given offline access to G and the distribution $\mathcal{D}$. Online, n i.i.d. draws of impressions $i \sim \mathcal{D}$ arrive, and we must immediately assign ad impression i to some advertiser a where $(a, i) \in E$, or not assign i at all. Each advertiser $a \in A$ may only be assigned at most once[4]. Our goal is to assign arriving impressions to advertisers and maximize the total number of assigned impressions. In the following, we will formally define the objective function of the algorithm.

Let $D(i)$ be the set of draws of impression type i that arrive during the run of the algorithm. We let a scenario $\hat{I} = \cup_{i \in I} D(i)$ be the set of impressions. Let $\hat{G}(\hat{I})$ be the "realization" graph, i.e., with node sets A and $\hat{I}$, and edges $\hat{E} = \{(a, i') : (a, i) \in E, i' \in D(i)\}$.

Given an instance $\Gamma = (G, \mathcal{D}, n)$ of the online matching problem, we wish an algorithm ALG for which for any instance Γ of the online matching problem, with high probability $\frac{\mathrm{ALG}(\hat{I})}{\mathrm{OPT}(\hat{I})} \geq \alpha$. In this case, we say that the algorithm achieves approximation factor α with high probability. One could also study weaker notions of approximation, namely $\frac{E[\mathrm{ALG}(\hat{I})]}{E[\mathrm{OPT}(\hat{I})]}$ (the approximation factor in expectation), or $E[\frac{\mathrm{ALG}(\hat{I})}{\mathrm{OPT}(\hat{I})}]$ (the expected approximation factor). Note that if one proves a high-probability factor of α, it implies an approximation factor in expectation, and an expected approximation factor of at least $\alpha - o(1)$.

2.1 Balls in Bins.

In this section we characterize two useful extensions of the standard balls-in-bins problem, where we are interested in the distribution of certain functions of the bins. We characterize the expectations of these functions, and use Azuma's inequality on appropriately defined Doob's Martingales to establish concentration results as needed. In particular, we will use the following facts. The proofs can be found in [14].

FACT 1. *Suppose n balls are thrown into n bins, i.i.d. with uniform probability over the bins. Let B be a particular subset of the bins, and S be a random variable that equals the number of bins from B with at least one ball. With probability at least $1 - 2e^{-\epsilon n/2}$, for any $\epsilon > 0$, we have $|B|(1 - \frac{1}{e}) - \epsilon n \leq S \leq |B|(1 - \frac{1}{e} + \frac{1}{en}) + \epsilon n$*

FACT 2. *Suppose n balls are thrown into n bins, i.i.d. with uniform probability over the bins. Let $B_1, B_2, \ldots, B_\ell$ be ordered sequences of bins, each of size c, where no bin is in more than d such sequences. Fix some arbitrary subset $\mathcal{R} \subseteq \{1, \ldots, c\}$. We say that a bin sequence $B_a = (b_1, \ldots, b_c)$ is "satisfied" if (i) at least one of its bins b_i with $i \notin \mathcal{R}$ has at least one ball in it; or, (ii) at least one of its bins b_i with $i \in \mathcal{R}$ has at least two balls in it. Let S be a random variable that equals the number of satisfied bin sequences. With probability at least $1 - 2e^{-\epsilon^2 n/2}$, we have $S \geq \ell(1 - \frac{2^{|\mathcal{R}|}}{e^c}) - \epsilon dn - \frac{2^{|\mathcal{R}|} c^2}{e^c} \frac{\ell}{n - c^2}$.*

3. HARDNESS

In this section we show that the expected approximation of every (randomized) online algorithm is bounded strictly away from 1.

