Advertisement Allocation and Mechanism Design in Native Stream Advertising

Iftah Gamzu[1] and Iordanis Koutsopoulos[2*]

[1] Amazon Research, Israel
iftah.gamzu@yahoo.com,
WWW home page: https://sites.google.com/site/iftahgamzu/
[2] Athens University of Economics and Business, Greece, Department of Informatics
jordan@aueb.gr,
WWW home page: http://www2.cs.aueb.gr/∼jordan/

Abstract. We study optimal advertisement allocation in native stream advertising, which exhibits notable fundamental differences from traditional web advertising. Our model highlights the influence of distance between consecutively projected ads on user engagement, which implies interesting *negative externalities* between ads: when there are fewer intervening posts between two ads, there is higher ad fatigue and lower user engagement. We study the problem of advertisement selection and placement in a stream so as to maximize social welfare. We fully characterize the computational complexity of the problem by demonstrating that it is tightly connected to a special form of *interval scheduling*. Next, we prove that it is strictly NP-hard (SNP-hard) but can be efficiently approximated to within $1/2$ of the optimum. We complement this result by studying the mechanism design variant of the problem and develop a $1/2$-approximation truthful-in-expectation mechanism. We also consider mechanisms that guarantee a stronger deterministic form of truthfulness. We believe that these results lay a valuable theoretical foundation for further research in the field.

Keywords: Native advertising, mechanism design, ad placement, algorithms.

1 Introduction

Native stream advertising is a form of online advertising in which ads are integrated seamlessly into user experience and visual design of the website. Examples of native ads are suggested posts on Facebook, promoted tweets on Twitter, and sponsored contents on Yahoo media streams. These ads are inserted between posts in a feed that is expended as a user scrolls down the page. Native advertising provides a non-intrusive advertising experience for users, and it has evolved into a multi-billion dollar business with almost 75% of the total US display ad revenue by 2021 [1].

It is crucial to understand the factors that affect user engagement with ads in a stream and derive accurate models for predicting it. It is also important to develop mechanisms

*I. Koutsopoulos acknowledges the support of the project "Research Reinforcement" by the General Secretariat for Research and Technology (GSRT).

for selection and placement of ads, and their pricing. Using an off-the-shelf solution may not be enough. For example, in sponsored search advertising, ads are displayed next to search results in a specified uninterrupted space. Ranking and pricing of ads is realized by the generalized second price (GSP) auction [2]. In native advertising, the notion of a rank is less clear since ads may be placed between different posts with different relevance to ads. Restricting native ads to pre-specified advertising slots misses the huge potential in placing them in a coherent context.

While ad engagement probability in native advertising still depends on attributes inherent to the ad (e.g., ad design and quality, underlying product or service, landing page quality) and attributes related to the user (e.g., user demographics, interests, past activity) as in traditional web advertising, a third type of attributes make native advertising fundamentally different. These relate to how ads are augmented to the content feed, and they are: (*i*) Contextual similarity of ads and posts. For example, an ad for a hotel is more likely to gain attention if shown after a traveling-related article than after a post on politics. (*ii*) Distance between consecutively projected ads. When there are fewer intervening posts between two ads, there is higher ad fatigue and lower user engagement. (*iii*) Position of the ad in the stream. Ads in higher positions are more likely to be seen since user attention decreases as she scrolls down. It is thus imperative to follow a clean-slate approach and address basic questions on ad selection and placement, and on truthfulness of allocation and pricing mechanisms.

A stream publisher provides an ordered sequence of content items $C = \langle c_1, \dots c_n \rangle$ displayed to a user. There is also a set of ads $A = \{a_1, a_2, \dots, a_m\}$, and each ad a_i is associated with a value $b_i \in \mathbb{R}_+$ that represents the bid (amount of money) that the advertiser is willing to pay when a user engages with her ad. Advertisers may specify pay-per-click bids, but there are other engagement types as well. The goal is to supplement the stream C with ads in a way that maximizes the overall value derived from the ads. Formally, let $\pi : A \to C \cup \{\perp\}$ be an assignment rule (algorithm) that specifies, for each ad, the content item (post) after which it is placed, with $\perp$ indicating that the ad does not appear in the stream. We restrict our attention to rules that forbid assignment of two or more ads to the same item, that is, assignments in which ads appear one after the other. The objective is to find an assignment rule π that maximizes the obtained revenue, $\sum_{i=1}^{m} b_i \cdot \Pr(\text{user interacts with } a_i : \pi)$.

