Prompt Mechanism for Ad Placement over Time

Yossi Azar* and Ety Khaitsin

Blavatnik School of Computer Science,
Tel-Aviv University,
Tel-Aviv 69978, Israel
azar@tau.ac.il, avienda1@gmail.com

Abstract. Consider video ad placement into commercial breaks in a television channel. The ads arrive online over time and each has an expiration date. The commercial breaks are typically of some uniform duration; however, the video ads may have an arbitrary size. Each ad has a private value and should be posted into some break at most once by its expiration date. The player who own the ad gets her value if her ad had been broadcasted by the ad's expiration date (obviously, after ad's arrival date), and zero value otherwise. Arranging the ads into the commercial breaks while maximizing the players' profit is a classical problem of ad placement subject to the capacity constraint that should be solved truthfully. However, we are interested not only in truthfulness but also in a prompt mechanism where the payment is determined for an agent at the very moment of the broadcast. The promptness of the mechanism is a crucial requirement for our algorithm, since it allows a payment process without any redundant relation between an auctioneer and players. An inability to resolve this problem could even prevent the application of such mechanisms in a real marketing process. We design a 6-approximation prompt mechanism for the problem. Previously Cole et al considered a special case where all ads have the same size which is equal to the break duration. For this particular case they achieved a 2-approximation prompt mechanism. The general case of ads with arbitrary size is considerably more involved and requires designing a new algorithm, which we call the Gravity Algorithm.

1 Introduction

Advertising has long since been ubiquitous in the world of trade activities. Recently, with online technologies participating in and transforming the economics realm, the WEB space naturally becomes an ads space. Consequently, mechanisms that help to arrange ads in physical and online space draw our attention. Consider a display ad space with fixed capacity, which contains a new set of ads every day. The ads arrive online over time, each one has parameters of size, value, arrival and expiration date before which it should be posted. Each day ads of a

* Partially supported by the Israeli Science Foundation (grant No. 1404/10) and by the Google Inter-university center for Electronic Markets and Auctions.

G. Persiano (Ed.): SAGT 2011, LNCS 6982, pp. 19–30, 2011.

fixed total size can be published. It is natural to assume that the size, arrival and expiration date are fixed parameters of the ad and cannot be reported falsely. Therefore we regard these parameters as public information. One may think for example of a poster advertising a concert. It has a fixed physical size and the date of the concert event determines the ad's expiration date. The only private information of an ad is its value. The ads are published for the entire day. The player gets her value if her ad had been published by the ad's expiration date (obviously, after the ad's arrival date), and zero value otherwise.

We are interested in a truthful (incentive compatible) mechanism. This is an algorithm which gets an input from selfish players who try to maximize their profit, and motivates them to report their private information truthfully. This goal is usually achieved by means of manipulating the payments collected from players depending on the algorithm's outcome. Our mechanism belongs to the single parameter problems domain, which is a well studied and understood class of problems. The single parameter truthful mechanisms are equivalent to monotone algorithms, which means that the winner would still win if she reports a higher value (keeping other players' values fixed). The mechanism charges every winner a critical price that is determined by a threshold, below which the player loses and above which wins. This critical price can be computed in polynomial time.

A classical technique to achieve truthfulness involves using VCG. Unfortunately, VCG often cannot be applied to online models, since it requires the exact optimality which rarely can be achieved in an online fashion, even with unbounded computational power. Moreover, even when exact optimality can be achieved, the payment cannot be determined at the time of service, since it depends on future events. This suggests the design of *prompt mechanisms*, meaning that a player gets to know the price at the time of publication - unlike the standard online truthful algorithm where the critical price may be determined only in the future.

This model is general and may have numerous and various applications. The ad space may be a physical newspaper sheet with new ads being published on it daily. Another example is a billboard that displays a set of ads on a fixed space with changes every specific time period.

Also, the given space may be a virtual one, and applied to online marketing. For example, Google TV deals with TV companies that offer ad slots, and advertisers that have ads to be published. Google puts together a schedule of ads that fills a commercial break (ads may have different length, but the breaks are typically of the same duration), and sends this schedule to a TV company, which broadcasts it as it is.