Consider the 6-cycle G defined by $A = \{a, b, c\}$, $I = \{x, y, z\}$, and $E = \{(x, a), (y, a), (y, b), (z, b), (z, c), (x, c)\}$. The distribution $\mathcal{D}$ is the uniform distribution $(1/3, 1/3, 1/3)$ on I, and $n = 3$. We show that no (randomized) algorithm can achieve an expected approximation factor better than $26/27$ on this instance. Without loss of generality (from the symmetry of the 6-cycle), assume that the first impression to arrive is x and that it gets assigned to advertiser a. Now, if the next two arrivals are both of impression y, then any

[4]All results in this paper hold for a more general case that each advertiser a has a capacity c_a and advertiser a can be assigned at most c_a times. This more general case can be reduced easily to the degree one case by repeating each node a c_a number of times in the instance.

algorithm will only be able to assign one of these. The optimal assignment for the scenario (x, y, y) is to assign x to c, and the two y impressions to a and b. Since the probability of (x, y, y) is $1/9$, the expected approximation factor is at most $(1/9)(2/3) + (8/9)1 = 26/27$.

To get a family of instances on which no algorithm can do better than a constant bounded away from 1, we will have to construct an instance consisting of a large number k copies of 6-cycles. Using this idea, we can prove the following theorem. The details of the proof can be found in [14].

THEOREM 3. *There is an instance of the online stochastic matching problem in which no algorithm can achieve an expected approximation factor better than $\frac{26}{27}$. Moreover, there exists a family of instances with $n \to \infty$ for which no algorithm can achieve an expected approximation of $1 - o(1)$.*

4. OFFLINE ALGORITHMS FOR ONLINE MATCHING

In this section, we present our improved online algorithms guided by offline solutions. Before stating the improved approximation result, we "warm up" with a simple, natural algorithm that uses the idea of computing an offline solution to "guide" our online choices. This algorithm will only achieve a $1-\frac{1}{e}$-approximation (which is tight). The proof of this part illustrates the framework we will use in the second section to beat $1 - \frac{1}{e}$; however we will need a new idea to achieve this—namely, the use of a *second* offline solution.

4.1 "Suggested Matching" Algorithm: a $1 - \frac{1}{e}$-Approximation.

The *suggested matching* algorithm is a first attempt at the approach of using an offline solution for online matching. In this algorithm, we simply find a maximum matching in the graph we "expect" to arrive, then restrict our online choices to this matching.

Offline Algorithm. We will describe this algorithm more formally in terms of the standard characterization of b-matching as a max-flow problem, since we will later use this flow graph explicitly to bound OPT. Given an instance $\Gamma = (G(A, I, E), \mathcal{D}, n)$ of the problem, we will find a max-flow in a graph G_f constructed from G as follows: define a new source node s and an edge (s, a) with capacity 1 to all $a \in A$, direct all edges in E from A to I, and add a sink node t with edges (i, t) from all $i \in I$ with capacity e_i. Let $f_{ai} \in \{0, 1\}$ be the flow on edge (a, i) in this max flow (since all the capacities are integers, we may assume that the resulting flow is integral [25]). For ease of notation, we say $f_{ai} = 0$ if edge $(a, i) \notin E$.

Online Algorithm. When an impression $i' \in D(i)$ arrives online, we choose a random ad a' according to the distribution defined by the flow; i.e., the probability of choosing a' is $\frac{f_{a'i}}{e_i}$. (Note that if $\sum_a f_{ai} < e_i$ there is some probability that no a' is chosen.) If a' is already taken, we do not match i to any ad.[5]

Bounding ALG. The performance of this algorithm is easily characterized with high probability in terms of the computed max-flow. Define $F_a = \sum_i f_{ai}$, and note that $F_a \in \{0, 1\}$; this indicates whether ad a was chosen in the max flow. Let $A^* = \{a \in A : F_a = 1\}$. When an impression $i \in \hat{I}$ arrives online, a particular ad $a : f_{a,i} = 1$ has probability $1/e_i$ of being chosen by the online algorithm; since each impression i has probability e_i/n of arriving, we conclude that each $a \in A^*$ has probability $1/n$ of being chosen by the online algorithm upon each arrival. Thus, to bound the total number of ads chosen we have a balls-in-bins problem with n balls and n bins, and we are interested in lower-bounding the number of bins (among a subset of size $|A^*|$) that have at least one ball. Applying Fact 2, we get that with probability $1 - e^{-\Omega(n)}$, ALG $\geq (1 - \frac{1}{e})|A^*| - \epsilon n$.

