Journal of Artificial Intelligence Research 59 (2017) 265-310 Submitted 01/17; published 07/17

Adopting the Cascade Model in Ad Auctions: Efficiency Bounds and Truthful Algorithmic Mechanisms

Gabriele Farina GFARINA@CS.CMU.EDU
Computer Science Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213 USA

Nicola Gatti NICOLA.GATTI@POLIMI.IT
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano
Piazza Leonardo da Vinci, 32
20133 Milano, Italy

Abstract

Sponsored Search Auctions (SSAs) are one of the most successful applications of microeconomic mechanisms, with a revenue of about $72 billion in the US alone in 2016. However, the problem of designing the best economic mechanism for sponsored search auctions is far from being solved, and, given the amount at stake, it is no surprise that it has received growing attention over the past few years. The most common auction mechanism for SSAs is the *Generalized Second Price* (GSP). However, the GSP is known not to be *truthful*: the agents participating in the auction might have an incentive to report false values, generating economic *inefficiency* and suboptimal revenues in turn. Superior, efficient truthful mechanisms, such as the *Vickrey-Clarke-Groves* (VCG) auction, are well known in the literature. However, while the VCG auction is currently adopted for the strictly related scenario of contextual advertising, e.g., by Google and Facebook, companies are reluctant to extend it to SSAs, fearing prohibitive switching costs. Other than truthfulness, two issues are of paramount importance in designing effective SSAs. First, the choice of the user model; not only does an accurate user model better target ads to users, it also is a critical factor in reducing the inefficiency of the mechanism. Often an antagonist to this, the second issue is the running time of the mechanism, given the performance pressure these mechanisms undertake in real-world applications. In our work, we argue in favor of adopting the VCG mechanism based on the cascade model with ad/position externalities (APDC-VCG). Our study includes both the derivation of inefficiency bounds and the design and the experimental evaluation of exact and approximate algorithms.

1. Introduction

Sponsored Search Auctions (SSAs), in which several *advertisers* bid to have their *ads* displayed in some slot alongside the search results of a keyword, constitute one of the most successful applications of *microeconomic mechanisms*, with a revenue of about $72 billion in the US alone in 2016 (IAB, 2017). The most common auction mechanism for SSAs is the *Generalized Second Price* (GSP). However, the GSP is known not to be *truthful* (Edelman, Ostrovsky, & Schwarz, 2007) and therefore the best strategy of the bidders may be to make

©2017 AI Access Foundation. All rights reserved.

a bid different from their true value. The untruthfulness of GSP generates inefficiency both in terms of social welfare and auctioneer's revenue. The social welfare provides a measure of the quality of the targeting of ads to users and therefore introducing inefficiency in the social welfare reduces the opportunities for the advertisers to be clicked by the users. On the other side, introducing inefficiency in the auctioneer's revenue reduces the monetary opportunities of the auctioneer itself. The best-known efficient truthful mechanism is the *Vickrey-Clarke-Groves* (VCG). The VCG is currently adopted for the scenario, strictly related to that one of SSAs, of contextual advertising, e.g., by Google (Varian & Harris, 2014) and Facebook (Hegeman, 2010), but it is not adopted in SSAs yet because the common opinion is that the switching costs could be prohibitive. In this perspective, studying the inefficiency of the GSP with respect to the VCG mechanism or a new auction mechanism is crucial to accurately evaluate a potential future switch. However, when designing an effective ad auction, truthfulness is not the only issue to consider. Two other issues are of paramount importance and cannot be disregarded. First, the choice of the user model; not only does an accurate user model better target ads to users, it also is a critical factor in reducing the inefficiency of the mechanism. Often an antagonist to this, the second issue to consider is the running time of the mechanism: it must be very short since the mechanism must be executed online. Thus, if the winner determination problem is hard within a given user model, polynomial-time approximation algorithms will likely be used, mining the efficiency of the VCG mechanism. In this paper, we provide some original contributions about the adoption of the cascade model with position- and ad-dependent externalities in ad auctions. In our study, we explore all the issues aforementioned. More precisely, we study theoretical inefficiency bounds of the GSP and the VCG adopting the separable model with respect to the VCG adopting the cascade model. We then provide exact and approximation algorithms employable in real-world settings.

1.1 User Models and Complexity

The commonly adopted user model in ad auctions is the *position-dependent cascade* (PDC), also called the *separable model*, in which the user is assumed to observe the slots from the top to the bottom and the probability whereby a user observing the ad in slot s continues to observe the ad in slot $s+1$ depends only on slot s (Narahari, Garg, Narayanam, & Prakash, 2009). The optimal allocation can be computed efficiently in a greedy fashion in $O(N \log K)$ time, where N is the number of ads and K is the number of slots. The PDC model is currently adopted by the GSP in SSAs and by VCG in contextual advertising. However, a number of works show that the PDC model gives a non-accurate description of the behavior of the users and that, instead, the user behavior is affected by externalities (Craswell, Zoeter, Taylor, & Ramsey, 2008; Joachims, Granka, Pan, Hembrooke, Radlinski, & Gay, 2007; Goldman & Rao, 2014). Craswell et al. (2008) give an empirical comparison of four user models for organic search results, including the PDC model, with an *ad-dependent cascade* (ADC) model. In the ADC model, the probability whereby a user observing ad a continues to observe the next ad depends on ad a itself.[1] This captures the externality of prematurely terminating the scanning process due to either the user satisfying his information need or

1. Craswell et al. (2008) make the restrictive assumption that the continuation probability from ad a is equal to the probability that the user does not click on ad a. Extensions of this model assume that

the user becoming annoyed with the ads, stopping the scan or switching to organic results. Craswell et al. also show that the ADC model provides the best fit to click logs of a large search engine and they suggest also that the best explanation of the ADC model is that a user stops the scan once he found the needed information. Since the click-through rates in organic search results and ads appear to be of a similar nature (and, so far, the same models have been used for both), the ADC model can achieve a significant improvement over the PDC model when used in ad auctions, as confirmed by empirical results described by Gunawardana and Meek (2008). Similarly to the PDC case, finding the best allocation in the ADP model can be efficiently performed by dynamic programming (Aggarwal et al., 2008; Kempe & Mahdian, 2008).