We consider three determinants for user engagement: (*i*) Contextual similarity. We model that by the relevance of the ad to the preceding content item. (*ii*) Ad saturation. High rate of ad projections has negative impact on user experience and engagement. Modeling negative externalities among ads has been key issue in the literature, e.g. [3, 4]. We model a new aspect of this by ad distance, i.e., number of content items between an ad and the ad that precedes it. (*iii*) Attention span. User attention decreases as she scrolls down the page, thus ads in higher positions are more likely to be engaged. We model this by the position of the content item that precedes an ad.

Predicting interaction probabilities. Let $d_\pi(a_i)$ be the distance of ad a_i from a preceding ad according to assignment π. If a_i is the first ad in the stream, its distance is simply the number of content items that appear before it. Having the above three determinants in mind, one can reinterpret interaction probability as

$$\Pr(\text{user interacts with } a_i : \pi) \triangleq \Pr(\text{user interacts with } a_i : \pi(a_i) = c_t, d_\pi(a_i) = d) \ . \tag{1}$$

The crucial parameters are c_t, which captures both the content item that appears before a_i and its position t, and d, the distance from the preceding ad.

We assume a model that predicts the probability that the user interacts with ad a_i when placed after item c_t and when in distance d from a preceding ad, for any combination of a_i, c_t, and d. Such model can be built using historical data. A shorter distance between ads implies higher ad fatigue and lower interaction rates. Suppose ad a_i is placed after content item c_t under some assignment; the probability that the user interacts with a_i may only increase if the preceding ad is further distanced, i.e.

$$\Pr(\text{user interacts with } a_i : \pi(a_i) = c_t, d_\pi(a_i) = d) \leq$$
$$\Pr(\text{user interacts with } a_i : \pi(a_i) = c_t, d_\pi(a_i) = d') \ , \ \forall d \leq d' \, .$$

A simple special scenario. Negative externalities between ads, captured by ad distances, make the problem non-trivial. If the probability of user engagement only depends on the relevance of an ad to a content item and its position, an optimal assignment of ads can be attained via the weighted bipartite matching problem [5] on the bipartite graph whose left-side nodes correspond to ads, right-side nodes correspond to content items, and there is an edge between each ad-node a_i and content-item-node c_t of weight equal to the expected value from an assignment of a_i to c_t. The ad distance induces interesting dependencies relating to selection of edges to the solution and the weight contribution of those edges. Specifically, the weight of each edge is no longer fixed but depends on the selection of other edges to the solution. However, deciding which edges to select to the solution inherently depends on their (still unknown) weights. This coupling makes the problem challenging.

1.1 Our contribution

We build on the basic model that has been proposed recently in [6, 7]. We fully characterize the computational complexity of the native ad allocation optimization problem by demonstrating that it is tightly connected to a special form of the interval scheduling problem, known as the *weighted group interval scheduling problem*. Subsequently, we establish that the problem is strictly NP-hard (SNP-hard), namely, it is NP-hard to approximate to within some constant factor. This rules out the possibility of a polynomial-time approximation scheme (PTAS). We also demonstrate that the problem can be efficiently approximated to within $1/2$ of the optimum. These results appear in Section 2.

We proceed to study the mechanism design variant of the problem, where advertisers may be strategic about their bids, and we devise a $1/2$-approximation mechanism that is truthful-in-expectation. We also consider mechanisms that guarantee stronger forms of truthfulness, and develop an efficient deterministic truthful mechanism with logarithmic approximation guarantee. These results are presented in Section 3.

1.2 Related work

Native stream advertising is well-motivated in practice, but only few papers attain to this novel setting (see, e.g., [8]-[11]). The work [8] is the closest to ours. They study a problem of advertising in a stream that incorporates negative externalities between ads. There is a collection of ads with rewards. Each ad can be assigned to a restricted set of positions in the stream. The goal is to find an ad placement that maximizes the overall expected reward. A key parameter is a decay function that captures the probability that the user views an ad. As the stream is augmented with more ads, the distance of the lower ads from the head of the stream increases, and their view probability and expected reward decrease. The authors develop a matching-based constant approximation algorithm and a more involved PTAS, and they show that the former algorithm can underlie a truthful mechanism.

Although the model of [8] and our model have similar flavor, they only intersect on very special cases (like, when ads can be placed only on pre-specified slots). The main distinction is that Ieong et al. capture externalities as a cumulative effect from a reference point, the top of the stream, while we explicitly model externalities between *pairs* of adjacent ads. This captures the extra annoyance induced by clutters of ads. For instance, our model differentiates between an ad assignment that has k ads spread evenly before it and k ads that are cluttered just before it. This is essential for capturing ad fatigue in a more plausible way. It also changes the underlying optimization task and leads to different challenges for truthful mechanisms. For example, we are able to establish that our model does not admit a PTAS, while Ieong et al. successfully design a PTAS for their model.