Bounding OPT. To bound the optimal solution, we will construct a cut in the realization graph $\hat{G} = (A, \hat{I}, \hat{E})$ using a min-cut of G_f (constructed using the max-flow found by the algorithm) as a "guide." Let (S, T) be a min $s-t$ cut in the graph G_f using the canonical "reachability" cut in G_f; i.e., S is defined as the set of nodes reachable from s using paths in the residual graph after sending the flow f found by the algorithm. This is always a min-cut.[25] Let $A_S = A \cap S$ and define A_T, I_S and I_T similarly.

We claim that there are no edges in E from A_S to I_T; suppose there is such an edge (a, i). Then, a much be reachable from s since $a \in S$, but i must not be reachable since $i \in T$. This implies that there is no residual capacity along (a, i); i.e., $f_{ai} = 1$. However this also implies that there is no residual capacity along (s, a) since (s, a) is the only edge entering a and it has capacity 1, and that there is no other flow leaving a. This implies that a is not reachable in the residual graph, a contradiction. Thus the only edges in the cut (S, T) are from s to A_T (capacity 1) and from $i \in I_S$ to t (capacity e_i). We may conclude using max-flow min-cut that $|A^*| = \sum_a F_a = |A_T| + \sum_{i \in I_S} e_i$.

Now consider the "realization" graph $\hat{G} = (A, \hat{I}, \hat{E})$, and define a max-flow instance $\hat{G}_f$ whose solution has size equal to the maximum matching in $\hat{G}$; i.e., create a source s with edges to all $a \in A$, direct edges of $\hat{E}$ toward $\hat{I}$, and create a sink t with edges from all $i' \in \hat{I}$. Set the capacity of every edge to one. Note that any $s-t$ cut in $\hat{G}_f$ is a bound on OPT.

We define an $s-t$ cut in $(\hat{S}, \hat{T})$ in $\hat{G}_f$ as follows. Let $\hat{I}_S = \cup_{i \in I_S} D(i)$ and $\hat{I}_T = \cup_{i \in I_T} D(i)$. Define $\hat{S} = A_S \cup \hat{I}_S$ and $\hat{T} = A_T \cup \hat{I}_T$. Note that since there are no edges from A_S to I_T in G_f, there are also no edges from A_S to $\hat{I}_T$ in $\hat{G}_f$. Thus the size of the cut $(\hat{S}, \hat{T})$ is equal to $|\hat{I}_S| + |A_T|$. An online impression ends up in the set $\hat{I}_S$ with probability $\sum_{i \in I_S} e_i/n$, independent of the other impressions. Using a Chernoff bound, we can conclude that for any $\epsilon > 0$, with probability $1 - e^{-\Omega(n)}$ (over the scenarios), the size of the cut (and therefore OPT) obeys OPT $\leq |A_T| + \sum_{i \in I_S} e_i + \epsilon n = |A^*| + \epsilon n$.

[5]Clearly, making an arbitrary available match is always as good (and in some cases better) than doing nothing; we present the algorithm this way for ease of presentation.

Tightness of the Analysis. Consider a special case of the online matching problem $\Gamma(G, \mathcal{D}, n)$ where $e_i = 1$ for each $i \in I$ and the underlying graph G is a complete bipartite graph. The algorithm will find a perfect matching between I and A, and so each ad is matched with probability at least $1 - \frac{1}{e}$. Using Fact 2, the algorithm achieves $\approx (1 - \frac{1}{e})n$ with high probability. However, the optimum is n. Therefore:

THEOREM 4. *The approximation factor of the* suggested matching *algorithm is* $1 - \frac{1}{e}$ *with high probability, and this is tight, even in expectation.*

4.2 "Two Suggested Matchings" (TSM) Algorithm: Beating $1 - \frac{1}{e}$.