It is also worthwhile to mention that our problem can model a buffer management or a broadcast problem. Specifically, we consider the classical model of packets that need to be transmitted through an output port. Each player has a packet with value, length, arrival and expiration date. Every time step packets with bounded total size can be transmitted. The switch has to decide which

packets would be transmitted at each time step while maintaining the size constraint. The goal is to design a truthful algorithm which transmits packets with (approximate) maximum total value.

Prompt mechanisms were introduced in [11]. As discussed there, several reasons for determining the payment at the time of the service exist: first, a player does not know how much money she has after winning until she leaves, and hence cannot participate in another auction. Second, prompt mechanisms can help to resolve some problems of the payment verification. There are two possible ways to ensure a payment and each of them is problematic in a distinct way: for example, if a player pays long after she had won, she may try to get out of the payment. Otherwise, a winner can be demanded to submit an empty check, and the real amount will be deducted later, which requires a considerable trust from the player. In prompt algorithms these situations are avoided, allowing a player to know the price at the moment of her win. Although the concept of prompt models is quite new in the mechanism design field, it has already drawn an attention as can be seen in [11,4].

The model studied in [11] is close to the particular case of ours - specifically, they also consider ads arriving over time. In contrast to our model they considered the special case where the size of each ad is 1, and each day a single ad is published. They have presented a prompt mechanism which yields the approximation ratio of 2 and shown that this is the best possible result for prompt mechanisms. It is natural to ask how to design an algorithm that deals with ads of general sizes. We note that the algorithm of [11] is tailored to the unit size case, hence extending the algorithm to deal with ads of arbitrary sizes is by no means obvious. In addition, the problem of ads with arbitrary size is considerably more complicated than the unit size case, as we would need to deal with integrality issues. Note that the 2 lower bound for the unit size case naturally holds for ads of arbitrary size.

Our Results. We solve the general problem with ads of different sizes.

- Our main result is a truthful, prompt algorithm for online ad placement over time problem, which attains a 6-approximation ratio to the social welfare.
- For relatively small ads ($size \leq \epsilon$) the approximation ratio is $2 + O(\epsilon)$.

We also extend the scope of our study to a more general model, the restricted assignment model, where each ad has a finite set of time periods at which it could be posted. Taking TV ads for example, there are some ads that should be published in mornings or at peak hours only (each time unit is a commercial break). Every player wants her ad to be published during one of those time periods, otherwise she gets zero value. Our algorithm can be easily generalized and modified to answer this case, and attain a 6-approximation ratio (and $2 + O(\epsilon)$ ratio for small ads of size at most ϵ).

Typically, the approximation ratio is calculated for an integral mechanism versus an integral optimum solution. In our case, the achieved approximation ratio also holds for an integral mechanism versus fractional optimum (meaning optimal algorithm that can accept parts of the ads and gain partial profit respectively).

The algorithm treats ads with size $> \frac{1}{2}$ and size $\leq \frac{1}{2}$ separately. The big size ads are treated similarly to the uniform unit size case ([11]). Our main contribution is designing the Gravity Algorithm which focuses on the small ads. It has a tentative schedule of small ads for each day, and always prefers ads with higher density (i.e., the ratio of value to size) even if ad's value is small. The crucial detail of the algorithm is a choice of the time step which a newly arriving ad would be assigned to. It is determined by comparing the densities in the available time steps at a "depth" which depends on the size of the new ad.

Related Works. In a generalized full information setting, the problem is similar to the Multiple Knapsack problem. Multiple Knapsack is a well known problem with a lot of peculiar variations and useful applications ([13]). Its basis, the Knapsack problem, is NPC hard, therefore only approximation algorithms exist. For the Knapsack problem a FPTAS algorithm is available [16] while for the Multiple Knapsack problem there is a PTAS algorithm [8,14]. Our model is intimately related to online MKP with preemption. Designing a truthful model for MKP problem has been considered: the authors of [7] have obtained an approximation ratio of $\frac{e}{e-1} \approx 1.582$ in an offline fashion with bins of the same capacity (this case was also studied by [3]). In an online fashion a $2+\epsilon$ approximation ratio is achieved in [7]. Note that the latter model is very similar to ours; however, our algorithm is prompt while their algorithm does not seem to be extendable to satisfy the promptness requirement.

