Advertising in a Stream

Samuel Ieong Mohammad Mahdian Sergei Vassilvitskii
Google, Inc.
Mountain View, CA, USA.
{sieong, mahdian, sergeiv}@google.com

ABSTRACT

One of the most important innovations of social networking websites is the notion of a "feed", a sequence of news items presented to the user as a stream that expands as the user scrolls down. The common method for monetizing such streams is to insert ads in between news items. In this paper, we model this setting, and observe that allocation and pricing of ad insertions in a stream poses interesting algorithmic and mechanism design challenges. In particular, we formulate an optimization problem that captures a typical stream ad placement setting. We give an approximation algorithm for this problem that provably achieves a value close to the optimal, and show how this algorithm can be turned into an incentive compatible mechanism. Finally, we conclude with a simple practical algorithm that makes the allocation decisions in an online fashion. We prove this algorithm to be approximately welfare-maximizing and show that it also has good incentive properties.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems; J.4 [**Social and Behavioral Sciences**]: Economics

Keywords

Auctions; Newsfeed Advertising

1. INTRODUCTION

A significant fraction of the web traffic nowadays originates at social networking websites such as Facebook, Twitter and Google+. The main attraction these websites use to sustain user interest and daily traffic is the *newsfeed*, a sequence of social news items typically sorted in reverse-chronological order and presented as an *infinite scroll* page. These news streams are usually monetized by inserting sponsored messages (e.g., promoted news items, pages looking for more "likes", or pay-per-click ads) in between the standard content items. Every time the user interacts with such sponsored messages, the advertiser is charged for this interaction.

Placement of sponsored messages in an infinite newsfeed presents both allocation and pricing challenges. From an allocation standpoint, there is a tension between front loading the advertisements (since fewer people scroll further down in a stream), and making sure the sponsored messages are presented in a coherent context, which increases the interaction rate. For example, travel news stories are more relevant when surrounded by other travel news items, rather than information about the latest sports scores. This already introduces an allocation structure that differs from classical online advertisements, which are sold by running an auction for each advertising opportunity. In the case of classical online ads, showing an ad now does not usually affect future revenues, and thus an auctioneer always prefers to show an ad to showing no ads. In contrast, under the streaming advertising scenario, if no advertisements form a good match for a particular slot, the optimal solution would have an auctioneer forgo immediate revenue, and make it up when the user scrolls further down.

Given an allocation of ads to slots, knowing how to charge for interaction is also a non-trivial problem. Because there is always a chance that a user abandons the newsfeed and stops scrolling further, showing an additional advertisement reduces the number of news items a user sees, and thus has cascading effects on future ads. Moreover, there may be only a single ad in the system that matches a particular slot (for example only a single travel ad), therefore, charging a "second-price" on a per slot level is not a viable solution.

1.1 Our Contributions

We propose a simple model for user behavior in a feed, and formally define the stream advertising problem. It is worth noting that in addition to newsfeeds, our model of stream advertising is also applicable to ad placement in personalized TV and radio stations.

Our model, formally defined in Section 3, captures two important considerations: that an ad can only be placed in between certain news items that provide a reasonable context for the ad, and that the expected value to the advertiser decreases as the ad is placed lower in the page, as the probability that the user scrolls past the ad decreases. The combination of these two constraints leads to a non-linear combinatorial optimization problem.

We then show how to solve the allocation problem. We show how to approximate the optimal allocation to any desired degree of accuracy in polynomial time. Special cases of

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2568030.

our algorithm that achieve a constant approximation factor are quite fast and practical.

Finally, we show how these algorithms can be coupled with payment schemes to get incentive compatible, approximately efficient mechanisms for the stream ad placement problem.

2. RELATED WORK

With the ever-increasing popularity of social networks such as Facebook and Twitter, the topic of advertising on these networks has been an active area of research. A number of studies have confirmed the efficacy of this advertising medium. Bakshy *et. al.* showed that social signals, such as users' peers and their affiliation, can help with targeting and increase the user's ad response rates [5]. Papadimitriou *et. al.* showed that a user with friends who have seen an ad is more likely to submit queries related to the ad [22]. Todi examined the importance of social networks as an advertising medium and presented a number of case studies on how businesses have successfully employed this medium [23].

Much work on advertising in social networks has focused on incorporating the effects of *positive externalities*—by influencing a user, one benefits from influencing the friends of the user as well. This literature, starting with the pioneering work of Kempe, Kleinberg, and Tardos [15], considers the question of how to select a set of *influencers* to advertise to, in order to maximize the resulting cascading effect among other users. This problem is studied extensively from both algorithmic and mechanism design perspectives; see, for example, [3, 7, 13, 14]. These works take the perspectives of an advertiser that wants to select users to advertise to. In contrast, we consider the problem of deciding, which one of the ads from multiple advertisers, to show in one user's stream, and where should this ad be shown. Our model is also applicable to other forms of streaming media, such as radio and TV advertising.

This work is also related to the study of ad auctions [2, 10, 24] (see Ch. 28 of [21] for a detailed survey). One of the main difficulties in our setting is due to the *negative externalities* among advertisers—by choosing to show an ad in the stream, all future ads suffer a drop in expected value as a user may leave the stream before seeing them. Similar models for negative externalities have been studied in ad auctions [1, 16] and more recently in TV ads [26]. Under a simple cascade model (also known as a Markovian user model), the optimal allocation can be found efficiently via a dynamic program and a truthful mechanism can be obtained using this allocation with VCG payments. In our case, however, there is the additional restriction that ads can only be matched to certain positions due to the surrounding context, which precludes this solution.

Another related area is that of *online matching.* Our ad placement problem can be thought of as a matching problem where each ad imposes negative externality on subsequent ads. In the absence of externalities, there is rich body of work on this topic. Under an adversarial arrival model, Mehta *et. al.* gave a $(1-1/e)$-competitive algorithm [18] for a very general version of the problem. Many extensions or variants of this model are studied; see, for example, [6, 9, 11, 17]. In contrast to this literature, negative externalities are an important element of our model, while our algorithm is not required to be online (although we will explore an almost-online algorithm in Section 7).

3. PROBLEM FORMULATION

A stream information provider has a sequence of content items (for example posts, status updates, etc.) $C = c_1, c_2, \ldots$. For the purposes of the model we assume that the sequence is infinite. We assume that the user will view the content sequentially, starting with c_1, then moving to c_2 and so on. We assume that after examining each item in the sequence the user may choose to leave, or quit the system with probability q. Therefore, the probability that a user views at least k items is $(1-q)^k$. As we will see later, this is not an important assumption for our results; we can handle the case that the probability of viewing at least k items is an arbitrary decreasing function of k.

Our task is to supplement the content stream with a set of advertisements. Let $A = \{a_1, a_2, \ldots, a_n\}$ be a set of possible advertisements. As we mentioned in the introduction, the value of an ad to an advertiser depends crucially on the context, in this case this is the content items viewed by the user before seeing the ad. For example, an advertisement for skis is more likely to be effective when shown after a ski or snow related blog post, and less so after someone talking about their beach vacation.

With each ad a_i, let r_i be the *reward* to the publisher for showing the ad in a suitable position. We denote such a set of positions by $S_i \subseteq \mathbb{Z}^+$. If the ad a_i is placed after one of the content items in S_i and the user reaches the advertisement, the publisher gains utility of r_i, in all other cases the utility gain is 0. As we will note in Section 5, restricting the value of the ad to r_i or 0 is merely for simplicity, and our results hold in the more general case where the value of placing the ad a_i between content items c_j and c_{j+1} is an arbitrary value r_{ij}.