To improve upon the *suggested matching* algorithm, we will instead use *two* disjoint (near-)matchings to guide our online algorithm. To find these matchings, we boost the capacities of the flow graph and then decompose the resulting solution into disjoint solutions. The second solution allows to to break the $1 - \frac{1}{e}$ barrier and prove:

THEOREM 5. *For any* $\epsilon > 0$*, with probability at least* $1 - e^{-\Omega(n)}$*, as long as* $\mathrm{OPT} = \Omega(n)$*, the* two suggested matchings *algorithm achieves approximation ratio*

$$\frac{\mathrm{ALG}}{\mathrm{OPT}} - \epsilon \;\geq\; \alpha := \frac{1 - \frac{2}{e^2}}{\frac{4}{3} - \frac{2}{3e}} \;\approx\; 0.67029 \;>\; 1 - \tfrac{1}{e}.$$

Moreover, this ratio is tight; specifically, there is a family of instances for which the two suggested matchings *algorithm has expected approximation factor at most* $\alpha + \epsilon$.

The proof of this theorem is given in the full version [14]; here we sketch the algorithm and some of the ideas in the analysis. Throughout the section, we assume $e_i = 1$ for all $i \in I$, which also implies $m = n$. Extending to integer e_i is a simple reduction to this case, which we sketch in Section 4.2.3.

4.2.1 The TSM Algorithm.

In this algorithm, we construct a *boosted flow graph* G_f, built from G in the standard reduction of matching to max-flow; i.e., create a source s with edges to all $a \in A$, direct the edges of G towards nodes in I, and create a sink t with edges from all $i \in I$. However, we set the capacities of the edges differently than in the max-flow reduction: (i) Edges (s, a) from the source get capacity 2, (ii) edges $(a, i) \in E$ get capacity 1, and (iii) edges (i, t) from I to t get capacity 2.

We find a max-flow in this graph from s to t. Since all the capacities are integers, we may assume that the resulting flow is integral [25]. Let E_f be the set of edges $(a, i) \subseteq E$ with non-zero flow on them, which must be unit flow. Since the capacities of edges (s, a) and (i, t) are all 2, we know that the graph induced by E_f is a collection of paths and cycles. Using this structure, we assign colors blue and red to the edges of E_f as follows:

- Color the cycle edges alternating blue and red.
- Color the edges of the odd-length paths alternating blue and red, with more blue than red.
- For the even-length paths that start and end with nodes $a \in A$, alternate blue and red.
- For the even-length paths that start and end with impressions $i \in I$, color the first two edges blue, and then alternate red, blue, red, blue, etc., ending in blue.

Note that all $i \in I$ are incident to either no colored edges, one blue edge, or a blue and a red edge.

The TSM algorithm for serving online ad impressions is simple: For each $i \in I$, the first time i arrives try the blue edge; the second time i arrives try the red edge. More formally, for all $i \in I$ maintain a count x_i of the number of impressions $i' \in D(i)$ that have arrived so far. When $i' \in D(i)$ arrives: if $x_i = 0$, set a' to be the ad along i's blue edge (if i has a blue edge); if $x_i = 1$, set a' to be the ad along i's red edge (if i has a red edge). Now assign i to a' if a' is unassigned. If this a' is already assigned, or if $x_i > 1$, do not make an assignment.[6]

4.2.2 Performance of the TSM Algorithm.

To analyze the performance of this algorithm, we first derive a lower bound on the number of ads assigned during the run of the algorithm. We do so in terms of the incidence pattern of the different ads with respect to the edges E_f. Specifically, let A_{BR} be the ads that are incident to a blue and a red edge, and A_{B} be the ads that are incident to only a blue edge. Similarly define A_{BB} and A_{R}. We have

$$|E_f| = 2A_{\mathrm{BR}} + 2A_{\mathrm{BB}} + A_{\mathrm{B}} + A_{\mathrm{R}}. \tag{1}$$