The special case of our model with uniform unit size ads received a lot of attention in scheduling, packet management and multiple unit auctions frameworks. Online, truthful auction with expiring items was studied by [7], who presented a truthful 2-competitive mechanism. Non-truthful online model of unit jobs scheduling is more widely studied with more thorough results being obtained: the best known deterministic online algorithm has 1.828 competitive ratio [12,17], the lower bound for competitive ratio of deterministic online algorithm is $\frac{2}{\sqrt{5}+1} \approx 1.618$ [2,10]. Giving the randomized setting, the best known online algorithm yields a competitive ratio of $\frac{e}{e-1} \approx 1.582$ [9,5], and it is known that no online algorithm can obtain a ratio better than 1.25 [10]. Note that in offline settings this problem can be solved optimally. If the ads are not of size 1 but of any other uniform size $\frac{1}{k}$ then [9,5] obtain an approximation ratio of $(1-(\frac{k}{k+1})^k)^{-1}$; with $k \to \infty$ the ratio tends to $\frac{e}{e-1} \approx 1.582$. The lower bound for this case is 1.17 [15].

The ad placement problem has already been widely studied in various fashions, including online algorithms and truthful mechanisms, for example [6,1,18]. One of the closest works is [1], studying an online model of ad auctions as a single knapsack problem, and designing a truthful mechanism that maximizes the revenue for this problem. However, most of those works have some differences from ours model. In most of them, for example, there is a factor of frequency (the number of times a single ad should appear), or some other distinctions.

2 The Model

Consider an ads display space of size 1 for a period of n days. Different ads of total size of at most 1 can be published each day. There are players arriving online, each has an ad to be published. An ad is represent by a tuple (s, v, a, e) where $s \in (0, 1]$ refers to its size, $v \in \mathbb{R}^+$ - the players value, and a, e - are the arrival and expiration time. A player wants her ad to be published once before the expiration time. If so, the benefit v is gained; otherwise, the player gets no benefit. The ad's value is a private information of a player, while the rest of the information is public. The algorithm should be incentive compatible. In a single parameter environment it is well that it is equivalent to being a monotone algorithm. Specifically, there is a threshold value above which the player wins and below which she loses. Also, the algorithm has to be prompt, which means the threshold value can be calculated at the very publishing moment. The goal of the algorithm is to maximize the social welfare which is the total value of all published ads.

The further structure of the paper: in the section 2.2 we describe the Gravity algorithm. In the section 3 we prove its truthfulness and promptness. In the section 4 a proof of the approximation ratio is supplied.

2.1 The Gravity Algorithm

Definition 1. Publishing window *of an ad is the time period between its arrival and expiration time.*

Definition 2. *Let density of an ad be* $d = v/s$. *When comparing two distinct ads, we call an* heavier *ad to the one with the higher density.*

The Gravity algorithm maintains a tentative ads schedule for each day. Whenever it handles a new ad's arrival event, it *assigns* it to one of those days. After the assignment there is a test for the ad to enters to the day schedule. If it passes the test it is incorporated into this schedule of that day, otherwise it is rejected. The assignment of the ad to a day is final. Once an ad is assigned to some day either it will be published at that day or it will be rejected. Our algorithm treats *big* ads (ads of size $s > 1/2$) and *small* (ads of size $s \leq 1/2$) separately. Hence, for every day it maintains two alternative schedules: one with a single *big* ad, and the other one of some *small* ads. At the daily publishing event only one of those daily schedules is actually published. The algorithm deals with the tentative daily schedule as if it is a physical bins (with fixed capacity 1). Whenever an ad is assigned to some day t, we say that it is assigned to bin b, where b is bin_{small} or bin_{big} depending on the ad's size. When a new *big* ad enters a bin, the previous ad in that bin is rejected. For the *small bins*, the algorithm maintains the ad of a daily schedule sorted by density. An heavier ad sinks deeper inside the bin and the possibly empty space in the bin is at the top. Every ad in the bin defines an interval in $[0, 1]$ where it is located. When a new small ad enters the bin, the algorithm inserts the ad to the sorted by density list. Note that the interval of the lighter ads in the bin will shift upwards. Moreover, that at this moment the total