Our work is also related to ad auctions. The most studied and used model is that of separable click-through-rates [12, 2] which assumes that the click probability of an ad is a product of two quantities, the first about relevance of the ad and the second about the quality of the slot it occupies. One of the main drawbacks is that this cannot account for externalities that ads impose on each other. Several papers proposed various models such as the cascade model to capture externalities between ads [4, 3]. Our work connects with this line of research of modeling and computationally analyzing externalities. Finally, the group interval scheduling problem has received a great deal of attention, see e.g. [13–16], and references therein.

2 Native Advertising Meets Interval Scheduling

We show that our native stream advertising model is equivalent to the weighted group interval scheduling one. We first develop a polynomial-time value-preserving reduction from weighted group interval scheduling to our problem. This reduction proves that our problem is at least as hard as the former problem and that it is SNP-hard [13], that is, NP-hard to approximate to within some constant factor. This rules out the possibility of a PTAS for the problem. We then design a polynomial-time value-preserving reduction in the other direction, from any instance of our problem to an instance of weighted group interval scheduling. This reduction implies that any algorithm for the latter problem can be applied to native stream advertising as well, resulting in an efficient $1/2$-approximation algorithm [14, 15].

The weighted group interval scheduling problem. An input instance for weighted group interval scheduling consists of a collection of groups $\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_m$ such that each group $\mathcal{G}_i$ includes k_i intervals, $\{I_{i1}, I_{i2}, \dots, I_{ik_i}\}$. Each interval I is characterized by a triplet (s_I, t_I, v_I), where s_I and t_I are the start and end points of the interval, and v_I is its value. Intervals are assumed to be open, i.e., the span of I is (s_I, t_I). Also, two intervals I and J intersect (marked as $I \cap J \neq \emptyset$), if there is $p \in (s_I, t_I) \cap (s_J, t_J)$. The goal is to select a maximum-value subset of intervals such that (1) no pair of intervals intersect, and (2) no pair of intervals belong to the same group. We assume that interval endpoints are positive integers and denote the largest integral value of all endpoints by n.

2.1 Hardness of native advertising

We show that any instance of weighted group interval scheduling can be efficiently translated into an instance of native stream advertising in a value-preserving way. Thus, we get that our problem is SNP-hard as the former one is SNP-hard [13]. This rules out the possibility of a PTAS for our problem, unless P=NP.

Reducing interval scheduling to native stream advertising. Given an input instance for weighted group interval scheduling, we construct an input instance for native stream advertising as follows. We first create a sequence $C = \langle c_1, c_2, \dots c_n \rangle$ of n content items. Let $v_{\max} = \max_I v_I$ be the maximum interval value among all intervals of all groups in the instance. We transform each group $\mathcal{G}_i$ to an ad a_i: we set the value of a_i to $b_i = v_{\max}$. Moreover, for each $I \in \mathcal{G}_i$, we provisionally set

$$\Pr(\text{user interacts with } a_i : \pi(a_i) = c_{t_I}, d_\pi(a_i) = d) = \frac{v_I}{v_{\max}} ,$$

for every $d \in \{t_I - s_I, t_I - s_I + 1, \dots, t_I\}$. In case two or more intervals of $\mathcal{G}_i$ wish to set the value of the same interaction probability, which may happen if they share the same end-point, we fix that probability to be the maximum of all values offered by those intervals. Interaction probabilities are set as follows

$$\Pr(\text{user interacts with } a_i : \pi(a_i) = c_t, d_\pi(a_i) = d) = \frac{\max_{I : I \in \mathcal{G}_i, t = t_I, d \in \{t_I - s_I, \dots, t_I\}} v_I}{v_{\max}} . \tag{2}$$

As a final step to the input transformation, we add n dummy ads $a_1^*, \dots, a_n^*$ to the instance, such that each ad a_i^* has an arbitrary value, and $\Pr(\text{user interacts with } a_i^* : \pi(a_i^*) = c_t, d_\pi(a_i^*) = d) = 0$, for every $t \in [n], d \in [t]$. Although these dummy ads are not essential to establish properties of the reduction, they simplify some arguments.