A natural extension of the ADC model is the *ad/position-dependent cascade* (APDC) model proposed by Kempe and Mahdian (2008), in which the probability whereby a user observing ad a displayed in slot s continues to observe the ad displayed in slot $s+1$, depends on both ad a and slot s. This generalization of the ADC model is motivated by the fact that, in empirical studies, it appears that users have been conditioned to assume results listed in higher positions to be more relevant and therefore the click-through rates will also depend on the position. Although there is no proof, the common belief is that finding the optimal allocation within the APDC model is a hard problem, and therefore it is unlikely that efficient algorithms exist for this task. Gatti and Rocco (2013) provide an algorithm finding the best allocation that can be used in real time only with 40 ads or less, while in real-world applications ads can be in the thousands. The problem admits a constant-ratio $(1-\epsilon)/4$, with $\epsilon \in [0,1]$, approximation algorithm (Gatti & Rocco, 2013; Kempe & Mahdian, 2008), but no approximation algorithm is known to lead to a truthful economic mechanism. Furthermore, the known approximation algorithms can be applied in real time with only 90 ads or less (Gatti & Rocco, 2013) and therefore the application of the APDC model in real-world settings remains an open challenge. Let us remark that, in SSAs, economic mechanisms are usually paired with learning algorithms to estimate the values of the model parameters, as discussed, e.g., in the work by Gatti, Lazaric, Rocco, and Trovò (2015).

Models more accurate than the APDC model include a contextual graph capturing externalities between each pair of ads (Gatti, Rocco, Serafino, & Ventre, 2015; Fotakis, Krysta, & Telelis, 2011). Here, the basic idea is that the probability whereby a user stops to scan may be different for each subsequent pair of ads. In models with a contextual graph, the allocation problem is Poly-APX-hard and the best possible approximation ratio is $\frac{\log K}{K}$, while the best-known approximation ratio leading to a truthful mechanism is $\frac{1}{K}$.

1.2 Untruthful Mechanisms and Inefficiency

It is known that the equilibria of the GSP may be inefficient in terms of social welfare when compared to the VCG outcome. Under the assumption that the user behaves as prescribed by the PDC model, locally-envy-free equilibria and symmetric Nash equilibria of the GSP auction provide the auctioneer a revenue not smaller than the one provided by the VCG. This shows that the Price of Stability (PoS) is 1. However, it is also known that the worst Nash equilibrium may be inefficient with a Price of Anarchy (PoA) of about 1.6 (Leme & Tardos,

the continuation probabilities and the click probabilities may be independent parameters (Aggarwal, Feldman, Muthukrishnan, & Pál, 2008; Kempe & Mahdian, 2008).

2010). The assumption of full information is hardly satisfied in real-world applications, requiring the bidders to have complete information about all the parameters of the opponents, and this makes locally-envy-free equilibria and Nash equilibria non-credible outcomes. A more realistic scenario is when information is uncertain with a Bayesian prior. In this case, the most appropriate solution concept is the Bayes-Nash equilibrium. The social welfare may be inefficient in a Bayes-Nash equilibrium with an upper bound of about 3.1 over the PoA (Leme & Tardos, 2010). Similarly, the revenue for the auctioneer may be inefficient in a Bayes-Nash equilibrium of the GSP with an upper bound of 6 over the PoA (Lucier, Leme, & Tardos, 2012). Furthermore, when no prior is available, bidders use automated bidding strategies to find their best bids. However, these bidding strategies may not even converge to any Bayes-Nash equilibrium and, under mild assumptions, the states they converge to are shown to be arbitrarily inefficient (Evangelos & Telelis, 2010). A very recent experimental work shows that the VCG mechanism is better than the GSP auction in terms of auctioneer's revenue (McLaughlin & Friedman, 2016). When considering externalities—and we recall that empirical studies show that externalities are present in the real world (Goldman & Rao, 2014)—the GSP may be arbitrarily inefficient (Kuminov & Tennenholtz, 2009; Gomes, Immorlica, & Markakis, 2009). In particular, Giotis and Karlin (2008) show that, under the assumption that the user behaves as prescribed by the APDC, the social welfare in the worst Nash equilibrium of the GSP may be arbitrarily worse than the social welfare of the VCG outcome when no restriction over the bidding policy is present, while the ratio may be $\frac{1}{K}$ in the restricted case in which bidders avoid to overbid. Finally, there is a recent increase in the use of additional features (such as larger formats, reviews, maps, or phone numbers) arranged by the search engines on the web page together with the ads to increase the attention of the user. It is known that the GSP behaves poorly in this setting, while the VCG directly applies to this setting as it does in the standard setting and it does not suffer from any efficiency loss (Bachrach, Ceppi, Kash, Key, & Kurokawa, 2014).