size of the ads may exceed 1. Hence all ads above 1 are rejected. There can be at most one ad whose open interval contains 1 and hence it fits the bin's capacity only partially. We call it a *partial* ad and it is also rejected. Yet, the algorithm treats it as if it remained inside, while being "cut" at 1. Its value is reduced so that its density remains the same. Note that although the algorithm treats a partial ad as if it is inside the bin, it does not participate in the publishing event.

Definition 3. *Define D_b^x to be the density at the depth x in the bin b which is the density of the ad whose interval in [0,1] contains the point $1-x$. If the depth x is between two ads, then the it is the lighter density between the two.*

Definition 4. *For any bin b define V_b to be the sum of values of all ads in b.*

Algorithm 1. Gravity Algorithm - contains two parts: Publishing event and Ad arrival event.

Publishing Event: Day t publishing - let b be $bin(t)$
if $\sum_{p \in b_{small}} v(p) \geq v(b_{big})$ **then**
  ads in b_{small} are published and ad in b_{big} is rejected
else
  ad in b_{big} is published and ads in b_{small} are rejected
end if

Ad Arrival Event: An ad $p = (s, v, a, e)$ just arrived
if $s \leq 1/2$ **then**
  Let b_{small} be the bin for which D_b^s is minimal in *publishing window* of p
  if $D_{b_{small}}^s < d_p$ **then** p enters the bin b_{small}
   (ads in this bin are reordered, and some possibly rejected or become *partial*)
  else p is rejected
  end if
else
  Let b_{big} be the bin for which V_b is minimal in *publishing window* of p
  if $V_{b_{big}} < v$ **then** p enters the bin b_{big} and the current ad in b_{big} is rejected
  else p is rejected
  end if
end if

The following observation follows from the definition of the algorithm and the usage of partial ad.

Observation 1. *The density D_b^x for any bin b and $0 \leq x \leq 1$ is non decreasing function throughout the execution of the algorithm.*

Our main result is the following theorem:

Theorem 2. *The Gravity Algorithm is truthful, prompt and 6-competitive.*

The theorem is proved in sections 3 and 4. In section 3 we show truthfulness and promptness. In section 4 we analyze the approximation ratio.

3 Truthfulness

In this section we prove the truthfulness and the promptness of the algorithm.

Lemma 1. *The Gravity Algorithm is truthful (i.e. monotone).*

Proof. We will prove that the algorithm is truthful by showing it is monotone. Suppose that a player i with an ad $p = (s, v, a, e)$ is published as a part of the bin b. We have to show that if she reports $v' \geq v$ then her ad is still published. If $s > 1/2$, then the proof is identical to the one provided in [11] and it is omitted.

Next we focus on small ads, i.e. $s \leq 1/2$. We prove by induction on all ads arriving after p, that the state of all the bins remains exactly the same (albeit the false value report) up to the ads' order inside b. That means that the bin b contains the same ads at the publishing event, and p is published again. Before the arrival of p the flow of the algorithm is exactly as before. At its arrival p is assigned to a certain bin independently of its value. As p would be accepted with the original value, it will also be accepted with a new higher value and density. Now we will consider one by one all further arriving ads and prove that the state after each arrival stays the same as in the original case.