The implied prediction model satisfies the natural monotonicity constraint attaining to the relationship between ad fatigue and the distance parameter. This property follows since we extend the (immediate) interaction probability associated with each interval I (relating to a distance of exactly $t_I - s_I$) to all interaction probabilities having larger distance, and since we select the maximum interaction probability when several probabilities are implied by different intervals.

Reduction properties. The reduction can be implemented in polynomial-time. Any feasible solution for weighted group interval scheduling, and an optimal solution, implies a feasible ads assignment in the newly-created instance, having at least the same value. This follows by assigning the ad associated with each selected interval to the content item corresponding to the end-point of the interval. Formally, each interval $I \in \mathscr{G}_i$ that is selected to the solution defines an assignment of a_i to c_{t_I}. Since the length of I is $d = t_I - s_I$, the expected contribution of this ad to revenue is

$$v_{\max} \cdot \Pr\left(\text{user interacts with } a_i : \pi(a_i) = c_{t_I}, d_\pi(a_i) = d\right) \geq v_{\max} \cdot \frac{v_I}{v_{\max}} = v_I \ , \qquad (3)$$

assuming that there is some ad assigned to c_{s_I}. Note that the inequality is due to the max condition in the probabilities definition (2). Now, the so-called dummy ads come to play. If no ad were assigned to c_{s_I} as part of the above process relating to the intervals in the solution, we select an arbitrary unassigned dummy ad, and assign it to c_{s_I}. Notice that there are always enough dummy ads to augment the initial assignment. Although the contribution of a dummy ad to the social welfare is 0, it makes a_i contribute at least v_I. Hence, the revenue of the resulting ad assignment is at least the value of the interval scheduling solution. Conversely, one can easily verify that given an ads assignment to the newly-created instance, there is an interval scheduling solution with the same value. Details are provided in the extended version [17].

Theorem 1. *The native stream advertising model is SNP-hard.*

Proof. Any hardness result for weighted group interval scheduling follows to our model. In conjunction with the result of Spieksma [13], our model is SNP-hard.

2.2 Algorithms for native stream advertising

We establish that any instance of our native stream advertising model can be efficiently translated into an instance of weighted group interval scheduling in a value-preserving way. This implies that any algorithm for the latter problem can be utilized to construct an ad assignment. In particular, this implies an efficient $1/2$-approximation algorithm for our problem [14, 15].

Reducing native stream advertising to interval scheduling. Given an input instance for native advertising, we construct an input instance for weighted group interval scheduling. We transform every ad a_i to a group $\mathscr{G}_i$ of $\binom{n}{2}$ intervals. Specifically, $\mathscr{G}_i$ has an interval I for every two positions $s_I, t_I \in [n], s_I < t_I$ of the content items sequence. Let $d = t_I - s_I$, the value of the interval $(s_I, t_I) \in \mathscr{G}_i$ is

$$v_I = b_i \cdot \Pr\left(\text{user interacts with } a_i : \pi(a_i) = c_{t_I}, d_\pi(a_i) = d\right) \ .$$

Reduction properties. The reduction can clearly be implemented in polynomial-time. One can verify that any feasible ads assignment for native stream advertising, and in particular, an optimal ad assignment, implies a feasible interval scheduling solution in the newly-created instance with at least the same value. Consider an ad assignment, and suppose that ad a_i is assigned to content item c_t, while being in distance d from a preceding ad, and having an expected contribution v to the social welfare. By the input construction, there must be an interval $I \in \mathscr{G}_i$, characterized by the triplet $(t-d, t, v)$. This interval is selected for the solution. The set of selected intervals is feasible, namely, no pair of intervals intersect or share the same group. Also, the value of the interval scheduling solution is equal to the social welfare of the ads assignment. Conversely, one can show that given an interval scheduling solution to the newly-created instance, one can perform a similar transformation in the opposite direction, and get an ad assignment with at least the same value [17].

Theorem 2. *Our ad model admits an efficient* $1/2$- *approximation algorithm.*

Proof. The reduction above implies that any efficient algorithm for weighted group interval scheduling can be utilized to efficiently construct an ad assignment with at least the same approximation guarantees. One can use the reduction above to translate any native stream advertising instance to a weighted group interval scheduling one, solve the latter instance using any efficient algorithm, and translate the solution back to an ad assignment without any loss of value. In accordance with the algorithmic results of Bar-Noy et al. [14, 15], we get a polynomial-time $1/2$-approximation algorithm for our problem. The entire algorithm is formalized as Algorithm 1.