Note that since $S_i \subseteq C$ there is no utility gained by placing an ad immediately following another advertisements, in other words there must be at least one piece of content between successive advertisements. This constraint is natural to ensure a reasonable user experience, and is common in practice. Also, we adopt the standard convention of frequency capping, and insist that each ad in A is shown at most once to the user [1].

A solution M is feasible if for every ad a_i, the position a_i is matched to (if any) is in S_i. The value of placing ad a_i in position $j \in S_i$ is r_i times a discount factor δ_i that captures the probability that the user scrolls past j. This discount factor depends on the number of content items before this position (which is j) as well as the number of ads that are placed before this position. We denote the number of ads that M places before j by z_j^M (or z_j when there is no confusion). In other words, $z_j^M = |\{j' < j : j' \text{ is matched under } M\}|$. With an independent quitting probability q, the discount factor δ_i for placing an ad a_i in position j is defined as $\delta_i := (1-q)^{j+z_j}$. The total value of an assignment M is:

$$U(M) = \sum_{(a_i,j)\in M} r_i \delta_i = \sum_{(a_i,j)\in M} r_i (1-q)^{j+z_j}.$$

In the Stream-Advertising problem we are given a sequence of content C, a quitting probability q, a set of possible advertisements $A = \{a_1, \ldots, a_n\}$, each with a set of valid positions $S_i \subseteq C$ and rewards r_i as described above.

[1]This assumption can be easily removed by replicating each advertiser sufficiently many times.

Our goal is to find a placement M of ads in locations in the stream to maximize the total value $U(M)$.

4. WARM UP: SPECIAL CASES

Before we proceed to give an approximation algorithm for the general model we consider a few special cases to hone intuition about the streaming model.

4.1 Advertising Slots

One simple scenario is when the publisher specifies some slots in the newsfeed as "advertising slots." This is most applicable in the case of TV or radio advertising during a single program, where the commercials are typically played every 10 minutes and the surrounding content does not change significantly enough for the advertisers to use for targeting.

The relaxation of the need for the advertisement to match the surrounding content is best modeled by letting S be the set of advertising opportunities and setting $S_i = S$ for every advertiser $a_i \in A$. In this case there is a simple greedy algorithm that achieves the optimal allocation: simply sort all of the ads in non-increasing order of the rewards (r_i) and allocate them in this order.

4.2 Flat Pricing

A different extreme scenario is to ensure that the positions of the ads remain contextually relevant, but charge a flat price for showing the ads. Let r be the flat reward, then this is equivalent to setting $r_i = r$ for all advertisers $a_i \in A$.

When all of the rewards are the same, there are two factors at play. We simultaneously want to find the largest matching possible, while also making sure that the matched advertisements appear as early as possible in the newsfeed. We will show that a single round of maximum weight matching with appropriately set weights leads to an optimal solution.

Let $S = \cup_i S_i$ denote the set of feasible slots. Although S is infinite, for the purposes of the proof it's enough to consider at most the first n feasible slots for each advertiser. We construct a bipartite graph with one vertex for each advertiser $a \in A$, one for each possible slot $s \in S$ and edges e_{ij} if $s_j \in S_i$. With each slot j, we associate a weight $w_j = (1-\epsilon)^j$ for some small $\epsilon > 0$.

Let $G = (A \cup S, \cup_{ij} e_{ij})$ be the graph representing feasible matches, and suppose each edge e_{ij} has weight w_j.

LEMMA 1. *The maximum weight matching on the graph G described above maximizes the total expected reward.*

PROOF. Observe that since all of the rewards are identical, the utility of a particular matching M is simply

$$U(M) = \sum_{(a_i,j)\in M} (1-q)^{j+z_j}.$$

Let M^* denote the optimum matching (the one maximizing the utility above) and M denote the maximal weight matching on graph G. To prove the Lemma we consider augmenting the set of vertices matched in M by those matched in M^*.

Formally, orient all of the edges in M from A to S, and those in M^* from S to A. Let $H = (A \cup S, M \cup M^*)$ be the graph of the union of the two matchings. If we consider connected components in H, they potentially fall into three categories: cycles, even length paths and odd length paths. We deal with each in turn.

Consider a cycle in H. This implies that all of the vertices in the cycle are matched both in M and in M^*, albeit the matching itself is different. The easiest such an example occurs on a complete graph $K_{2,2}$ which has two possible matchings. Since our objective function is agonistic to the identity of advertisers who are matched, these utility gained by these two matchings is identical.

Consider a component representing an even length path in $P = (m_1, m_1^*, m_2, m_2^*, \ldots, m_k, m_k^*)$, where $m_i \in M$ and $m_i^* \in M^*$. Notice that all of the intermediate nodes on the path are matched both in M and in M^*. Let u be a node matched in M but not in M^*, and v be matched in M^* but not in M. If $u \in A$, then $v \in A$, since the path has even length, therefore the same slots $s \in S$ are present in both matchings and both have the same utility.

Similarly, if $u \in S$, then $v \in S$ as well. If $w_u < w_v$ then M is not an optimal matching: augmenting M along the alternating path P results in a matching of higher weight. On the other hand if $w_u > w_v$ then M^* is not optimal: augmenting M^* by the alternating path P results in a matching of higher utility.

Finally, consider a component representing an odd length path in P. If the first and the last edges of the path are both in M, then by augmenting M^* along P, we gain an additional matched vertex, thus increasing utility and contradicting the fact that M^* was optimal. On the other hand, if both the first and last edges are in M^*, augmenting M by P leads to a matching of higher weight, contradicting its optimality.

Therefore, if M is the maximum weight matching in the graph above, $U(M) = U(M^*)$. □

4.3 Finely Targeted Ads

The final special case we consider is that of very finely targeted ads. In particular, suppose that each set $S_i = \{s_i\}$ is a singleton, that is there is a single position where any ad may be placed. In this case, a simple greedy algorithm gives the optimal assignment.

The algorithm considers the advertisers in a "bottom-up" fashion, starting with the advertiser who can only be appear in the lowest (highest index) slot. Generally, let π be the ordering on the advertisers such that $\pi(i) < \pi(j)$ if the slot s_i is lower than s_j, or if they can be matched to the same slot, then advertiser i generates a higher reward $(r_i > r_j)$.

Consider the advertisers in order given by π. To maximize welfare, $a_{\pi(1)}$ should always be matched, let the initial matching consist of this pair: $M_1 = \{(a_{\pi(1)}, s_{\pi(1)})\}$. At time step t, we consider matching advertiser $a_{\pi(t)}$. If $s_{\pi(t)}$ is already matched, we leave the matching unchanged: $M_t = M_{t-1}$. If not, we consider the marginal benefit and cost of adding $(a_{\pi(t)}, s_{\pi(t)})$ to the matching. The marginal benefit is simply $r_{\pi(t)}$. However, the addition of this edge also reduces the probability of a user seeing any of the following advertisements by a factor of $(1-q)$. Therefore, the marginal cost of adding $(a_{\pi(t)}, s_{\pi(t)})$ to the matching is:

$$\sum_{(a_j,s_j)\in M_{t-1}} q \cdot r_j.$$

If the marginal cost is higher than the marginal benefit, $r_{\pi(t)}$ then $M_t = M_{t-1}$, otherwise $M_t = M_{t-1} \cup \{(a_{\pi(t)}, s_{\pi(t)})\}$.