Let $c = (s_c, v_c, a_c, e_c)$ be an arbitrary ad arriving after p. Note that the *density at depth* s_c in the bin b became larger after p's value increased. One of the following three cases holds:

- If originally c is assigned to other bin, it will be assigned there again, and the algorithm takes care of c in the same way.
- If originally c is assigned to the bin b, and ad p is deeper inside the bin than s_c (i.e. the interval of p ends below $1 - s_c$), then we know that c does not compete with p. In the untruthful case, since p is in the same place or perhaps deeper, the algorithm acts in the same way as before.
- If originally c is assigned to the bin b and competes with p then there can be 2 possibilities (notice that in original case p wins, which means that c is rejected when arrived):
  - Ad c is assigned to bin b again; then it will be rejected again.
  - Ad c is assigned now to some other bin h: that means that $D_b^{s_c} \leq D_h^{s_c} \leq D'^{s_c}_b$. We know that originally c was rejected, which means $d_c \leq D_b^{s_c}$, then $d_c \leq D_h^{s_c}$ thus c is rejected from bin h as in the original case.

This flow works for all ads that arrive after p, and then p is still published with the increased value. The algorithm is monotone, and consequently, truthful. ■

3.1 Promptness

Recall that the algorithm is monotone; hence, the critical price is well defined. It is easy to see that every ad can be published only within the bin it was assigned to. Moreover, the publishing of this ad does not depend on the ads that would arrive after the publishing moment. This means that the critical price can be calculated at the very publishing moment - which makes the algorithm prompt.

4 Approximation Ratio

In this section we compute the approximation ratio. We do it separately for *big* and *small* ads, and show that for *small* ads the ratio is 4, and for *big* ads it is 2. Then we show how to combine the analysis and achieve the total approximation ratio of 6 for the entire algorithm.

4.1 Big Ads Approximation Ratio

In this subsection we assume that all ads are *big* (i.e. of sizes larger than $\frac{1}{2}$). For the *big* ads the proof is identical to proof in [11], in which they prove that $\text{Opt}_{\text{big}} \leq 2 \cdot \text{Alg}_{\text{big}}$. Here the proof is omitted.

4.2 Small Ads Approximation Ratio

Within this subsection we assume that all ads are *small* (i.e. of sizes smaller than or equal to $\frac{1}{2}$).

Theorem 3. $\text{Opt}_{\text{small}} \leq 4 \cdot \text{Alg}_{\text{small}}$.

Proof. We fractionally match every ad that was published in Opt to some bin. Then for every bin we prove that the total value of all ads that were matched to that bin is at most 4 times the total value of ads in the bin of Alg .By summing by all ads of Opt and all bins of Alg we get a ratio of at most 4 between the total social benefit and Alg.

Definition 5. *Let O be the set of all ads that were published in* Opt*. Specifically $O = \cup_{i=1}^{i=n} O_i$ where O_i is the set of ads that were assigned to bin i in* Alg*.*

Definition 6. *Given $o \in O$, define $Home(o)$ as the bin where o was published in* Opt*.*

Now we will describe the fractional matching. An ad o in O_i will be matched by one (or sometimes two) of those rules:

Rule 1. If an ad o was rejected at arrival event then it is matched to the bin $Home(o)$.

Rule 2. If an ad o had entered to bin i but was preempted later, then it is fractionally matched to the *Home* bins (in Opt) of the ads in O that entered the bin after o's preemption. Let $O_i = (o_1, o_2, ...o_{last})$ be an ordered set, ordered by arriving timer and assume o is r'th in the order so $o = o_r$. For every $k > r$ we define the *paying* function. Let $o_k \in O_i$ be an ad that was in the bin i and arrived after the ad o_r had been preempted; o_k will *pay* for o_r a fractional part as described below. Note that some fraction of the ad o_r may remain "unpaid" for, and it is the rule 3 that will take care of it. *Pay* is a recursive function defined as follows (it is defined only for $k > r$):

$$pay(o_r, o_k) = Min(size(o_k) - \sum_{l=1}^{r-1} pay(o_l, o_k), size(o_r) - \sum_{l=r+1}^{k-1} pay(o_r, o_l)).$$

Whenever we say that the ad o_k pays for a part of the ad o_r, the meaning is that the part of o_i is viewed as a separate ad of size $pay(o_r, o_k)$ and matched to $Home(o_k)$. Several ads can *pay* for o_r, so that the sum of parts they *paid* for is up to the size of o_r. It may happen as well that one ad pays for several ads but the sum of these ads or ads' parts is always up to the paying ad's size. If no one pays for a part of some ad, this part would be matched by the rule 3.