3 Truthful Native Advertising Mechanisms

We now study the problem from a game-theoretic point of view where advertisers are strategic and dishonest about their bids so as to manipulate the ad assignment algorithm in a way that maximizes their own utility. We aim to develop efficiently computable truthful mechanisms that are robust against manipulation, i.e., every advertiser is rationally motivated to truthfully report its bid. In what follows, we utilize the equivalence in Section 2 to derive an efficient randomized $1/2$-approximation mechanism that is truthful in expectation. We then proceed by developing an efficient deterministic truthful mechanism with logarithmic approximation guarantee.

3.1 Preliminaries

The game-theoretic setting of native advertising is identical to the original model formulation, however advertisers now are *strategic* and may be dishonest about their private value v_i for user engagement with their ad. Advertiser of ad a_i may bid a value b_i, different than her true value, if this can improve her expected utility. Advertisers are *single-parameter* in the sense that their private data consists of a single number. All other parameters in the model, namely the sequence of content items and the set of ads along with user engagement predictions, are known to the stream publisher.

The stream publisher attends to this strategic setting by designing a mechanism. A *mechanism* is a pair (π, p) consisting of an ad assignment algorithm $\pi : A \to C \cup \{\perp\}$ and a payment function $p : A \to \mathbb{R}_+$. The idea is that an appropriate payment function can motivate advertisers to bid truthfully. A mechanism is called *truthful* or *incentive compatible* if no advertiser can gain by being dishonest. The Vickrey-Clarke-Groves (VCG) mechanism [18], is known to be truthful for optimizing social welfare. Nonetheless, implementing VCG requires an exact optimal solution for the underlying optimization problem. Plugging an approximation algorithm into VCG may not yield a truthful mechanism [19]. As the native advertising model is NP-hard, one cannot utilize VCG to get an efficient truthful mechanism. Fortunately, designing truthful mechanisms in single-parameter settings boils down to designing *monotone* algorithms.

Algorithm 1 Native Stream Ads Assignment

Input: A stream of content items $C = \langle c_t \rangle_{t=1}^{n}$, set of ads $A = \{a_i\}_{i=1}^{m}$ with bids $b = \{b_i\}_{i=1}^{m}$, and prediction model specifying $\Pr(a_i, c_t, d) = \Pr(\text{user interacts with } a_i : \pi(a_i) = c_t, d_\pi(a_i) = d)$, for any (a_i, c_t, d)

Output: Ads assignment $\pi : A \to C \cup \{\perp\}$

▷ Step 1: instance translation to interval scheduling (IS)
foreach $a_i \in A$ **do**
  $\mathscr{G}_i \leftarrow \emptyset$, $\pi(a_i) \leftarrow \perp$
  foreach $c_t \in C$, $d \in [t]$ **do**
    $\mathscr{G}_i \leftarrow \mathscr{G}_i \cup \{(t-d, t, b_i \cdot \Pr(a_i, c_t, d))\}$
▷ Step 2: solving IS using a local ratio algorithm [15]
$\mathscr{I} \leftarrow \cup_{i=1}^{m} \mathscr{G}_i$, $\mathscr{L} \leftarrow \emptyset$, $S \leftarrow \emptyset$
while $\mathscr{I} \neq \emptyset$ **do**
  $I \leftarrow (s_I, t_I, v_I) \in \mathscr{I}$ with min t_I, let $\mathscr{G}_\ell$ be I's group
  if $v_I > 0$ **do**
    $\mathscr{L} \leftarrow \mathscr{L} \cup \{I\}$
    foreach $J \in \mathscr{I}$ **do**
      $\bar{v}_J \leftarrow \begin{cases} v_I & \text{if } I \cap J \neq \emptyset \textbf{ or } J \in \mathscr{G}_\ell, \\ 0 & \text{otherwise} \end{cases}$
      $v_J \leftarrow v_J - \bar{v}_J$
  $\mathscr{I} \leftarrow \mathscr{I} \setminus \{I\}$
while $\mathscr{L} \neq \emptyset$ **do**
  $I \leftarrow (s_I, t_I, v_I) \in \mathscr{L}$ with max t_I, let $\mathscr{G}_\ell$ be I's group
  if $I \cap J = \emptyset$ **and** $J \notin \mathscr{G}_\ell$ for every $J \in S$ **do**
    $S \leftarrow S \cup \{I\}$
  $\mathscr{L} \leftarrow \mathscr{L} \setminus \{I\}$
▷ Step 3: solution translation back to ads assignment
foreach $I = (s_I, t_I, v_I) \in S$ **do**
  let $\mathscr{G}_\ell$ be I's group
  $\pi(a_\ell) \leftarrow c_{t_I}$
return π

Definition 1. *An ad assignment algorithm π is* monotone *with respect to the bid of ad a_i if it satisfies the following property: Suppose p_i is the probability that a user interacts with a_i when its bid is b_i (as implied by an ad assignment generated by π), and $\tilde{p}_i$ is the probability that the user interacts with a_i when its bid changes to $\tilde{b_i}$, and bids of all other ads are fixed (again, as implied by a possibly different ad assignment generated by π). If $\tilde{b_i} > b_i$ then $\tilde{p}_i \geq p_i$.*

Monotonicity implies that engagement probability is non-decreasing in user bid.