To show the correctness of the above algorithm, let t be the first time the greedy option differs from the optimal matching, M^*. Consider the matching $\tilde{M}$ which agrees with M^* on advertisers $\{a_i : i > \pi(t)\}$ and with the greedy matching on advertisers in $\{a_j : j \le \pi(t)\}$. This is a feasible solution, moreover by the greedy property, the utility of this matching is strictly higher than that of M^*, a contradiction.

5. ALGORITHMS

In this section, we give a polynomial-time approximation scheme for the Stream Advertising problem. We start by writing the problem as a mathematical program. We let x_{ij} be a binary variable that indicates whether ad a_i is matched to position j. These variables should satisfy the usual matching constraints ($\sum_i x_{ij} \le 1$ for every j and $\sum_j x_{ij} \le 1$ for every i). Furthermore, the variable z_j can be written as $z_j = \sum_i \sum_{j'<j} x_{ij'}$. Therefore, Stream Advertising is equivalent to the following non-linear integer program:

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} \sum_{j \in S_i} r_i x_{ij} (1-q)^{j+z_j} \\ \text{subject to} & \forall j: \ \displaystyle\sum_i x_{ij} \le 1 \\ & \forall i: \ \displaystyle\sum_j x_{ij} \le 1 \\ & \forall j: \ \ z_j = \displaystyle\sum_i \sum_{j'<j} x_{ij'} \\ & \forall i,j: \ \ x_{ij} \in \{0,1\}. \end{array} \qquad (1)$$

The main obstacle in solving this program exactly is the non-linear term in the objective function. Without this term, the program would have been equivalent to a matching integer program, and could have been solved with any maximum matching algorithm (despite the integrality constraint). To overcome this obstacle, we discretize this non-linear term, and replace it with cardinality constraints that limit the number of ads up to any position. We start by describing a simpler version of the algorithm that achieves a constant-factor approximation ratio, and then show how it can be turned into a polynomial-time approximation scheme.

5.1 A 4-approximation algorithm

First, we define $w_{ij} = r_i(1-q)^j$ for $j \in S_i$ and $w_{ij} = 0$ elsewhere. Using this notation, the objective function can be written as $\sum_i \sum_j w_{ij} x_{ij} (1-q)^{z_j}$. Let h denote the largest integer such that $(1-q)^h \ge \frac{1}{2}$. In other words, $h = \lfloor \frac{\log(1/2)}{\log(1-q)} \rfloor$. For positions j such that $z_j \le h$, the term $(1-q)^{z_j}$ is between 1/2 and 1. Therefore, we can remove this term for such j's, losing at most a factor of 2. In addition, we prove in the following lemma that the total value contributed by positions with $z_j > h$ is at most half of the optimal solution, and therefore, removing ads from all such positions costs us at most another factor of 2.

Lemma 2. *Let (x^*_{ij}, z^*_j) be an optimal solution of the integer program (1) with value OPT, and let $h_\theta = \lfloor \frac{\log(\theta)}{\log(1-q)} \rfloor$ for $\theta \in [0,1]$. Then for every $\theta \in [0,1]$, we have*

$$\sum_i \sum_{j: z^*_j > h_\theta} w_{ij} x^*_{ij} (1-q)^{z^*_j} \le \theta \cdot OPT.$$

Proof. Assume, for contradiction, that this inequality does not hold, and construct a new solution by removing all the ads that are assigned to positions with $z^*_j \le h_\theta$ and keeping the rest. By definition of z^*_j, there are precisely $h_\theta + 1$ ads that are removed (since there is exactly one ad placed in a position j with $z^*_j = l$, for every $l = 0, \dots, h_\theta$), and for every ad that is not removed, the corresponding z_j value in the new solution is precisely the old value z^*_j minus $(h_\theta + 1)$. This means that the objective value at the new solution is precisely

$$\sum_i \sum_{j: z^*_j > h_\theta} w_{ij} x^*_{ij} (1-q)^{z^*_j - (h_\theta+1)} > \theta \cdot OPT \cdot (1-q)^{-(h_\theta+1)}$$
$$> OPT,$$

where the first inequality follows from our assumption and the second follows from the definition of h_θ. This means that there is a solution of value more than OPT, which is a contradiction. □

By the above lemma, imposing the constraint that no ad should be assigned to a position with $z_j > h_{1/2}$ reduces the value of the optimal solution by at most a factor of 2. As mentioned earlier, removing the $(1-q)^{z_j}$ term from the objective function of this more constrained problem costs an additional factor of 2. Now, notice that this constraint is equivalent to requiring that in total, no more than h_θ ads should be placed. Therefore, the problem is a maximum weight matching (with weight w_{ij} between ad i and position j) with a constraint on the cardinality of the matching. This problem can be solved in polynomial time using standard weighted matching algorithms.

5.2 A polynomial-time approximation scheme

The idea behind the above 4-approximation algorithm can be extended as follows to get a polynomial-time approximation scheme (PTAS): instead of keeping ads in one set of positions and discarding the rest, we divide the positions into a number of tiers, with the i'th tier corresponding to positions that have a $(1-q)^{z_j}$ term between $(1-\epsilon)^i$ and $(1-\epsilon)^{i-1}$. The non-linear term for each tier can then be replaced by a constant, losing at most a $1-\epsilon$ factor in the process. In return, we need a constraint on the number of ads assigned in each tier. In the rest of this section, we formulate this tiered matching problem and prove that it is still solvable in polynomial-time, and that its solution provides a good approximation to the stream advertising problem.

Let ϵ and k be two parameters that will be fixed later. For every $l = 1, \dots, k$, let $\theta_l = (1-\epsilon)^l$. Consider an optimal solution (x^*_{ij}, z^*_j) to the integer program (1). For $l = 1, \dots, k$, let j_l denote the largest value of j such that $z^*_j \le h_{\theta_l}$, where h_θ is the value defined in Lemma 2. In other words, j_l is the largest value of j such that $(1-q)^{z^*_j} \ge \theta_l$. Also, let $j_0 = 0$.

Assume we know the values of $j_1, \dots, j_k$. We define an ad matching problem $\textsc{Matching}_{j_1,\dots,j_k}$ as follows: The ad a_i can be assigned to any slot $j \in S_i$. The value of assigning this ad to slot j, where $j \in (j_{l-1}, j_l]$ is defined as $\theta_l w_{ij}$. The value of assigning an ad to a slot $j > j_k$ is zero. The total number of ads assigned to slots $(j_{l-1}, j_l]$ is constrained to be at most $h_{\theta_l} - h_{\theta_{l-1}}$ if $l > 1$ and h_{θ_1} if $l = 1$.

First, we observe that any solution SOL to $\textsc{Matching}_{j_1,\dots,j_k}$ corresponds to a solution of the same value (or higher) to Stream Advertising. This is easy, since by the definition of $\textsc{Matching}_{j_1,\dots,j_k}$, there are at most h_{θ_l} ads assigned to slots $1, \dots, j_l$. Therefore, if we treat SOL as a solution of Stream Advertising and define the z_j values, we have

$(1-q)^{z_j} \geq \theta_l$ for every $j \in (j_{l-1}, j_l]$. This means that the value $(1-q)^{z_j} w_{ij}$ of matching ad a_i to position j in STREAM ADVERTISING is greater than or equal to the value of the corresponding matching in $\text{MATCHING}_{j_1,\ldots,j_k}$. Therefore, the value of SOL in STREAM ADVERTISING is at least as much as its value in $\text{MATCHING}_{j_1,\ldots,j_k}$.