1.3 Original Contributions

In this paper, we provide some original contributions, constituted by theoretical results and algorithms, and based on the APDC model and the use of the VCG mechanism. From here on, we call PDC-VCG auction the VCG within the PDC model and APDC-VCG auction the VCG within the APDC model. All the algorithms proposed are experimentally evaluated with the Yahoo! Webscope A3 dataset with 10 or less available slots.

- We assume that the user behaves according to the APDC model and we study the inefficiency of GSP auction and of the PDC-VCG auction in terms of social welfare and auctioneer's revenue with respect to the PDC-VCG auction. We focus on full-information Nash equilibria and we formally derive bounds on Price of Anarchy (PoA) and Price of Stability (PoS), showing that inefficiency can be arbitrarily large and that the PDC-VCG auction may not admit any Nash equilibrium. Our results extend those presented by Giotis and Karlin (2008).

- We propose a novel polynomial-time algorithm, called DOMINATED-ADS, that, exploiting the structure of the problem, prunes the set of ads while preserving the optimal allocation within the APDC model. The experimental evaluation of DOMINATED-ADS shows that the number of ads surviving the prune is logarithmic in N.

- We propose a novel exact exponential-time algorithm called COLORED-ADS. This is inspired by the Color Coding algorithm (Alon, Yuster, & Zwick, 1994) and it finds the optimal allocation in the APDC model. The algorithm is randomized and requires a running time $O(2^K N)$ per random restart. The probability of finding the optimal solution is strictly positive when the number of random restarts is exponential in K. When the success probability is set to 50%, the algorithm is applicable to real-time settings with 5 slots or less and provides an empirical approximation larger than 0.999. Furthermore, the algorithm is maximal-in-its-range, allowing the design of truthful mechanisms, and can be used in an anytime fashion stopping the algorithm once a deadline has been reached, preserving the truthfulness of the mechanism.

- We propose a novel polynomial-time maximal-in-its-range approximation algorithm, called SORTED-ADS, finding an approximate solution in the APDC model. We prove that our algorithm provides an approximation guarantee of $\frac{1}{2}$ in some special instances and we conjecture that the same holds in the general case. Furthermore, we show that in the worst case the approximation cannot be better than $\frac{1}{2}$. Experimentally, we observe that SORTED-ADS provides an empirical average approximation ratio of 0.992 while having compute times of about 0.005 seconds.

2. Problem Formulation

We adopt the same SSA model used by Gatti and Rocco (2013) and Kempe and Mahdian (2008). An SSA without externalities is defined as a tuple $\Gamma = (\mathcal{A}, \mathcal{K}, \{q_a\}_{a\in\mathcal{A}}, \{v_a\}_{a\in\mathcal{A}})$ where:

- $\mathcal{A} = \{1, \ldots, N\}$ is the set of ads. W.l.o.g. we assume each advertiser (i.e., an *agent*) to have a single ad, so each agent $a \in \mathcal{A}$ can be identified with ad a;

- $\mathcal{K} = \{1, \ldots, K\}$ is the set of slots s, ordered from the top to the bottom. We assume w.l.o.g. $K \leq N$;

- $q_a \in [0, 1]$ is the *quality* of ad a, i.e., the probability a user will click ad a once observed;

- $v_a \in V_a \subseteq \mathbb{R}^+$ is the value for agent a when ad a is clicked by a user, while $\mathbf{v} = (v_1, \ldots, v_N)$ is the value profile and, as customary in game theory, $\mathbf{v}_{-a}$ is the value profile of all the agents except agent a.

Furthermore, we introduce the following notation that we use along the paper:

- $\bar{v}_a$ is the product of q_a and v_a, and can be interpreted as the expected value for agent a when the user is observing the corresponding ad;

- $\hat{v}_a \in V_a \subseteq \mathbb{R}^+$ is the value reported by agent a to the auctioneer, while $\hat{\mathbf{v}} = (\hat{v}_1, \ldots, \hat{v}_N)$ is the reported value profile and $\hat{\mathbf{v}}_{-a}$ is the reported value profile of all the agents except agent a;

- Θ is the set of (ordered) allocations θ of ads to slots, where each ad cannot be allocated in more than one slot;

- $\text{ads}(\theta) \subseteq \mathcal{A}$ denotes the subset of ads allocated in θ;
- $\text{slot}_\theta(a)$ denotes the slot in which ad a is allocated, if any;
- $\text{ad}_\theta(s)$ denotes the ad allocated in slot s.

The APDC model assumes the user to have a Markovian behavior, starting to observe the slots from the first (i.e., slot 1) to the last (i.e., slot K) where the transition probability from slot s to slot $s+1$ is given by the product of two parameters:

- (*ad-dependent externalities*) $c_a \in [0,1]$ is the *continuation probability* of ad a displayed in slot s;
- (*position-dependent externalities*) $\lambda_s \in [0,1]$ is the *factorized prominence* of slot s (it is assumed $\lambda_K = 0$).

An SSA within the APDC model is a tuple $\Gamma = (\mathcal{A}, \mathcal{K}, \{q_a\}_{a\in\mathcal{A}}, \{v_a\}_{a\in\mathcal{A}}, \{c_a\}_{a\in\mathcal{A}}, \{\lambda_s\}_{s\in\mathcal{K}})$.