Theorem 3. ([20]) *If an ad assignment algorithm π is monotone with respect to the bid of each advertiser, then there exists a truthful mechanism (a payment function) that can be efficiently computed using π.*

3.2 Truthfulness in expectation

Let $v_{itd} = b_i \cdot \Pr(\text{user interacts with} a_i : \pi(a_i) = c_t, d_\pi(a_i) = d)$ be the expected value when ad a_i is placed after item c_t while in a distance d from the preceding ad. A packing integer program that captures native advertising is the following:

$$\text{maximize} \sum_{i=1}^{m} \sum_{t=1}^{n} \sum_{d=1}^{t} v_{itd} \cdot x_{itd} \tag{4}$$

$$\begin{aligned} \text{subject to } (1) \quad & \sum_{t=1}^{n} \sum_{d=1}^{t} x_{itd} \le 1 && \forall i \in [m] \\ (2) \quad & \sum_{i=1}^{m} \sum_{t=j}^{n} \sum_{d=t-j+1}^{t} x_{itd} \le 1 && \forall j \in [n] \\ (3) \quad & x_{itd} \in \{0,1\} && \forall i \in [m], t \in [n], d \in [t] \end{aligned}$$

Variable x_{itd} indicates whether ad a_i is placed after content item c_t, while in distance d from a preceding ad. Constraint (1) guarantees that each ad is assigned to at most one item, and (2) ensures that each item is within distance span of exactly one ad.

Packing integer programs have seen special attention in mechanism design (see, e.g., [21]). In [22], a general approach is presented for devising truthful-in-expectation mechanisms for packing domains. Given any α-approximation algorithm that bounds the integrality gap of the corresponding LP-relaxation of the packing integer program, their approach attains a randomized truthful-in-expectation mechanism with the same approximation ratio. A mechanism is *truthful-in-expectation* if an advertiser always maximizes his expected utility by bidding his true value, where expectation is taken over the internal randomization in the mechanism.

This approach can be applied to the native advertising model to yield a randomized truthful-in-expectation $1/2$-approximation mechanism. A $1/2$-approximation algorithm for rounding and bounding the integrality gap of the above LP is implicit in [14]. They presented a deterministic rounding procedure for a restricted version of weighted group interval scheduling that maps to the special case of native advertising, in which the value of an ad is independent of its concrete assignment, i.e., $v_{itd} = v_i$ for all t, d. This rounding procedure can be applied to our problem as well. The key observation to extend their approach is that their procedure which decomposes the fractional solution into a convex combination of integral solutions is independent of the values of ads. We obtain that following theorem.

Theorem 4. *The native stream advertising model admits an efficient randomized* $1/2$-*approximation mechanism that is truthful-in-expectation.*

3.3 Deterministic truthfulness

We try to develop a deterministic truthful mechanism for native stream advertising, whereby an advertiser always maximizes utility by bidding truthfully. Although there exists some literature on weighted group interval scheduling, none of the techniques that lead to constant approximation can be directly utilized to our setting.

The most natural candidate is the greedy one [14, 16] that processes intervals in a non-decreasing end-point order. It tentatively accepts an interval for the solution if its value is at least twice the overall value of previously accepted intervals that intersect with it or share the same group. Once an interval is accepted, all past interfering intervals are rejected from the solution. It is easy to design concrete instances for which