Conversely, we argue that if $j_1, \ldots, j_k$ are the values defined above based on an optimal solution (x^*_{ij}, z^*_j) of (1) of value OPT, then the value of the corresponding matching in $\text{MATCHING}_{j_1,\ldots,j_k}$ is close to OPT. To see this, observe that the value of an ad a_i matched to a position $j \in (j_{l-1}, j_l]$ is $\theta_l w_{ij}$ in $\text{MATCHING}_{j_1,\ldots,j_k}$ and $(1-q)^{z^*_j} w_{ij} \leq \theta_{l-1} w_{ij}$ in STREAM ADVERTISING. Therefore, the value of such ads decrease by at most a factor of $\theta_l/\theta_{l-1} = (1-\epsilon)$. Also, the value of any ad assigned to a slot $j > j_k$ drops to zero. By Lemma 2, the total value of such ads is at most $\theta_k \cdot OPT$. Therefore, the value of the solution we get for $\text{MATCHING}_{j_1,\ldots,j_k}$ is at least

$$(1-\epsilon)OPT - \theta_k \cdot OPT = (1 - \epsilon - (1-\epsilon)^k) \cdot OPT.$$

For a given $\delta > 0$, if we take $\epsilon = \delta/2$ and $k = \lceil \log(\frac{\delta}{2}) / \log(1 - \frac{\delta}{2}) \rceil = O(\frac{1}{\delta} \log(\frac{1}{\delta}))$, the above bound would be at least $(1 - \delta) \cdot OPT$.

We are now ready to prove the main result of this section.

THEOREM 1. *For any given $\delta > 0$, there is a $(1-\delta)$-approximation algorithm with running time $N^{O(\frac{1}{\delta}\log(\frac{1}{\delta}))}$ for* STREAM ADVERTISING, *where N is the size of the input.*

PROOF. The algorithm proceeds as follows: try all the $O(N^k)$ possible values of $j_1, \ldots, j_k$, and for each such combination, solve $\text{MATCHING}_{j_1,\ldots,j_k}$. Output the best of these solutions.

All that remains is to argue that $\text{MATCHING}_{j_1,\ldots,j_k}$ can be solved in polynomial time. This follows from the observation that $\text{MATCHING}_{j_1,\ldots,j_k}$ can be formulated as a maximum flow problem: the graph has a source s, a destination t, and three levels between s and t. Each node in the first level corresponds to an ad. There is an edge of capacity 1 and value 0 from s to any such vertex. Vertices in the second level correspond to positions. For each $j \in S_i$, there is an edge from a_i to j of capacity 1 and value $\theta_l w_{ij}$. For each $l = 1, \ldots, k$, there is a vertex v_l in the third level, with edges of capacity 1 and value 0 from each $j \in (j_{l-1}, j_l]$ to v_l. Finally, there is an edge of value 0 from each v_l to t. The capacity of this edge is $h_{\theta_l} - h_{\theta_{l-1}}$ if $l > 1$ and h_{θ_1} if $l = 1$. It is easy to see that the maximum value flow from s to t corresponds to the optimal solution of $\text{MATCHING}_{j_1,\ldots,j_k}$. $\square$

Generalizations.

It is easy to see that all the arguments in the previous section go through in the more general case where the value of placing the ad a_i in position j is an arbitrary function w_{ij} of i and j, times a *non-increasing* discount factor that is a function of z_j. In other words, neither the assumption that the value is r_i if $j \in S_i$ and 0 otherwise, nor the exponential form of discounting is necessary to obtain a PTAS for this problem. However, the assumptions that the value of an assignment can be written as the product of two terms, first only depending on i and j and the second on z_j, and that the latter term is a non-increasing function of z_j are necessary in our arguments.

6. MECHANISM DESIGN

Thus far, we have focused on the algorithmic challenges posed by the STREAM-ADVERTISING problem. All inputs are assumed to be known to the algorithm. In most advertising scenarios, the value of showing a user an ad is known only to the advertiser. The network in charge of placing the ads in the stream will have to obtain the input from the advertisers, who may choose to misrepresent the values if it is in their interest to do so. We now consider the *mechanism design* challenges posed by stream advertising, and propose an approximately optimal solution where advertisers can do no better for themselves than to correctly report these values.

6.1 Problem Definition

In the STREAM-ADVERTISING-MECHANISM-DESIGN problem, a network is responsible for solving the ad placement problem in the presence of self-interested advertisers that have private knowledge of the value of showing their ads to the user.[2] The problem is specified by

- A sequence of content C, known to the network;
- A quitting probability q, known to the network; and
- A set of advertisers $A = \{a_1, \ldots, a_n\}$ interested in showing their ads to the user. We use a_i to denote both the advertiser and its ad. Like before, each ad a_i is associated with a set of valid positions $S_i \subseteq C$ and a reward r_i when the ad is shown. However, while the set S_i is known to the network, as it is determined by the context of the surrounding content items, the reward r_i is only known to the advertiser.

The network solves the problem by specifying a *mechanism*, consisting of an allocation function x and payment functions $p_1, \ldots, p_n$. The mechanism asks the advertisers to report their rewards, and given the reported rewards $\hat{R} = \{\hat{r}_1, \ldots \hat{r}_n\}$, it determines a placement $M = x(\hat{R})$, and charges advertiser a_i the amount $p_i(\hat{R})$. For notation, from the perspective of advertiser a_i, we may refer to the reported rewards as $\hat{R} = (\hat{r}_i, \hat{R}_{-i})$, to highlight the part of the input that advertiser a_i has control over. Note that the reported reward $\hat{r}_i$ needs not equal the true reward r_i; indeed, an advertiser will choose to report the reward that will maximize its utility, given the mechanism, as explained below.

Recall the notations defined in Section 3 regarding placements. Given a placement M, the probability δ_i that an ad a_i will be shown to the user equals

$$\delta_i(M) = \begin{cases} (1-q)^{j+z_j} & \text{if } (a_i, j) \in M \\ 0 & \text{otherwise} \end{cases}.$$

[2]We make the simplifying assumption that an advertiser is only interested in showing the ad to the user. This is for ease of exposition only. Our analysis extends to the more general setting where an advertiser receives its rewards only if the users engage with the ad (for example, click on it), and that the advertiser pays only if an engagement event occurs. While this places additional challenges in estimating the likelihood of engagement, from a mechanism design perspective, our solution continues to hold provided that the probability of engagement is known to the network, and that advertisers are risk-neutral, i.e., they are only interested in maximizing their expected utilities.