Consider some $a \in A_{\mathrm{B}}$ with blue edge (a, i). The event that a is ever chosen is exactly the event that some $i' \in D(i)$ is ever drawn from D, since then we will choose a (and no other impression will choose a). Since $e_i = 1$, this is exactly the probability that a particular bin is non-empty in a balls-in-bins problem with n balls (the online impressions), and $m = n$ bins (the impression types I). Applying Fact 1, we get that with high probability the number of ads chosen from A_{B} is at least $|A_{\mathrm{B}}|(1 - \frac{1}{e}) - \epsilon n$. Now consider some $a \in A_{\mathrm{BR}}$ with blue edge (a, i_b) and red edge (a, i_r). If $|D(i_b)| \geq 1$, or if $|D(i_r)| \geq 2$, then a will definitely be chosen. Thus we can apply Fact (2) with n balls, $m = n$ bins, $c = 2$, bin sequences equal to the neighborhood sets of A_{BR} along the blue and red edges (ordered blue, red), $d = 2$ (since each impression is incident to at most 2 edges of E_f), and $\mathcal{R}$ set to the second (red) bin of the bin sequence. We conclude that with high probability, the number of ads chosen from A_{BR} is at least $|A_{\mathrm{BR}}|(1 - \frac{2}{e^2}) - \epsilon n$. Similar reasoning gives bounds with coefficients of $(1 - \frac{1}{e^2})$ for A_{BB} and $(1 - \frac{2}{e})$ for A_{R}. We may conclude that with high probability (over the scenarios), $\mathrm{ALG} \geq (1 - \frac{1}{e^2})|A_{\mathrm{BB}}| + (1 - \frac{2}{e^2})|A_{\mathrm{BR}}| + (1 - \frac{1}{e})|A_{\mathrm{B}}| + (1 - \frac{2}{e})|A_{\mathrm{R}}| - 4\epsilon n$. Note that since $|A_B| \geq |A_R|$, we can also assert

$$\mathrm{ALG} \geq (1 - \tfrac{1}{e^2})A_{\mathrm{BB}} + (1 - \tfrac{2}{e^2})A_{\mathrm{BR}} + (1 - \tfrac{3}{2e})(A_{\mathrm{B}} + A_{\mathrm{R}}) - 4\epsilon n. \tag{2}$$

[6] A slight improvement to this algorithm is to try to match along the red edge if matching along the blue edge fails; we do not make use of this in the analysis so we leave it out for clarity.

4.2.3 Bound on the Optimal Solution.

To bound the optimal solution, just as in the "Suggested Matching" algorithm we construct a cut in the realization graph $\hat{G} = (A, \hat{I}, \hat{E})$ using a min-cut of G_f as a "guide." The derivation of this bound is quite a bit more involved, and we refer the interested reader to [14] for details. What we end up with is the following bound:

$$\text{OPT} \leq (\frac{4}{3} - \frac{2}{3e})A_{\text{BR}} + (\frac{5}{3} - \frac{4}{3e})A_{\text{BB}} + (1 - \frac{1}{e})(A_{\text{B}} + A_{\text{R}}) + \epsilon' n.$$

Putting this together with Equation 2, we can prove Theorem 5 as follows:

$$\begin{aligned} \frac{\text{ALG}}{\text{OPT}} + \epsilon &\geq \min\left\{ \frac{1 - \frac{1}{e^2}}{\frac{5}{3} - \frac{4}{3e}}, \frac{1 - \frac{2}{e^2}}{\frac{4}{3} - \frac{2}{3e}}, \frac{1 - \frac{3}{2e}}{1 - \frac{1}{e}} \right\} \\ &= \min\{.735\ldots, .670\ldots, .709\ldots\} \\ &= \frac{1 - \frac{2}{e^2}}{\frac{4}{3} - \frac{2}{3e}} \approx .670. \end{aligned}$$

This analysis is in fact tight; i.e., there is an example instance in which our algorithm meets this bound with high probability (see the full version [14] for details).