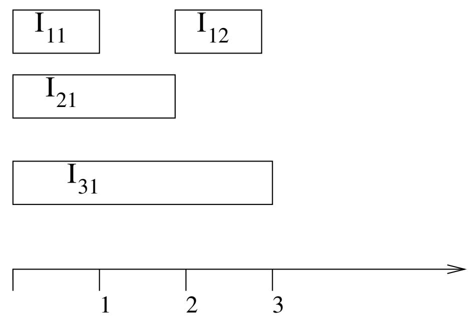

Fig. 1: A native advertising instance represented by ad intervals. Each interval (s,t) corresponds to an assignment of the ad, specified on the left, to position t while there is no assignment of another ad in the $t-s$ positions that precede it. Missing intervals correspond to assignments with an engagement probability (and expected contribution) of 0, with the exception of intervals that extend I_{12} by ad fatigue property. Now, suppose the respective user engagement probability with intervals $I_{11}, I_{12}, I_{21}, I_{31}$ is $1.0, 0.5, 1.0, 1.0$, and observe that when the respective bids of a_1, a_2, a_3 are $1, 2, 4-\varepsilon$, the greedy algorithm results in an ads assignment corresponding to selection of I_{12} and I_{21}. However, if the bid of a_1 increases to $2-\varepsilon$ only I_{31} is selected, breaking monotonicity.

the greedy approach fails due to monotonicity considerations. For example, the native stream advertising instance in Figure 1 illustrates this. When an advertiser increases her bid, there is increase in the value of all assignment intervals associated with her ad. This change in the values of multiple intervals leads to substantial alteration of the ad assignment, which results in non-monotonicity. This phenomenon happens for the greedy algorithm, but we observed it also in other algorithms.

We develop an efficient deterministic monotone algorithm for native stream advertising whose approximation ratio is $O(\log n)$. By Theorem 3, we then attain an efficient deterministic truthful $O(\log n)$-approximation mechanism. Our approach is elementary but requires a few careful observations to maintain monotonicity. We use terminology of weighted group interval scheduling when describing our algorithm.

Our algorithm employs the classify-and-select paradigm. We partition the collection of intervals into $O(\log n)$ pairwise-disjoint classes. For each such class, we separately compute a feasible solution whose overall value is equal to the optimal value attainable from this class. In particular, each class is treated in a completely independent fashion, as if there were no other classes under consideration. The final step is to return the solution with maximum value. If there are several solutions with the same value, the algorithm breaks ties in a consistent way, i.e., it always picks one over the other. We now present the two main components of our algorithm, namely, the classification process, and the single class algorithm.

Intervals classification. This corresponds to a recursive centroid decomposition of line $\{1,2,\ldots,n\}$. Let $\mathscr{I}$ be the collection of all intervals of all groups. The first class of intervals $\mathscr{I}_1$ consists of all intervals that contain the center point of the line $n/2$. To classify the remaining intervals $\mathscr{I} \setminus \mathscr{I}_1$, we recursively apply the previously-described procedure with respect to each of the line segments resulting from the removal of that center point. Specifically, the second class of intervals $\mathscr{I}_2$ consists of all yet-unclassified intervals that contain the center point of any remaining segment. This corresponds to selection of all remaining intervals that contain one of the points $n/4$ or $3n/4$. The remaining classes are defined similarly. In general, class $\mathscr{I}_k$ consists of all intervals that were not selected to $\bigcup_{i=1}^{k-1} \mathscr{I}_i$, and contain one of the points in $\{n/2^k, 2n/2^k, \ldots, (2^k - 1)n/2^k\}$. The recursive process ends as soon as segments consist of a single point. The overall number of levels in the recursion, and equivalently, the number of interval classes is $O(\log n)$.

Single class algorithm. A class is a disjoint union of subsets of intervals, each of which corresponds to a different centroid decomposition. Although those subsets are disjoint, they are not independent from a solution point of view since an interval in a subset may share the same group with an interval in another subset of the class. Thus, we cannot simply focus on computing a feasible solution for a single centroid decomposition and its induced set of intervals, and then merge all the solutions to form one unified solution for that class. We have to consider all class intervals together. We apply the following algorithm to every class of intervals.

Consider some class of intervals $\mathscr{I}_k$. This class can be written as the disjoint union of interval subsets $\mathscr{I}_k^1, \ldots, \mathscr{I}_k^\ell$, where $\mathscr{I}_k^j$ is the subset of intervals that were first separated by the point $jn/2^k$. We construct a weighted bipartite matching instance whose left-side nodes correspond to the subsets of the class, right-side nodes correspond to the groups, and there is an edge between each subset-node $\mathscr{I}_k^j$ and every group-node $\mathscr{G}_i$ if there is an interval $I \in \mathscr{I}_k^j \cap \mathscr{G}_i$. In the simplest form, the weight of this edge is equal to the value of the interval v_I. If there are multiple such intervals, the weight of the edge is the maximum value across all intervals under consideration. We can optimally solve that [5] and return the intervals that are implied by the solution. The solution for the matching problem is a set of edges. Each edge $(\mathscr{I}_k^j, \mathscr{G}_i)$ corresponds to one interval $I \in \mathscr{I}_k^j \cap \mathscr{G}_i$ that defined the weight of that edge. This interval is returned as part of the solution of the class.