The value v_i of a placement M to advertiser a_i is the expected reward the advertiser will receive, i.e.,

$$v_i(M) = r_i \delta_i(M).$$

The advertisers are assumed to have *quasi-linear* utility functions — given a placement M and a payment p_i, the utility u_i of the advertiser equals

$$u_i(M, p_i) = v_i(M) - p_i.$$

Given a mechanism, the advertisers will choose to report their rewards so as to maximize their utilities. A mechanism is *dominant-strategy incentive compatible*, or *truthful*, if for each advertiser a_i, given the reported rewards of the other advertisers $\hat{R}_{-i}$, reporting r_i maximizes its utility, i.e.,

$$r_i \in \arg\max_{r'_i} u_i(x(r'_i, \hat{R}_{-i}), p(r'_i, \hat{R}_{-i})_i).$$

The goal of the STREAM-ADVERTISING-MECHANISM-DESIGN problem is to design a truthful mechanism that maximizes the total value of the placement $U(M)$.

6.2 A Truthful, Efficient, and Approximately Optimal Mechanism

Given an algorithm that can solve the STREAM-ADVERTISING problem optimally, a truthful mechanism can be designed using the algorithm as the allocation function and Vickrey-Clarke-Groves (VCG) payment scheme as the payment function [8, 12, 25]. Unfortunately, in our setting, we do not know if a polynomial-time optimal algorithm exists. With an algorithm that can only solve the problem approximately, the resulting allocation function need not give rise to a truthful mechanism. Fortunately, if the allocation function is *monotonic*, it can be turned into a truthful mechanism with an appropriately chosen payment function [4, 20]. We now define monotonicity for the stream advertising problem.

DEFINITION 1. *An allocation function x for the* STREAM-ADVERTISING-MECHANISM-DESIGN *problem is* monotonic *if for each advertiser a_i, for any reported rewards by the other advertisers $\hat{R}_{-i}$, if $r_i > r'_i$, then*

$$\delta_i(x((r_i, \hat{R}_{-i}))) \geq \delta_i(x((r'_i, \hat{R}_{-i}))).$$

In other words, the probability δ_i of an ad a_i being shown is a non-decreasing function of its reported reward.

Next, we show that the 4-approximation algorithm presented in Section 5 is monotonic, provided that the solution to the weighted matching problem breaks ties consistently, i.e., given two matchings with the same value, it will always pick one over the other. Formally, let the value function $V(M, R)$ be defined as

$$v(M, R) = \sum_{(a_i, j) \in M} r_i (1-q)^j.$$

An algorithm is consistent if for two matchings M and M', and two reward vectors R and R', if $V(M, R) = V(M', R)$ and $V(M, R') = V(M', R')$, then the algorithm always prefer M or always prefer M'. This can be viewed as the algorithm placing a lexicographical ordering over matchings with equal values, and can be achieved by applying a small perturbation to the input instance deterministically.

Fixing inputs C, q, and $S = \{S_1, \dots S_n\}$'s, an algorithm can be viewed as a function that maps a reward vector R to a placement. Denote the function induced by the 4-approximation algorithm by $\textsc{Alg}_{C,q,S}$. This function can be expressed as

$$\begin{aligned} \textsc{Alg}_{C,q,S}(R) = \arg\max_M & \; v(M, R) \\ \text{s.t.} & \; M \text{ is feasible under } C \text{ and } S \\ & |M| \leq h_{1/2} = \left\lfloor \frac{\log 1/2}{\log(1-q)} \right\rfloor. \end{aligned} \tag{2}$$

Note that since rewards are non-negative, the second constraint is actually tight, i.e., regardless of R, the matching will always have cardinality $h_{1/2}$.

In the proceeding, we will consider C, q, and S as fixed, as for any problem instance, these inputs will be known to the algorithm. Henceforth we drop the subscripts and simply write $\textsc{Alg}_{C,q,S}$ as $\textsc{Alg}$. We start by observing the following lemma.

LEMMA 3. *For any two reward vectors R and R',*

$$v(\textsc{Alg}(R), R) \geq v(\textsc{Alg}(R'), R).$$

Further, if $\textsc{Alg}(R) \neq \textsc{Alg}(R')$, then the inequality is strict.

PROOF. Since the constraints in Equation (2) are independent of R, both $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$ are feasible placements. Together with the objective of maximizing $v(M, R)$, we have the desired inequality. The strictness of the inequality is an immediate consequence of consistency. $\square$

To establish the monotonicity of the 4-approximation algorithm, we will show that holding all other rewards fixed, if the reward of an ad increases, if it was matched before, it will continue to be matched, and further, the sum of its position and number of ads above it (i.e., $j + z_j$) will not decrease. We show this via three lemmas.

Let $R = (r_i, R_{-i})$ denote the original reward vector, and $R' = (r_i + \epsilon, R_{-i})$ the reward vector where the reward of ad a_i increases by ϵ.

LEMMA 4. *Given reward vectors R and R', if $(a_i, j) \in \textsc{Alg}(R)$ for some j, then $(a_i, j') \in \textsc{Alg}(R')$ for some $j' \leq j$.*

PROOF. First, we show that ad a_i will be in the placement $\textsc{Alg}(R')$. Suppose not, i.e., $(a_i, k) \notin \textsc{Alg}(R')$ for all k. The key observation is that this matching will have the same value under both reward vectors R and R', since the only difference between R and R' is in the reward of ad a_i, which is not in the matching. Thus,

$$\begin{aligned} v(\textsc{Alg}(R'), R') &\overset{(1)}{\geq} v(\textsc{Alg}(R), R') \overset{(2)}{>} v(\textsc{Alg}(R), R) \\ &\overset{(3)}{\geq} v(\textsc{Alg}(R'), R) \overset{(4)}{=} v(\textsc{Alg}(R'), R'), \end{aligned}$$

where (1) and (3) follows from Lemma 3, (2) due to the increase in reward of a_i, and (4) from the key observation. The contradiction $v(\textsc{Alg}(R'), R') > v(\textsc{Alg}(R'), R')$ confirms the presence of a_i in the placement.

Next, we show that the position at which it is matched is no lower. We do so by deriving two relationships regarding the difference $v(\textsc{Alg}(R'), R') - v(v\textsc{Alg}(R), R)$. First,

$$\begin{aligned} v(\textsc{Alg}(R), R) &\overset{(1)}{\geq} v(\textsc{Alg}(R'), R) \\ &\overset{(2)}{=} v(\textsc{Alg}(R'), R') - \epsilon(1-q)^{j'} \\ \Rightarrow \quad \epsilon(1-q)^{j'} &\geq v(\textsc{Alg}(R'), R') - v(\textsc{Alg}(R), R)), \end{aligned}$$

where (1) follows from Lemma 3, and (2) from the increase in reward of a_i. Analogously,

$$\begin{aligned} v(\textsc{Alg}(R'), R') &\geq v(\textsc{Alg}(R), R') \\ &= v(\textsc{Alg}(R), R) + \epsilon(1-q)^j \\ \Rightarrow \quad \epsilon(1-q)^j &\leq v(\textsc{Alg}(R'), R') - v(\textsc{Alg}(R), R)). \end{aligned}$$

Combining the two yields $\epsilon(1-q)^{j'} \geq \epsilon(1-q)^j$, which together with $\epsilon > 0$ and $0 < (1-q) < 1$, shows that $j' \leq j$. □

Lemma 5. *Given reward vectors R and R', for $(a_i, j) \in \textsc{Alg}(R)$ and $(a_i, j') \in \textsc{Alg}(R')$, $z_{j'}^{\textsc{Alg}(R')} \leq z_j^{\textsc{Alg}(R)}$.*

Proof. From Lemma 4, we have $j' \leq j$. We consider the equality and (strict) inequality cases, separately.