The click through rate, $\text{CTR}_\theta(a) \in [0,1]$, of ad a in allocation θ is the probability a user will click ad a and it is formally defined as

$$\text{CTR}_\theta(a) = q_a \ \Lambda_{\text{slot}_\theta(a)} \ C_\theta(a),$$

where

$$C_\theta(a) = \prod_{s < \text{slot}_\theta(a)} c_{\text{ad}_\theta(s)}, \qquad \Lambda_s = \prod_{k<s} \lambda_k.$$

Parameter Λ_s is commonly named *prominence* (Kempe & Mahdian, 2008).[2] Easily, in the PDC model, $c_a = 1$ for every $a \in \mathcal{A}$. Given an allocation θ, its social welfare is defined as:

$$\text{SW}(\theta) = \sum_{a \in \text{ads}(\theta)} v_a \ \text{CTR}_\theta(a) = \sum_{a \in \text{ads}(\theta)} \bar{v}_a \ \Lambda_{\text{slot}_\theta(a)} \ C_\theta(a).$$

The problem we study is the design of an economic mechanism $\mathcal{M}$, composed of

- an *allocation function* $f : \times_{a\in\mathcal{A}} V_a \to \Theta$, and
- a *payment function* for each agent a, $p_a : \times_{a'\in A} V_{a'} \to \mathbb{R}$.

Each agent a has a linear utility $u_a(v_a, \hat{\mathbf{v}}) = v_a \ \text{CTR}_{f(\hat{\mathbf{v}})}(a) - p_a(\hat{\mathbf{v}})$, in expectation over the clicks and pays $p_a(\hat{\mathbf{v}})/\text{CTR}_{f(\hat{\mathbf{v}})}(a)$ whenever its ad is clicked. Notice that $p_a(\hat{\mathbf{v}})$ is the payment in expectation over the clicks and therefore represents the average payment per impression. We are interested in mechanisms satisfying the following properties:

Definition 1 (Incentive compatible (Truthful))**.** *Mechanism $\mathcal{M}$ is dominant strategy incentive compatible (DSIC) if reporting true values is a dominant strategy for every agent, formally, if $u_a(v_a, (v_a, \hat{\mathbf{v}}_{-a})) \geq u_a(v_a, (\hat{v}_a, \hat{\mathbf{v}}_{-a}))$ for every $v_a, \hat{v}_a, \hat{\mathbf{v}}_{-a}$.*

2. Let us remark that, in the work of Kempe and Mahdian (2008), the authors use 'λ_s' to denote the prominence that, in the present paper, we denote with 'Λ_s'. Our definition forces the prominence Λ_s to be monotonically non-increasing in s. Instead, Kempe and Mahdian directly assume the prominence to be monotonically non-increasing in s. This assumption makes our model and that one studied by Kempe and Mahdian equivalent.

Definition 2 (Individually rational)**.** *Mechanism $\mathcal{M}$ is individually rational (IR) if no agent acting truthfully prefers to abstain from participating to the mechanism rather than participating, formally, if $u_a(v_a, (v_a, \hat{\mathbf{v}}_{-a})) \geq 0$ for every $v_a, \hat{\mathbf{v}}_{-a}$.*

Definition 3 (Weakly budget balance)**.** *Mechanism $\mathcal{M}$ is weakly budget balanced (WBB) if the mechanism is never in deficit, formally, if $\sum_a p_a(\hat{\mathbf{v}}) \geq 0$ for every $\hat{\mathbf{v}}$.*

Definition 4 (Computationally tractable)**.** *Mechanism $\mathcal{M}$ is computationally tractable when both f and p_a are computable in polynomial time.*

Definition 5 (Maximal-in-its-range)**.** *Allocation function f is maximal in its range if f returns an allocation maximizing the social welfare among a given subset of allocations that is independent of the agents' reports, formally, if $f := \arg\max_{\theta \in \Theta'} \mathrm{SW}(\theta)$ where the choice of $\Theta' \subseteq \Theta$ does not depend on $\hat{\mathbf{v}}$.*

The most common DSIC mechanism is the VCG mechanism, that is defined as follows:

$$f(\hat{\mathbf{v}}) = \arg\max_{\theta \in \Theta} \mathrm{SW}(\theta)$$

$$p_a(\hat{\mathbf{v}}) = \max_{\theta \in \Theta} \left\{ \left(\sum_{\substack{a' \in \mathrm{ads}(\theta) \\ a' \neq a}} \hat{v}_{a'} \ \mathrm{CTR}_\theta(a') \right) - \left(\sum_{\substack{a' \in \mathrm{ads}(f(\hat{\mathbf{v}})) \\ a' \neq a}} \hat{v}_{a'} \ \mathrm{CTR}_{f(\hat{\mathbf{v}})}(a') \right) \right\}.$$

We recall that, when f is maximal in its range, VCG-like payments can be used, obtaining a DSIC mechanism (Nisan & Ronen, 2000).[3] Let us remark that if f is maximal in its range and computable in polynomial time, then also the VCG-like payments are computable in polynomial time requiring one to call f for K times.

3. Inefficiency of GSP and PDC-VCG

In this section, we study the inefficiency of the GSP auction and of the PDC-VCG auction when the user behaves as prescribed by the APDC model. We employ the following bounds for the social welfare over PoA and PoS respectively (notice that the social welfare is defined in expectation over the clicks).

$$\overline{\mathrm{PoA}} = \max_{\Gamma} \max_{\theta^*_{\mathcal{M}} \in \mathrm{nash}(\mathcal{M},\Gamma)} \frac{\mathrm{SW}(\theta_{\mathrm{VCG}}(\Gamma))}{\mathrm{SW}(\theta^*_{\mathcal{M}})}$$

$$\overline{\mathrm{PoS}} = \max_{\Gamma} \min_{\theta^*_{\mathcal{M}} \in \mathrm{nash}(\mathcal{M},\Gamma)} \frac{\mathrm{SW}(\theta_{\mathrm{VCG}}(\Gamma))}{\mathrm{SW}(\theta^*_{\mathcal{M}})}$$

$$\underline{\mathrm{PoS}} = \min_{\Gamma} \min_{\theta^*_{\mathcal{M}} \in \mathrm{nash}(\mathcal{M},\Gamma)} \frac{\mathrm{SW}(\theta_{\mathrm{VCG}}(\Gamma))}{\mathrm{SW}(\theta^*_{\mathcal{M}})}$$

where:

3. VCG-like payments differ from VCG payments only for the range. In VCG payments, the range is composed of all the possible allocations Θ. In VCG-like payments, the range is the same used for the allocation function.