Rule 3. If an ad o were not preempted, then it is matched to bin i. In addition, the parts of ads which were not matched in rule 2 will also be matched to the bin i.

This way, every ad in O is matched via one (or more) of those matching rules to a bin (or fractionally to several bins). For every bin and every matching rule we will calculate the ratio between the values of the ads that were matched to this bin and the values of the ads published in the bin in Alg. Then we calculate the general ratio.

Definition 7. *Given a set of ads (or parts of ads) A. Then let* $S(A) = \sum_{p \in A} s_p$ *be their total size and* $V(A) = \sum_{p \in A} v_p$ *be their total value. Let* Z_j *be the set of ads published at bin* j *by the algorithm not including the* partial ad *and let* $V_j = V(Z_j)$.

It is enough to show that the total value of all ads that were matched to bin j, does not exceed $4V_j$. We show it by bounding the total value of all ads that were matched to j by every one of the rules:

Rule 1

Let $G_1(j)$ be the set of all ads that were matched by the first rule: they were published in Opt at bin j (since we have fixed j we omit the index and denote it as G_1). These ads were assigned to some bins in Alg and rejected immediately at their arrival event. Let d_{max} be the maximal density of an ad in G_1. We can conclude that d_{max} is at least as the average density of the set G_1. Let an ad $p = (s, v, a, e) \in G_1$ be a one for which $v/s = d_{max}\}$. The ad p was assigned to some bin h. Note that p could have been assigned to bin j, and hence $D_h^s \leq D_j^s$ (when p arrives). By that and by the fact that p was rejected when arrived, we derive that $d_{max} \leq D_h^s \leq D_j^s$. Since $s \leq 1/2$, then at the moment of p's arrival the bin j was filled up to a half with ads of density at least d_{max}. The observation before implies that the density at any depth the bin is non-decreasing. Hence, at j's publishing event the bin was full up to half with ads of density at least d_{max} (this half does not contain the *partial ad*, so we can use V_j). Hence

$$V_j \geq 1/2 \cdot d_{max} \rightarrow d_{max} \leq 2V_j \rightarrow V(G_1) \leq 2 \cdot S(G_1) \cdot V_j.$$

Rule 2

Let $G_2(j)$ be the set of all ads or ads' parts that have been matched by the second rule: they were *paid* by ads or ads' parts that were published

at bin j in Opt (since we have fixed j we omit the index and denote it as G_2). We call the set of those *paying* ads or ads' parts G'_2. For example, if an ad of size 1/2 has *paid* for two other ads, each being of the size 1/6, then the *paying* ad is divided into two different ads of size 1/6 and the same density; the rest 1/6 is ignored. For each $p \in G_2$, define $Payers(p) = \{q \in G'_2 | q \text{ paid on } p\}$. Notice that $s_p = S(Payers(p))$ and $S(G_2) = S(G'_2)$. By the monotone density observation the density of any ad in $Payers(p)$ is at least d_p. Let d_{max} be the maximal density in G_2 and $p = (s, v, a, e) \in G_2$ for which $v/s = d_{max}$. Ad p was assigned to a bin h. Let $q \in Payers(p)$ such that q was assigned to the bin h. Note that q could have been assigned to the bin j and hence at the moment q's arrival $D_h^{s_q} \leq D_j^{s_q}$. At this moment p has been preempted already; this means $d_{max} \leq D_h^{s_p}$ and $s_q \leq s_p$ which implies that $d_{max} \leq D_h^{s_q} \leq D_j^{s_q} \leq D_j^{1/2}$. We conclude that at the moment of q's arrival, the bin j is full up to half with ads of density at least d_{max}. The density inside the bin is monotone non-decreasing over time and at j's publishing event the bin is filled up to a half with ads of a density at least d_{max} (this half does not contain the *partial ad*, so we can use V_j). Hence

$$V_j \geq 1/2 \cdot d_{max} \rightarrow d_{max} \leq 2V_j \rightarrow V(G_2) \leq 2 \cdot S(G_2) \cdot V_j.$$

Rule 3

Let $G_3(j)$ be the set of all ads (or parts of ads) that are currently matched by third rule: they had entered bin j and were either published or preempted but were not paid for (since we have fixed j we omit the index and denote it as G_3). Unlike G_1 and G_2, we view G_3 as evolving over the time.