(1) $j' = j$. By consistency of the algorithm, $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$ will be identical, so $z_j = z_{j'}$.

(2) $j' < j$. consider the set $\mathcal{P}$ of *alternating paths and cycles* formed by the symmetric differences of $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$, i.e., if we color the edges in $\textsc{Alg}(R)$ blue and the edges in $\textsc{Alg}(R')$ red, the set of paths and cycles of alternating colors after the common edges in their intersection are removed. Let the path or cycle that contains the blue edge (a_i, j) and the red edge (a_i, j') be denoted P^*; it must be in the same path or cycle as the two edges are connected.

Given a path or cycle $P \in \mathcal{P}$, let $r(P)$ denote the set of red edges in P and $b(P)$ denote the set of blue edges in P. By construction, any path or cycle in $\mathcal{P}$ can differ by at most one in the number of blue edges and the number of red edges. For any $P \neq P^*$, if P has the same number of red and blue edges, then $M' = (\textsc{Alg}(R) \setminus b(P) \cup r(P))$ and $M'' = (\textsc{Alg}(R') \setminus r(P) \cup b(P))$ are both feasible matchings. By Lemma 3,

$$\begin{aligned} v(\textsc{Alg}(R), R) &\geq v(M', R) \\ &= v(\textsc{Alg}(R), R) - v(b(P), R) + v(r(P), R) \\ v(\textsc{Alg}(R'), R') &\geq v(M'', R') \\ &= v(\textsc{Alg}(R'), R') - v(b(P), R') + v(r(P), R'). \end{aligned}$$

By construction, $b(P)$ and $r(P)$ have the same value under R and R', since they do not involve r_i. Thus, we have $v(r(P), R) = v(b(P), R)$. Due to consistency, $\textsc{Alg}$ would have picked either $b(P)$ for both or $r(P)$ under both R and R', thus there can be no such paths or cycles in $\mathcal{P}$ except for possibly P^*.

Now consider the set $\mathcal{B}$ of paths with one more blue edge. For each path in $\mathcal{B}$, there must be some path in $\mathcal{P}$ with one more red edge, since the size of the two placements are equal; call this set $\mathcal{R}$. Excluding P^*, for each path $P' \in \mathcal{B}$, consider its union P with a path $P'' \in \mathcal{R}$. The same argument from above applies, showing that no such pair can exist, except possibly if a path is paired with P^*. Thus, summarizing, $\mathcal{P}$ either consists of just P^* if P^* has the same number of blue and red edges, or P^* with another path P_*, where P^* has one more blue edge and P_* one more red edge, or vice versa. We now consider these cases separately.

Let Z_k^M be the set of positions matched under M that is smaller than position k. Note that $z_k^m = |Z_k^M|$.

(2a) P^* has same number of blue and red edges. If P^* is a cycle, then the set of positions matched under $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$ are the same, hence for all k, $Z_k^{\textsc{Alg}(R)} = Z_k^{\textsc{Alg}(R')}$. Since $j' < j$, $z_{j'}^{\textsc{Alg}(R')} < z_j^{\textsc{Alg}(R')} = z_j^{\textsc{Alg}(R)}$.

If P^* is a path, then except for the endpoints of the path, the set of positions matched under both $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$ are the same. By parity, either both endpoints end on ads or end on positions. If the former, then the set of positions matched are the same under both $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$, and the above argument applies. If the latter, let j_b be the position incident to the blue edge and j_r that to the red. For all k, $|Z_k^{\textsc{Alg}(R')}| - |Z_k^{\textsc{Alg}(R)}| \leq 1$, since $Z_k^{\textsc{Alg}(R)'}$ can at most contain the extra position j_r. Since $j' < j$, $z_{j'}^{\textsc{Alg}(R')} < z_j^{\textsc{Alg}(R')} \leq z_j^{\textsc{Alg}(R)} + 1$, and by integrality of z_j^M, $z_{j'}^{\textsc{Alg}(R')} \leq z_j^{\textsc{Alg}(R)}$.

(2b) P^* has one more red edge. By parity, one of the endpoints of P^* end on an ad and the other on a position, the latter of which is matched only under $\textsc{Alg}(R')$ but not $\textsc{Alg}(R)$, call this position j_r. Likewise, for P_*, denote by j_b the position matched under $\textsc{Alg}(R)$ but not $\textsc{Alg}(R')$. Other than j_b and j_r, the set of positions matched under $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$ are identical. We now apply the same argument as in the path case over j_b and j_r, yielding $z_{j'}^{\textsc{Alg}(R')} \leq z_j^{\textsc{Alg}(R)}$.

(2c) P^* has one more blue edge. This case is identical to (2b) except the role of P^* and P_* is reversed. □

Theorem 2. *The 4-approximation algorithm is monotonic.*

Proof. Given rewards R and R', consider the probability δ_i of ad a_i being allocated under $\textsc{Alg}(R)$ and $\textsc{Alg}(R')$. If $(a_i, j) \notin \textsc{Alg}(R)$, then $\delta_i(\textsc{Alg}(R)) = 0 \leq \delta_i(\textsc{Alg}(R'))$, since δ_i is a non-negative function. If $(a_i, j) \in \textsc{Alg}(R)$,

$$\begin{aligned} \delta_i(\textsc{Alg}(R)) = (1-q)^{j + z_j^{\textsc{Alg}}(R)} &\overset{(1)}{\leq} (1-q)^{j' + z_j^{\textsc{Alg}}(R)} \\ &\overset{(2)}{\leq} (1-q)^{j' + z_{j'}^{\textsc{Alg}}(R')} = \delta_i(\textsc{Alg}(R')), \end{aligned}$$

where (1) follows from Lemma 4 and (2) follows from Lemma 5. □

We now consider the computation of payments. Given reward vector R and a monotonic allocation function x, if payment p_i is defined as

$$p_i(R) = r_i \delta_i(x(R)) - \int_0^{r_i} \delta_i(x(R)),$$

then the mechanism (x, p) is truthful [4, 20]. Since the placement is a discrete structure, given R, there exists a set R_i of rewards where the allocation changes, i.e., $R_i = \{r_i' | x((r_i' - \epsilon, R_{-i})) \neq x((r_i', R_{-i}))$ and $r_i' < r_i\}$. Let the elements of this set be ordered as $r_i^1 < r_i^2 < \ldots < r_i^K$. Further, let $r_i^0 = 0$, and add it to R_i. The above payment function can be computed from this set via

$$\begin{aligned} p_i(R) = r_i \delta_i(x(R)) - \Big(\sum_{l=1}^{K} (r_i^l - r_i^{l-1}) \delta_i(x((r_i^l, R_{-i})) \\ + (r_i - r_i^K) \delta_i(x(R)) \Big). \end{aligned}$$

Note that for advertisers that are unmatched, their payments are zero. For a matched advertiser a_i where $(a_i, j) \in x(R)$, we can apply binary search over the space of rewards $(0, r_i)$ to find R_i. For each position $s \in S_i$ where $s < j$, we look for the minimum reward $r_i' \in (0, r_i)$ such that $(a_i, s) \in x((r_i', R_{-i}))$. Finding this reward (or determining none exists) requires $O(\log r_i)$ invocations of the 4-approximation algorithm via binary search. The collection of rewards thus found constitutes the set R_i, which requires $O(|S_i|\log r_i)$ invocations of the algorithm. Over all advertisers, we need at most $O(\sum_{1\leq i\leq n}|S_i|\log r_i)$ calls to the algorithm, which is polynomial in size of the input.