Theorem 5. *The classify-and-select approach is a polynomial-time deterministic monotone $O(\log n)$-approximation algorithm.*

Proof. The algorithm is clearly polynomial-time and deterministic. The approximation ratio is also straight-forward. For the proof about monotonicity, the reader is referred to the extended version of the paper [17].

4 Conclusions

We formalized native stream advertising as a combinatorial optimization problem and fully characterized its computational complexity. Native stream advertising is NP-hard to approximate to within some constant factor. We devised a constant-factor approximation algorithm and a truthful-in-expectation mechanism. From an algorithmic mechanism design standpoint, an open question is to resolve the gap between the approximation ratio of truthful-in-expectation mechanisms and deterministic ones. Indeed, randomized mechanisms are more powerful than deterministic ones [23]. From a modeling standpoint, our model is complementary to that in [8] but shares some intersection only on very special cases. A natural open question is to develop a unified computationally tractable model that extends both models.

References

1. Business Insider. Native ads will drive 74% of all ad revenue by 2021, 2016. Available at http://www.businessinsider.com/the-native-ad-report-forecasts-2016-5.
2. S.Lahaie, D.M.Pennock, A.Saberi, and R.V.Vohra. Sponsored search auctions. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, Algorithmic Game Theory, chapter 28. Cambridge University Press, 2007.
3. A. Ghosh and M. Mahdian. Externalities in online advertising. In 17th WWW, pages 161-168, 2008.
4. D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. In 4th WINE, pages 585-596, 2008.
5. H.W.Kuhn.The hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2:83-97, 1955.
6. I. Koutsopoulos. Optimal advertisement allocation in online social media feeds. In 8th HotPOST, pages 43-48, 2016.
7. I. Koutsopoulos and P. Spentzouris. Native advertisement selection and allocation in social media post feeds. In ECML PKDD, pages 588-603, 2016.
8. S. Ieong, M. Mahdian, and S. Vassilvitskii. Advertising in a stream. In 23rd WWW, pages 29-38, 2014.
9. M. Lalmas, J. Lehmann, G. Shaked, F. Silvestri, and G. Tolomei. Promoting positive post-click experience for in-stream yahoo gemini users. In 21th SIGKDD, pages 1929-1938, 2015.
10. N. Barbieri, F. Silvestri, and M. Lalmas. Improving post-click user engagement on native ads via survival analysis. In 25th WWW, pages 761-770, 2016.
11. K. Zhou, M. Redi, A. Haines, and M. Lalmas. Predicting pre-click quality for native advertisements. In 25th WWW, pages 299-310, 2016.
12. G. Aggarwal, A. Goel, and R. Motwani. Truthful auctions for pricing search keywords. In 7th EC, pages 1-7, 2006.
13. F. C. R. Spieksma. On the approximability of an interval scheduling problem. J. Scheduling, 2(5):215-227, 1999.
14. A. Bar-Noy, S. Guha, J. Naor, and B. Schieber. Approximating the throughput of multiple machines in real-time scheduling. SIAM J. Comput., 31(2):331–352, 2001.
15. A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Schieber. A unified approach to approximating resource allocation and scheduling. J. ACM, 48(5):1069- 1090, 2001.
16. T. Erlebach and F. C. R. Spieksma. Interval selection: Applications, algorithms, and lower bounds. J. Algorithms, 46(1):27-53, 2003.
17. I. Gamzu and I. Koutsopoulos, Native Advertising via Interval Scheduling. https://www.dropbox.com/s/slaog50kf0w8ty0/TR-2018-extended.pdf?dl=0.
18. T. Groves. Incentives in teams. Econemetrica, 41(4):617-631, 1973.
19. N.Nisan and A.Ronen. Computationally feasible vcg mechanisms. JAIR,29:19-47, 2007.
20. R. B. Myerson. Optimal auction design. Mathematics of Operations Research, 6(1):58-73, 1981.
21. Y. Azar and I. Gamzu. Truthful unification framework for packing integer programs with choices. In 35th ICALP, pages 833-844, 2008.
22. R. Lavi and C. Swamy. Truthful and near-optimal mechanism design via linear programming. J. ACM, 58(6):25, 2011.
23. N. Nisan and A. Ronen. Algorithmic mechanism design. Games and Economic Behavior, 35(1-2):166-196, 2001.