Lemma 2. *At any time $S(G_3) \leq 1$ where $S(G_3)$ is the sum of the sizes of the ads in G_3. The proof is omitted.*

Now, as we know that $S(G_3) \leq 1$, we will bound $V(G_3)$. We divide G_3 into 2 sets: $G_{in} \cup G_{out} = G_3$, G_{in} are the ads that have been finally published (may include the *partial ad*), and G_{out} are ads that have been preempted. Let P_j be the set of ads published at bin j including (incuding) the *partial ad*. Note that P_j is Z_j union with the the *partial ad* in j. We also divide P_j (ads that were published in Alg at bin j), into 2 groups $P_j = P_j^1 \cup P_j^2$ such that $G_{in} = P_j^1$ and $P_j^2 = P_j - P_j^1$. Clearly $V(G_{in}) = V(P_j^1)$ and we will show $V(G_{out}) \leq V(P_j^2)$. If G_{out} is empty we are done. Else we know that $S(P_j) = 1$, because if an ad was preempted from a bin, this bin will be always filled with ads and a partial ad. By Lemma 2, $S(G_{out}) \leq 1 - S(G_{in}) = 1 - S(P_j^1) = S(P_j^2)$. Let d_{max} be the maximal density of ads in G_{out}, $d_{max} \geq V(G_{out})/S(G_{out})$. Observe that every ad in P_j^2 has at least density d_{max}. We obtain that $V(G_{out}) \leq d_{max} \cdot S(G_{out}) \leq d_{max} \cdot S(P_j^2) \leq V(P_j^2)$. We put it together and conclude:

$$\begin{aligned} V(G_3) &= V(G_{in}) + V(G_{out}) \leq V(P_j^1) + V(P_j^2) = V(P_j) \\ &= V(Z_j) + v_{partial\ ad} = V_j + v_{partial\ ad} \end{aligned}$$

It remains to handle is the *partial ad* that was viewed as a regular ad but eventually does not get published. The size of the *partial ad* is less than half and it has the least density in the bin. Hence $v_{partial\ ad} \leq V_j$ which implies that $V(G_3) \leq 2 \cdot V_j$.

Summary

Now we sum these 3 results together. For all j we have

1. $V(G_1(j)) \leq 2S(G_1(j)) \cdot V_j$
2. $V(G_2(j)) \leq 2S(G_2(j)) \cdot V_j$
3. $V(G_3(j)) \leq 2V_j$

Notice that $G_1(j)$ and $G'_2(j)$ both were published in Opt in bin j, meaning that their size together is up to 1, and as we already know $S(G_2(j)) = S(G'_2(j))$. Then we get,

$$\begin{aligned} V(G_1(j)) + V(G_2(j)) + V(G_3(j)) &\leq 2(S(G_1(j)) + S(G_2(j)))V_j + 2V_j \\ &\leq 2V_j + 2V_j = 4V_j. \end{aligned}$$

Now we sum over all j to obtain:

$$V_{OPT}^{small} = \sum_{j=1}^{n} V(G_1(j)) + V(G_2(j)) + V(G_3(j)) \leq 4 \sum_{j=1}^{n} V_j = 4V_{ALG}^{small}.$$

■

Proof of Theorem 2. The Gravity Algorithm is truthful by the Lemma 1, and also prompt. We have calculated the approximation ratio separately for *big* and *small* ads, and shown that for *small* ads the ratio is 4 in Theorem 3, and for *big* ads it is 2 (the proof for *big* ads is the same as in [11]). This implies that the total approximation ratio of the algorithm is 6 - details are omitted. ■

5 Concluding Remarks

We designed a prompt mechanism for ads with arbitrary sizes that are placed over time. Our mechanism is truthful, prompt and achieves 6-approximation. It would be interesting to know the best approximation for prompt mechanism as the best lower bound is 2. Moreover, for ads of relatively small sizes the lower bound does not hold and it may be possible to get an approximation better than 2 for prompt mechanisms.