Since the 4-approximation algorithm is monotonic, and we can compute the payments in polynomial time, we have a truthful and efficient mechanism that is 4-optimal.

Note that because the PTAS in Section 5 relies on taking the best matching among all possible values of $j_1,\ldots,j_k$, the argument for monotonicity above no longer holds for the PTAS. If the matching produced with given $j_1,\ldots,j_k$ satisfies a stronger version of monotonicity known as *bitonicity*, then the PTAS can be coupled with a suitable payment scheme to create a truthful mechanism [19]. Proving (or disproving) bitonicity of the tiered matching algorithm given j's is a promising direction of future work.

7. A PRACTICAL ALGORITHM

Despite their theoretical significance, polynomial-time approximation schemes are notorious for often being impractical and slow. In the case of the stream ad placement problem, the webserver that compiles the news feed has only a few milliseconds to pick the ads. This is not enough time to even look ahead to see which ads are compatible with the forthcoming pieces of content and perform a global optimization. This means that not only our polynomial-time approximation scheme, but also our 4-approximation algorithm is not practical for this application. What we need is an almost online algorithm that can decide which ad (if any) to place at any position as the user scrolls to that position, with little or no information about what comes after that position. As we will observe later (in Proposition 1), obtaining a proper online competitive algorithm for this problem is provably impossible. However, we use the ideas developed in Section 5 to design an approximation algorithm that processes positions one by one using very little information about the global structure of the problem.

Specifically, we give an online algorithm GREEDY$_R$ parameterized by one real value R, and show that for every instance of the problem, there is a value of R such that GREEDY$_R$ is a constant-factor approximation to the optimal solution of Stream Advertising. The value of R depends on the instance, but in practice, this value can be estimated using historical data. Furthermore, our algorithm has a simple interpretation as a greedy algorithm with a (changing) reserve price, and can be easily explained to the advertisers. Last but not least, GREEDY$_R$ can be paired with a Generalized-Second-Price-style payment scheme (see [10]) that is simple and natural, and even though it is not incentive compatible, it shares some of the desirable properties that has kept GSP the most popular auction mechanism for sponsored search.

Algorithm GREEDY$_R$ proceeds by treating R as an ex ante reserve price, and allocating each position to the highest bidder, if the bid of this bidder multiplied by the probability of reaching that position (ignoring the ads in the stream) is at least R. In effect, this means that the reserve price increases from each position to the next by a factor of $(1-q)^{-1}$. See a formal description of GREEDY$_R$ in Algorithm 1.

Algorithm 1 (GREEDY$_\mathbf{R}$)

```
 1: $A = \{1,\ldots,n\}$
 2: for all positions $j = 1,2,\ldots$ do
 3:     $i^* \to \mathrm{argmax}_{i\in A}\{r_i : j \in S_i\}$
 4:     if $(1-q)^j r_{i^*} < R$ then
 5:         do not assign position $j$ to any advertiser.
 6:     else
 7:         assign position $j$ to advertiser $i^*$.
 8:         $A \to A \setminus \{i^*\}$
 9:     end if
10: end for
```

Theorem 3. *For any instance of* Stream Advertising *there is a value of R such that the output of GREEDY$_R$ is an 8-approximation to the optimal solution.*

Proof. First, we assume, for convenience, that we have no ties in the values, i.e., for any two ad/position pairs (i,j) and (i',j'), $(1-q)^j r_i \neq (1-q)^{j'} r_{i'}$. It is not hard to remove this assumption by using symbolic perturbations in favor of higher slots, but we leave the details to the full version of this paper.

The structure of the proof is as follows: we first prove that the number of ads picked by GREEDY$_R$ changes continuously and monotonically with R, i.e., as R increases, this number decreases, but never by more than one. This means that either there is a value R^* for which GREEDY$_{R^*}$ selects precisely $h_{1/2}$ ads, or at $R^* = 0$ the algorithm picks less than $h_{1/2}$ ads. Next, we prove that the output of GREEDY$_{R^*}$ is a 2-approximation to the maximum weight matching problem with a cardinality constraint of $h_{1/2}$, and therefore, using the arguments in Section 5, it is an 8-approximation to the optimal solution of Stream Advertising.

To prove the first part, consider a value of R such that by slightly increasing R to R^+, the output of the algorithm changes. At some point during the execution of GREEDY$_R$, we must have $(1-q)^j r_{i^*} = R$ in line 4, since otherwise the two executions must be identical. Furthermore, by our no-ties assumption, this happens only once. We now examine the execution of the two algorithms GREEDY$_R$ and GREEDY$_{R^+}$, and prove by induction that in every iteration, the set A of available advertisers in GREEDY$_{R^+}$ is the same as the set A in GREEDY$_R$, plus possibly one advertiser. This obviously holds for all iterations up to (and including) where the tie in line 4 happens. Assume the statement holds after iteration $j-1$ and consider iteration j. In this iteration, either i^* in the two executions is the same, in which case the difference between the A's in the two executions does not change, or i^* in GREEDY$_{R^+}$ is the one advertiser that at the end of iteration $j-1$ is in A in GREEDY$_{R^+}$ but not in GREEDY$_R$. The value of r_{i^*} must be larger in GREEDY$_{R^+}$ than in GREEDY$_R$, and therefore GREEDY$_R$ removes its i^* from A, GREEDY$_{R^+}$ must do similarly with its i^*. Therefore, at the end of iteration j, the difference between the two A's is still either in one advertiser (the i^* in GREEDY$_R$), or the two sets are equal (in case GREEDY$_{R^+}$ removes its i^* but GREEDY$_R$ doesn't). This completes the inductive proof.

This means that at the end of the algorithm, GREEDY$_{R^+}$ has found a placement for all the ads that GREEDY$_R$ has placed, minus possibly one ad. Given this, we can either find an R^* such that GREEDY$_{R^*}$ picks precisely $h_{1/2}$ ads, or that for $R^* = 0$, GREEDY$_{R^*}$ picks at most $h_{1/2}$ ads. In either case, we prove that the output of GREEDY$_{R^*}$ is a 2-approximation to the maximum weight matching problem with cardinality constraint that was defined by the 4-approximation algorithm in Section 5. We prove this using a charging argument. Consider the optimal solution OPT to this matching problem. Let (i, j) be an edge in this matching. If this edge is picked by GREEDY$_{R^*}$, we charge it to itself. If not, it must be that either ad i is matched to another position j' at the time that position j is considered, or j is assigned to another ad i' despite i's availability, or that i does not meet the reserve price for position j. In the first case, the value $w_{ij'}$ of the edge (i, j') must be at least w_{ij} since j' is before j, and therefore we can charge (i, j) to (i, j'). In the second case, the value $w_{i'j}$ of (i', j) must be at least w_{ij} by the definition of the algorithm, and hence we charge (i, j) to (i', j). In the third case, the value of (i, j) is less than or equal to the value of any edge that GREEDY$_{R^*}$ picks, and therefore we can keep such edges until the end and charge them to any edge in the solution of GREEDY$_{R^*}$ that has not been charged to. It is easy to see that this charging scheme charges at most two edges of OPT to any edge in our solution. Therefore, the solution found by GREEDY$_{R^*}$ is a 2-approximation to the cardinality-constrained weighted matching problem. By the argument in Section 5 the solution of this problem is a 4-approximation to STREAM ADVERTISING. $\square$

The parameter R in GREEDY$_R$ captures everything that the algorithm needs to know about the subsequent positions in the input instance. It would have been nice if the algorithm did not require knowing such a value. However, the following proposition shows that without knowing anything about the global structure of the instance, achieving a constant-factor approximation is impossible.