- $\mathcal{M}$ is either the GSP auction or the PDC-VCG auction;
- Γ is an SSA instance within the APDC model;
- $\theta_{\mathrm{VCG}}(\Gamma)$ is the best allocation returned by the APDC-VCG auction when all the agents are truthful;
- $\mathrm{nash}(\mathcal{M}, \Gamma)$ is the set of all the allocations achievable by some full-information Nash equilibrium of mechanism $\mathcal{M}$ with SSA instance Γ when the users behave as prescribed by the APDC model;
- $\theta^*_{\mathcal{M}}$ is one allocation of $\mathrm{nash}(\mathcal{M}, \Gamma)$.

In words:

- $\overline{\mathrm{PoA}}$: it expresses the maximum inefficiency—w.r.t. the optimal social welfare—of a Nash equilibrium over all the instances and therefore for every auction instance the inefficiency cannot be larger than $\overline{\mathrm{PoA}}$;
- $\overline{\mathrm{PoS}}$: it expresses the maximum inefficiency—w.r.t. the optimal social welfare—of the Nash equilibrium maximizing the social welfare over all the instances and therefore for every auction instance there is at least a Nash equilibrium with an inefficiency not larger than $\overline{\mathrm{PoS}}$;
- $\underline{\mathrm{PoS}}$: it expresses the minimum inefficiency—w.r.t. the optimal social welfare—of a Nash equilibrium over all the instances and therefore for every auction instance the inefficiency is at least $\underline{\mathrm{PoS}}$.

We define the same indices for the auctioneer's revenue, in which we substitute the value of the social welfare with the auctioneer's revenue (notice that the revenue is defined in expectation over the clicks). Furthermore, we study both the unrestricted case in which agents may overbid and the restricted case in which no agent overbids. The inefficiency in these two cases may be dramatically different and, even if in principle agents may overbid if it is the best to do for them, it is improbable they will do since they could be charged payments larger than their valuation and thus their final utility would be strictly negative. Finally, let us observe that both $\overline{\mathrm{PoA}}$ and $\overline{\mathrm{PoS}}$ are commonly studied in literature (Giotis & Karlin, 2008), while $\underline{\mathrm{PoS}}$ is not. We investigate this last index here since, in the specific case of the auctioneer's revenue, it allows one to show whether there is a Nash equilibrium more efficient than the revenue obtained by the truthful mechanism. Here, we do not analyze the $\underline{\mathrm{PoA}}$ index (whose definition is omitted, but it easily follows from the definitions of the indices above), and leave its analysis open for future research.

The results are summarized in Table 1. The GSP, as previously showed, always admits a pure-strategy Nash equilibrium and the inefficiency upper bounds are linear or arbitrarily large in K. Surprisingly, there are instances in which at some Nash equilibrium the GSP provides a revenue arbitrarily larger than that one provided by the APDC-VCG auction. The PDC-VCG auction always admits a pure-strategy Nash equilibrium (in which the agents truthfully report their types) in the restricted non-overbidding case, while in the unrestricted case a Nash equilibrium (even in mixed strategies) may not exist. In terms of inefficiency, the

PDC-VCG auction presents bounds even worse that those of the GSP. We notice that these results also hold when, instead of considering full-information Nash equilibria, we consider more realistic settings in which information is uncertain and therefore Bayes-Nash equilibria are more appropriate.

	$\overline{\text{PoA}}$		$\overline{\text{PoS}}$		$\underline{\text{PoS}}$	
	Ovb	No ovb	Ovb	No ovb	Ovb	No ovb
Social Welfare	∞ ♣	K ♣	$\geq K/2, \leq K$ ♠	K ♠	1	1
Revenue	∞	∞	$\geq K/2$	$\geq K$	0	0

(a) GSP auction vs. APDC-VCG auction.
The GSP always admits a pure Nash equilibrium, as showed by Giotis and Karlin (2008).
The proofs of the bounds marked with ♣ are provided by Giotis and Karlin (2008). The bounds marked with ♠ are claimed by Giotis and Karlin (2008), but no proof is given in that paper (we provide here the proofs).

	$\overline{\text{PoA}}$		$\overline{\text{PoS}}$		$\underline{\text{PoS}}$	
	Ovb	No ovb	Ovb	No ovb	Ovb	No ovb
Social Welfare	(∞)	$\geq K$		$\geq K$	(1)	1
Revenue	(∞)	∞		∞	(0)	≤ 1

(b) PDC-VCG auction vs. APDC-VCG auction.
In the restricted non-overbidding case, the PDC-VCG auction always admits a pure Nash equilibrium in which each agent reports truthfully the type, while in the unrestricted case a Nash equilibrium (even in mixed strategies) may not exist. For the 'Ovb' case, we report between '( )' the value of the bounds restricted to the subset of instances admitting a Nash equilibrium when known.

Table 1: Summary of inefficiency results ('ovb' means 'overbidding').