In this section we have assumed that $e_i = 1$. For arbitrary integer e_i, we can easily give a reduction to the case $e_i = 1$, as follows. Given a set of instance $\Gamma = (G, \mathcal{D}, n)$, we reduce to a new instance $\Gamma' = (G', \mathcal{D}', n)$ with $e'_i = 1$ by making e_i copies of each impression type i. Then, when an impression of type i arrives online, "name" it randomly according to one of its copies. The resulting distribution $\mathcal{D}'$ is uniform over the impression types I' in the new instance. Let $\hat{I}$ be the impressions that are drawn from $\mathcal{D}$ in one run of the algorithm, and let $\hat{I}'$ be the resulting draws from $\mathcal{D}'$. By the arguments above, we achieve the desired bound on ALG/OPT$'$ with high probability, where OPT$'$ is with respect to $\hat{I}'$; however we have OPT$'$ = OPT, since the realization graphs $\hat{G} = (A, \hat{I}', \hat{E}')$ and $\hat{G}' = (A, \hat{I}, \hat{E})$ are in fact the same graph.

5. FREQUENCY CAPPING

A useful generalization of the online matching problem that is well-motivated by the ad allocation application is when the advertisers have "frequency caps;" i.e., they do not want the same user to see their ad more than some fixed (constant) number of times. We can regard the user as a "feature" of the impression; i.e., that an "impression" i as we've used it in this paper is in fact a pair $\langle i, u \rangle$, where u is a particular user, and we have a distribution that gives us $e_{\langle i,u \rangle}$, the expected number of impressions of each type from each user. Also as part of the input, we are given, for each advertiser a, a total number of impressions d_a and a cap c per user. We could also regard these caps as operating as impression limits on other features, e.g., demographic or geographic.

Our $1 - \frac{1}{e}$-approximation algorithm (the *suggested matching* algorithm) from Section 4.1 is easily extended to this generalization of the problem. Here we give a sketch of this extension. For the algorithm, we simply make another layer U of nodes in our max-flow computation, with one node $\langle a, u \rangle$ for each (advertiser, user) pair. We make edges from each $a \in A$ to this layer with capacity c, and set the capacity of the edge edge (s, a) to d_a. The algorithm proceeds as before, and one can easily show with the same argument that the number of impressions matched is $\simeq F(1 - 1/e)$, where F is the value of the flow. Then, by reasoning about the min-cut in this graph, with some simple reasoning about where this new layer sits in the min-cut, one can still show that OPT is bounded by F with high probability, giving the desired approximation ratio.

Interestingly, it is more challenging to generalize the TSM algorithm. Setting the capacities to $2d_a$ and 2, respectively, of the top and mid-layer edges does not work as desired, since then the flow could be spread among more than d_a nodes in the middle layer.

6. CONCLUDING REMARKS

Applying the insights to the display ads application. The approach of using the offline solution to allocate ads online may be quite useful in practice because while one can invest some time offline to find the guiding solutions, the online allocation has to be done very quickly in this application. One can use this approach to model other objective functions such as fairness in quality of ad slots assigned to ads, which may be solvable offline with some computational effort. Our example of frequency caps is a promising start in this direction.

Generalizing the algorithm. One can generalize the two-matching algorithm to a "k-matching" algorithm by computing k matchings instead of 2 matchings, and then using them online in a prescribed order. We can easily show that if the underlying expected graph G admits k edge-disjoint perfect matchings, the approximation factor of such an algorithm is $1 - \frac{2}{e^2} \simeq 0.72$ and $1 - \frac{5}{e^3} \simeq 0.75$ for $k = 2$ and $k = 3$ respectively, however for $k = 3$, we do not know how to generalize our result to all graphs. One natural question left open by this work is what constant $c(k)$ is achieved by extending to k matchings, where $.67 \leq c(k) \leq .99$.

Fractional version. A theoretical version of online stochastic matching problem that may be of interest is the case in which e_i's are not necessarily integers, but arbitrary rational numbers. We observe the analysis of the "one suggested matching" algorithm can be generalized to this case, but do not know how to generalize the analysis of the "two suggested matchings" algorithm. The details of this extension can be found in [14].

Acknowledgements

We thank Ciamac Moallemi, Nicole Immorlica and the anonymous reviewers for pointing us to related work.

7. REFERENCES

[1] Zoe Abrams, Ofer Mendelevitch, and John Tomlin. Optimal delivery of sponsored search advertisements subject to budget constraints. In *Proceedings of the 8th ACM conference on Electronic commerce*, 2007.