PROPOSITION 1. *For any constant c, there is no online c-competitive algorithm for* STREAM ADVERTISING.

PROOF. Consider the following instance, denoted by I_L: for each $i = 1, 2, \ldots, L$, there is an ad a_i of value M^i for a large constant M that can only be placed at position i. Consider any online c-competitive algorithm A. For any i, A must place a_i in position i, since if it does not, it will get a poor competitive ratio on the instance I_i. This means that on the instance I_L, the value achieved by A is precisely

$$\begin{aligned}\sum_{i=1}^{L}(1-q)^{2i}M^i &< (1-q)^{2L}M^L\sum_{i=0}^{\infty}(1-q)^{-2i}M^{-i} \\ &\leq 2(1-q)^{2L}M^L\end{aligned}$$

for $M \geq 2(1-q)^{-2}$. On the other hand, the optimal algorithm on this instance only picks the ad a_L, achieving a value of $(1-q)^L M^L$. The ratio of the two values is $\frac{1}{2}(1-q)^{-L} > c$ for large L, contradicting c-competitiveness of A. $\square$

Finally, we note that there is a natural payment scheme that can be associated with GREEDY$_R$: charge each advertiser that wins a slot j the maximum of the reserve price $R/(1-q)^j$ and the bid of the second highest bidder for this position. This is similar to the generalized second price (GSP) mechanism for sponsored search [10, 24] in that it charges the advertiser the minimum the advertiser should bid to stay in the same slot. Neither of these two mechanisms are incentive compatible, but the simplicity of the GSP mechanism as well as desirable properties such as the fact that a bidder cannot reduce her price without reducing her allocation has made the mechanism a widely-used mechanism for sponsored search.

8. CONCLUSION

We introduced the problem of stream advertising to study one aspect of the challenges social networking websites face when monetizing news feeds through the insertion of sponsored messages. The primary tension in the problem is between the revenue of monetizing an ad opportunity now versus the decrease in future value due to user quitting the stream. We formalize the problem as a non-linear combinatorial optimization problem, and gave a PTAS that solves the problem through dividing the positions into tiers and solving a maximum matching subproblem in each tier. We also show that when the values of matching depend on advertisers private valuation, a simpler version of the algorithm that can achieve a 4-approximation is monotonic and can be used as a building block to create a truthful mechanism. From a practical standpoint, we investigate online algorithms that do not need to perform matching over the entire potentially infinitely long stream. We show that while there does not exist a constant-competitive algorithm in general, but can derive a 8-competitive algorithm through estimating a reserve value with past history. While our model is aimed at modeling the problem social network faces, it is also applicable to other form of streaming media such as personalized radios and video streams.

There are many interesting open problems that were raised from this study. From an algorithmic standpoint, the main open question in the offline setting is whether STREAM ADVERTISING is computationally hard to solve optimally. Resolving this question will deepen our understanding of the matching with externalities problem. In the online setting, the main question is whether one can make use of statistical knowledge of the items to design more competitive algorithms.

There are also a number of open problems related to incentives. An immediate question from our analysis is whether the allocation according to the PTAS can be turned into a truthful mechanism. A promising direction is to establish the stronger monotonicity condition of bitonicity of the tiered matching algorithm. From a modeling standpoint, we have focused on a mechanism design setting where the set of valid positions is known to the network and only the rewards are private to the advertisers. This is an important scenario to consider as the set of valid positions are usually determined by the context and the content of the ad. However, in situations where an advertiser can select its targeting criteria to change the set of valid positions, these sets now constitute part of the strategy space of the mechanism as well. We believe this richer setting is of interest and is a promising direction for future work.

9. REFERENCES

[1] G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pál. Sponsored search auctions with markovian users. In *Proc. 4th WINE*, pages 621–628, 2008.

[2] G. Aggarwal, A. Goel, and R. Motwani. Truthful auctions for pricing search keywords. In *Proc. 7th EC*, 2006.

[3] N. Alon, I. Gamzu, and M. Tennenholtz. Optimizing budget allocation among channels and influencers. In *WWW '12*, 2012.

[4] A. Archer and E. Tardos. Truthful mechanisms for one-parameter agents. In *Proc. 42nd FOCS*, 2001.

[5] E. Bakshy, D. Eckles, R. Yan, and I. Rosenn. Social influence in social advertising: Evidence from field experiments. In *Proc. 13th EC*, 2012.

[6] N. Buchbinder, K. Jain, and J. S. Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *Algorithms–ESA 2007*, pages 253–264. Springer, 2007.

[7] O. Candogan, K. Bimpikis, and A. Ozdaglar. Optimal pricing in the presence of local network effect. In *Proc. 6th WINE*, 2010.

[8] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, pages 17–33, 1971.

[9] N. R. Devenur and T. P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 71–78. ACM, 2009.

[10] B. Edelmen, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction. *Amer. Econ. Review*, 2007.

[11] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 117–126. IEEE, 2009.

[12] T. Groves. Inventive in teams. *Econometrica*, pages 617–631, 1973.

[13] N. Haghpanah, N. Immorlica, K. Munagala, and V. S. Mirrokni. Optimal auctions with positive network externalities. In *Proc. 12th EC*, 2011.

[14] J. D. Hartline, V. S. Mirrokni, and M. Sundararajan. Optimal marketing strategies over social network. In *WWW '08*, 2008.

[15] D. Kempe, J. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proc. 9th KDD*, 2003.

[16] D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. In *Proc. 4th WINE*, pages 585–596, 2008.

[17] M. Mahdian, H. Nazerzadeh, and A. Saberi. Allocating online advertisement space with unreliable estimates. In *Proc. 8th EC*, 2007.

[18] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized on-line matching. In *Proc. 46th FOCS*, pages 264–273, 2005.

[19] A. Mu'alem and N. Nisan. Truthful approximation mechanisms for restricted combinatorial auctions. In *Proc. 18th AAAI*, 2002.

[20] R. B. Myerson. Optimal auction design. *Math. Oper. Res.*, 6(1):58–73, 1981.

[21] N. Nisan, T. Roughgarden, Éva Tardos, and V. V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[22] P. Papadimitriou, H. Garcia-Molina, P. Krishnamurthy, R. A. Lewis, and D. H. Reiley. Display advertising impact: Search lift and social influence. In *Proc. 17th KDD*, pages 1019–1027, 2011.

[23] M. Todi. Advertising on social networking websites. *Wharton Research Scholars Journal*, 2008.

[24] H. Varian. Position auctions. *Int. J. Industrial Organization*, 2007.

[25] W. Vickrey. Conunterspeculation, auctions and competitive sealed tenders. *J. Finance*, pages 8–37, 1061.

[26] K. C. Wilbur, L. Xu, and D. Kempe. Correcting audience externalities in television advertising. Marketing Science, Forthcoming. Available at `SSRN: http://ssrn.com/abstract=1423702`, September 2013.