3.1 Analysis of the GSP Auction

In the GSP, ads are allocated according to decreasing reported values: assuming $\hat{v}_1 \geq \ldots \geq \hat{v}_N$, the allocation function maps ad a to slot $s = a$, for each $a = 1, \ldots, K$, while all the remaining ads are not displayed. Every agent $a = 1, \ldots, K-1$ is charged a price $p_a = q_{a+1}\hat{v}_{a+1}$.[4] Agent K is charged a price $p_K = 0$ if $K = N$, or $p_K = q_{K+1}\hat{v}_{K+1}$ otherwise. The GSP is well known not to be DSIC when users behave as prescribed by the PDC model. The same result holds also when users behave as prescribed by the APDC model, this model being a generalization of the PDC model. In order to evaluate the effectiveness of the GSP, we study the inefficiency of its Nash equilibria, starting from the analysis of $\overline{\text{PoA}}$. Since bounds over $\overline{\text{PoA}}$ for the social welfare have already been studied, we focus on the auctioneer's revenue, so far unexplored. We state the following results. For the sake of presentation, we report some proofs of the theoretical results stated in this section and in the next section in Appendix A.

Lemma 1. *Both in the restricted and in the unrestricted case, the $\overline{PoA}$ of the auctioneer's revenue in the GSP when users behave as prescribed by the APDC model is unbounded.*

4. Throughout the paper we consider only payments in expectation over the clicks. Cost-per-click payments can be easily obtained by dividing expected payments by the click trough rate of the ad.

We now turn our attention to $\overline{\text{PoS}}$. As aforementioned, Giotis and Karlin (2008) provide some claims over the upper bounds of $\overline{\text{PoS}}$ both in restricted and unrestricted case, but no proof is provided. For the sake of completeness, we provide here the proofs.

Lemma 2. *In the restricted case, the $\overline{PoS}$ of the social welfare in the GSP auction when users behave as prescribed by the APDC model is $\geq K/2$.*

Lemma 3. *In the restricted case, the $\overline{PoS}$ of the auctioneer's revenue in the GSP auction when users behave as prescribed by the APDC model is $\geq K/2$.*

We now turn our attention to $\underline{\text{PoS}}$.

Lemma 4. *Both in the restricted and in the unrestricted case, the $\underline{PoS}$ of the social welfare in the GSP when users behave as prescribed by the APDC model is 1.*

Lemma 5. *Both in the restricted and in the unrestricted case, the $\underline{PoS}$ of the auctioneer's revenue in the GSP when users behave as prescribed by the APDC model is 0.*

The above results show that the GSP auction is largely inefficient also in terms of auctioneer's revenue.

3.2 Analysis of the PDC-VCG Auction

The PDC model is very similar to the APDC: the difference lies in the fact that $c_a = 1$ for each ad $a \in \mathcal{A}$. Payments are calculated according to the rules attaining the VCG mechanism, thus providing an IR, WBB, DSIC mechanism under the assumption that users experience ad fatigue due only to the positions of the slots and not to the allocated ads.

We start with the following lemma.

Lemma 6. *In the unrestricted case, the PDC-VCG auction may not admit any (even mixed-strategy) Nash equilibrium when users behave as prescribed by the APDC model.*

Instead, in the restricted case in which no agent overbids, the truthful strategy is always a Nash equilibrium as showed in the following lemma.

Lemma 7. *In the restricted case, the PDC-VCG auction always admits a truthful Nash equilibrium when users behave as prescribed by the APDC model.*

Proof. Suppose without loss of generality that the value profile is sorted in non-increasing order, that is $\mathbf{v} = (v_1, \ldots, v_N)$ with $v_1 \geq \cdots \geq v_N$. By bidding truthfully (i.e., $\hat{\mathbf{v}} = \mathbf{v}$) the allocation calculated by the PDC-VCG auction is $\bar{\theta} = (1, \ldots, N)$. In order to show that truthful bidding is a Nash equilibrium for the agents even when the actual underlying behavioral model for the users is the APDC model, we need to prove that no agent has an incentive to deviate.

Let $\text{pcp}(a, \hat{\mathbf{v}})$ denote the per-click payment charged to each agent a by the mechanism, when the reported types are $\hat{\mathbf{v}}$. We introduce the following symbols:

$$u_a^{\text{pdc}}(v_a, \hat{\mathbf{v}}) = \text{CTR}_{f(\hat{\mathbf{v}})}^{\text{pdc}}(a)(v_a - \text{pcp}_a(\hat{\mathbf{v}})),$$

$$u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}) = \text{CTR}_{f(\hat{\mathbf{v}})}^{\text{apdc}}(a)(v_a - \text{pcp}_a(\hat{\mathbf{v}})),$$

where $\text{CTR}_\theta^{\text{pdc}}(a)$ and $\text{CTR}_\theta^{\text{apdc}}(a)$ represent the click-through rates for ad a in allocation θ when users behave as prescribed by the PDC and the APCD model, respectively. When no agent overbids, all the quantities above are non-negative. Furthermore, by the properties of the PDC-VCG auction, we know that

$$u_a^{\text{pdc}}(v_a, \mathbf{v}) \geq u_a^{\text{pdc}}(v_a, \hat{\mathbf{v}}) \tag{1}$$

for all agents a and all reported value profiles $\hat{\mathbf{v}}$. Inequality 1 can also be rewritten as

$$\frac{\text{CTR}_{\bar{\theta}}^{\text{pdc}}(a)}{\text{CTR}_{f(\hat{\mathbf{v}})}^{\text{pdc}}(a)} \geq 1. \tag{2}$$