[2] S. Alaei and A. Malekian. Maximizing sequence-submodular functions, manuscript. 2009.

[3] Y. Azar, B. Birnbaum, A.R. Karlin, C. Mathieu, and C.T. Nguyen. Improved Approximation Algorithms for Budgeted Allocations. In *Proceedings of the 35th*

international colloquium on Automata, Languages and Programming, Part I, pages 186–197. Springer-Verlag Berlin, Heidelberg, 2008.

[4] Yossi Azar, Andrei Z. Broder, Anna R. Karlin, and Eli Upfal. Balanced allocations. *SIAM J. Comput.*, 29(1):180–200, 1999.

[5] D. Bertsekas. *Dynamic programming and optimal control*. 2007.

[6] Dimitri P. Bertsekas and David A. Castanon. Rollout algorithms for stochastic scheduling problems. *Journal of Heuristics*, 5(1):89–108, 1999.

[7] C. Boutilier, D. Parkes, T. Sandholm, and W. Walsh. Expressive banner ad auctions and model-based online optimization for clearing. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2008.

[8] N. Buchbinder, K. Jain, and J.S. Naor. Online Primal-Dual Algorithms for Maximizing Ad-Auctions Revenue. In *Algorithms–ESA 2007 15th Annual European Symposium, Eilat, Israel, October 8-10, 2007: Proceedings*, page 253. Springer, 2007.

[9] D. Chakrabarty and G. Goel. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science-Volume 00*, pages 687–696. IEEE Computer Society Washington, DC, USA, 2008.

[10] Daniela Pucci de Farias and Benjamin Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Math. Oper. Res.*, 29(3):462–478, 2004.

[11] Brian Dean, Michel Goemans, and Jan Vondrak. Adaptivity and approximation for stochastic packing problems. In *SODA*, 2005.

[12] Nikhil Devanur and Thomas Hayes. The adwords problem: Online keyword matching with budgeted bidders under random permutations, manuscript. 2009.

[13] Vivek F. Farias and Benjamin Van Roy. Approximation algorithms for dynamic resource allocation. *Oper. Res. Lett.*, 34(2):180–190, 2006.

[14] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1 - 1/e. FOCS 2009, to appear. Manuscript, `http://arxiv.org/abs/0905.4100`., 2009.

[15] Naveen Garg, Anupam Gupta, Stefano Leonardi, and Piotr Sankowski. Stochastic analyses for online combinatorial optimization problems. In *SODA*, pages 942–951, 2008.

[16] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *SODA*, pages 982–991, 2008.

[17] Fabrizio Grandoni, Anupam Gupta, Stefano Leonardi, Pauli Miettinen, Piotr Sankowski, and Mohit Singh. Set covering with our eyes closed. In *FOCS*, pages 347–356, 2008.

[18] R.M. Karp, U.V. Vazirani, and V.V. Vazirani. An optimal algorithm for online bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, 1990.

[19] Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. Allocating online advertisement space with unreliable estimates. In *ACM Conference on Electronic Commerce*, pages 288–294, 2007.

[20] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. In *FOCS*, 2005.

[21] Michael Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1094–1104, 2001.

[22] Rajeev Motwani, Rina Panigrahy, and Ying Xu 0002. Fractional matching via balls-and-bins. In *APPROX-RANDOM*, pages 487–498, 2006.

[23] D. Parkes and T. Sandholm. Optimize-and-dispatch architecture for expressive ad auctions. In *Proceedings of the First Workshop on Sponsored Search Auctions, at the ACM Conference on Electronic Commerce*, 2005.

[24] Elan Pavlov. Truthful polynomial time optimal welfare keyword auctions with budget constraints. In *NetEcon+IBC*, 2007.

[25] Alexander Shrijver. *Combinatorial Optimization-Polyhedra and Efficiency*. 2003.

[26] A. Srinivasan. Budgeted Allocations in the Full-Information Setting. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 11th International Workshop, Approx 2008 and 12th International Workshop, Random 2008, Boston, Ma, USA, August 25-27, 2008*, page 247. Springer, 2008.