Acknowledgements. The authors would like to express sincere gratefulness to Iftah Gamzu for his valuable contribution to the present work.

References

1. Aggarwal, G., Hartline, J.D.: Knapsack auctions. In: Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA (2006)

2. Andelman, N., Mansour, Y., Zhu, A.: Competitive queueing policies for qos switches. In: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2003, pp. 761–770. Society for Industrial and Applied Mathematics, Philadelphia (2003)
3. Azar, Y., Gamzu, I.: Truthful unification framework for packing integer programs with choices. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 833–844. Springer, Heidelberg (2008)
4. Babaioff, M., Blumrosen, L., Roth, A.L.: Auctions with online supply. In: Fifth Workshop on Ad Auctions (2009)
5. Bartal, Y., Chin, F.Y.L., Chrobak, M., Fung, S.P.Y., Jawor, W., Lavi, R.: Online competitive algorithms for maximizing weighted throughput of unit jobs. In: Diekert, V., Habib, M. (eds.) STACS 2004. LNCS, vol. 2996, pp. 187–198. Springer, Heidelberg (2004)
6. Borgs, C., Immorlica, N., Chayes, J., Jain, K.: Dynamics of bid optimization in online advertisement auctions. In: Proceedings of the 16th International World Wide Web Conference, pp. 13–723 (2007)
7. Chekuri, C., Gamzu, I.: Truthful mechanisms via greedy iterative packing. In: Proceedings 12th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 56–69 (2009)
8. Chekuri, C., Khanna, S.: A polynomial time approximation scheme for the multiple knapsack problem, vol. 35, pp. 713–728. Society for Industrial and Applied Mathematics, Philadelphia (2005)
9. Chin, F.Y.L., Chrobak, M., Fung, S.P.Y., Jawor, W., Sgall, J., Tich, T.: Online competitive algorithms for maximizing weighted throughput of unit jobs. 4, 255–276 (2006)
10. Chin, F.Y.L., Fung, S.P.Y.: Online scheduling with partial job values: Does timesharing or randomization help? 37, 149–164 (2003)
11. Cole, R., Dobzinski, S., Fleischer, L.: Prompt mechanisms for online auctions. In: Monien, B., Schroeder, U.-P. (eds.) SAGT 2008. LNCS, vol. 4997, pp. 170–181. Springer, Heidelberg (2008)
12. Englert, M., Westermann, M.: Considering suppressed packets improves buffer management in qos switches. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, pp. 209–218. Society for Industrial and Applied Mathematics, Philadelphia (2007)
13. Fidanova, S.: Heuristics for multiple knapsack problem. In: IADIS AC, pp. 255–260 (2005)
14. Kellerer, H.: A polynomial time approximation scheme for the multiple knapsack problem. In: Hochbaum, D.S., Jansen, K., Rolim, J.D.P., Sinclair, A. (eds.) RANDOM 1999 and APPROX 1999. LNCS, vol. 1671, pp. 51–62. Springer, Heidelberg (1999)
15. Kesselman, A., Lotker, Z., Mansour, Y., Patt-shamir, B., Schieber, B., Sviridenko, M.: Buffer overflow management in qos switches. SIAM Journal on Computing, 520–529 (2001)
16. Lawler, E.L.: Fast approximation algorithms for knapsack problems. In: Proceedings of the 18th Annual Symposium on Foundations of Computer Science, pp. 206–213. IEEE Computer Society, Washington, DC (1977)
17. Li, F., Sethuraman, J., Stein, C.: Better online buffer management. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, pp. 199–208. Society for Industrial and Applied Mathematics, Philadelphia (2007)
18. Zhou, Y., Chakrabarty, D., Lukose, R.: Budget constrained bidding in keyword auctions and online knapsack problems. In: Papadimitriou, C., Zhang, S. (eds.) WINE 2008. LNCS, vol. 5385, pp. 566–576. Springer, Heidelberg (2008)