Now, suppose that a deviates from his previous truthful bid $\hat{v}_a = v_a$, and call $\hat{\mathbf{v}}_a$ the resulting reported value profile. Since agents do not overbid, a non-truthful bid can only result in agent a being allocated to a slot $b > a$. The resulting allocation $f(\hat{\mathbf{v}}_a)$ is therefore equal to $\hat{\theta}$, with the only difference that ad a gets moved from position a to position b. We are interested in showing that this deviation does not pay off, that is

$$u_a^{\text{apdc}}(v_a, \mathbf{v}) \geq u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a).$$

To this end, notice that if $u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a) = 0$ the thesis follows immediately. Therefore, we focus on the case $u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a) > 0$; we have

$$\begin{aligned}
\frac{u_a^{\text{apdc}}(v_a, \mathbf{v})}{u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a)} &= \frac{\text{CTR}_{\bar{\theta}}^{\text{apdc}}(a)}{\text{CTR}_{f(\hat{\mathbf{v}}_a)}^{\text{apdc}}(a)} \\
&= \frac{\text{CTR}_{\bar{\theta}}^{\text{pdc}}(a) \prod_{i=1}^{a-1} c_i}{\text{CTR}_{f(\hat{\mathbf{v}}_a)}^{\text{pdc}}(a) \prod_{i=1}^{a-1} c_i \prod_{i=a+1}^{b-1} c_i} \\
&= \frac{\text{CTR}_{\bar{\theta}}^{\text{pdc}}(a) \prod_{i=1}^{a} c_i}{\text{CTR}_{f(\hat{\mathbf{v}}_a)}^{\text{pdc}}(a) \prod_{i=1}^{b-1} c_i}.
\end{aligned}$$

Since $b > a$, we have $\prod_{i=1}^{a} c_i \geq \prod_{i=1}^{b-1} c_i$. Furthermore, Inequality 2 guarantees that $\text{CTR}_{\bar{\theta}}^{\text{pdc}}(a) \geq \text{CTR}_{f(\hat{\mathbf{v}}_a)}^{\text{pdc}}(a)$, so that we conclude

$$\frac{u_a^{\text{apdc}}(v_a, \mathbf{v})}{u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a)} \geq 1 \quad \iff \quad u_a^{\text{apdc}}(v_a, \mathbf{v}) \geq u_a^{\text{apdc}}(v_a, \hat{\mathbf{v}}_a)$$

as we wanted to show. □

Now, we analyze the inefficiency of the Nash equilibria when they exist. We start from the analysis of the $\overline{\text{PoA}}$ for the social welfare.

Lemma 8. *In the unrestricted case, when a Nash equilibrium exists, the $\overline{PoA}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Lemma 9. *In the restricted case, the $\overline{PoA}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is $\geq K$.*

We focus on the analysis of the $\overline{\text{PoA}}$ for the auctioneer's revenue, obtaining similar results.

Lemma 10. *In the unrestricted case, when a Nash equilibrium exists, the $\overline{PoA}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Lemma 11. *In the restricted case, the $\overline{PoA}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

We now turn our attention to the analysis of $\overline{\text{PoS}}$.

Lemma 12. *In the restricted case, the $\overline{PoS}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is $\geq K$.*

Lemma 13. *In the restricted case, the $\overline{PoS}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

We leave the analysis of the $\overline{\text{PoS}}$ in the unrestricted case (assuming the equilibrium exists) open. We now turn our attention to the analysis of $\underline{\text{PoS}}$.

Lemma 14. *Both in the restricted and unrestricted case, the $\underline{PoS}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is 1.*

Lemma 15. *In the unrestricted case, when a Nash equilibrium exists, the $\underline{PoS}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC is 0.*

Lemma 16. *In the restricted case, the $\underline{PoS}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is ≤ 1.*

The above results show that the PDC-VCG auction has several drawbacks: a Nash equilibrium may not exist and when it exists it may be largely inefficient both in terms of social welfare and auctioneer's revenue.

4. DOMINATED-ADS Algorithm

We present an algorithm meant to reduce the number of ads, while preserving the optimality of the solution of the social welfare maximization problem. The central observation is that, under certain circumstances, given two ads a, b with parameters $(\bar{v}_a, c_a)$ and $(\bar{v}_b, c_b)$ respectively, it is possible to establish *a priori* that, in any optimal allocation a is allocated above b; whenever this is the case we say that ad a "dominates" ad b. As an example, consider two ads a and b, satisfying the condition $(\bar{v}_a > \bar{v}_b) \wedge (c_a > c_b)$: in accordance with intuition, a simple exchange argument shows that a dominates b. A weaker sufficient condition for deciding whether a dominates b is given in Lemma 17.

Definition 6. *Given an allocation problem defined over the set of slots $\mathcal{K}$, we consider a modified version of such allocation problem in which no ad can be displayed in the first $i-1$ slots and $\lambda_s = 1$ for every $s \leq i$. Let* $\mathrm{alloc}(i, K)$ *the optimal social welfare of this modified allocation problem.*

Lemma 17. *Let $\lambda_{\max} = \max\{\lambda_s : s \in \mathcal{K}\}$, and let B be a (weak) upper bound of the quantity*

$$\tilde{B} = \max_{i \in \{1,\dots,K-1\}} \{\lambda_i \,\mathrm{alloc}(i+1, K)\}. \tag{3}$$

Furthermore, let $\mathcal{D} = [0, \lambda_{\max}] \times [0, B]$. Given two ads a, b with parameters $(\overline{v}_a, c_a)$ and $(\overline{v}_b, c_b)$, consider the affine function $w_{a,b} : \mathcal{D} \to \mathbb{R}$, defined as

$$w_{a,b}(x, y) = \det \begin{pmatrix} x & -y & 1 \\ 1 & \overline{v}_b & c_b \\ 1 & \overline{v}_a & c_a \end{pmatrix}.$$

If the minimum of $w_{a,b}$ over $\mathcal{D}$ is greater than 0, then a dominates b.

Proof. Without loss of generality, we introduce a fictitious slot, say $K+1$, with $\lambda_K = 0$, representing the case in which an ad is not displayed. By simple calculations, it can be observed that, given an allocation θ and two allocated ads a, b such that b is allocated above a (notice that a may be allocated in any possible slot below the slot of b and therefore even in slot $K+1$), swapping a and b leads to an improvement of the social welfare when the following condition holds:

$$\overline{v}_a + c_a\, D + \overline{v}_b\, c_a\, E > \overline{v}_b + c_b\, D + \overline{v}_a\, c_b\, E$$

where:

$$D = \left(\sum_{s=\mathrm{slot}_\theta(b)+1}^{\mathrm{slot}_\theta(a)-1} q_{\mathrm{ad}_\theta(s)}\, v_{\mathrm{ad}_\theta(s)}\, C_\theta(\mathrm{ad}_\theta(s))\, \Lambda_s \right) / \left(C_\theta(b)\, \Lambda_{\mathrm{slot}_\theta(b)} \right), \quad E = \frac{C_\theta(a)\, \Lambda_{\mathrm{slot}_\theta(a)}}{C_\theta(b)\, \Lambda_{\mathrm{slot}_\theta(b)}}.$$

Notice that both D and E do not depend on the parameters (v, q, c) of a and b.

Ad a will be always swapped with ad b independently of θ (i.e., independently of the slots in which they are allocated and of how the other ads are allocated) and therefore ad a dominates ad b, if the above inequality holds for every feasible value of D and E.

More precisely, D is lower bounded by 0 and upper bounded by

$$\lambda_b \,\mathrm{alloc}(\mathrm{slot}_\theta(b) + 1, K) \leq \max_{i \in \{1\dots,K-1\}} \{\lambda_i \,\mathrm{alloc}(i+1, K)\} = \tilde{B};$$

the value E is lower bounded by 0 and upper bounded by $\lambda_{\max}$. Rearranging the above inequality and replacing D with y and E with x, we obtain:

$$x\,(\overline{v}_b\, c_a - \overline{v}_a\, c_b) + y\,(c_a - c_b) + \overline{v}_a - \overline{v}_b = w_{a,b}(x, y).$$

This concludes the proof. □

Algorithm 1 DOMINATED-ADS

1: **procedure** DOMINATED-ADS(ads, slots)
2: Determine $\mathcal{D}$, as defined in Lemma 17
3: For each ad a, compute $|\text{dom}(a)|$
4: Discard all ads a having $|\text{dom}(a)| \geq K$

We will use the notation $a \prec b$ to denote that ad a dominates ad b, in the sense of Lemma 17. Note that $\prec$ defines a partial order over the set of ads. Since $w_{a,b}$ is an affine function defined on a convex set, it must attain a minimum on one of the vertices of $\mathcal{D}$, hence the following holds:

Lemma 18. *If the four numbers $w_{a,b}(0,0)$, $w_{a,b}(0,B)$, $w_{a,b}(\lambda_{\max},0)$, $w_{a,b}(\lambda_{\max},B)$ are all positive, then $a \prec b$.*

We introduce the following definition.

Definition 7. *The set of* dominators *of an ad a is the set $dom(a) = \{b \in \mathcal{A} : b \prec a\}$.*

The following lemma is central to our algorithm, as it expresses a sufficient condition for discarding an ad from the problem:

Lemma 19. *If $|\text{dom}(a)| \geq K$, then ad a can be discarded, as it will never be chosen for an allocation.*

This suggests a straightforward algorithm to determine the set of safely deletable ads, as streamlined in Algorithm 1. Lemma 20 characterizes the complexity of the algorithm.

Lemma 20. *The* DOMINATED-ADS *algorithm has a runtime complexity of $O(N^2+T(N,K))$, where $T(N,K)$ is the runtime complexity of computing $\mathcal{D}$ on Line 2, and a memory complexity of $O(N + M(N,K))$, where $M(N,K)$ is the memory complexity of computing $\mathcal{D}$ on Line 2.*

Notice that iterating the algorithm could lead to successive simplifications of the ads set, as the upper bound B may decrease in successive runs of DOMINATED-ADS. However, the maximum number of iterations is N, since, if no ad has been discarded, then B does not change and no further ad can be discarded in future iterations.

We show two methods, respectively named CONST-λ and DECOUPLE, for calculating a "good" upper bound B that is needed in Line 2 in order to determine $\mathcal{D}$. The first method (CONST-λ) has a computational complexity of $O(NK)$ time, while DECOUPLE runs in $O(N + K \log K)$ time and makes use of the Fast Fourier Transform (FFT) algorithm (Cooley & Tukey, 1965). Both methods require linear, i.e. $O(N+K) = O(N)$, memory. Of course, when K is small enough, both algorithms can be run and the smallest upper bound can be taken as the value of B. Once $\mathcal{D}$ has been computed, the algorithm computes the number of dominating ads, $|\text{dom}(a)|$ for all ads $a \in \mathcal{A}$ (Line 3) and removes all ads having at least K dominators (Line 4), in accordance with Lemma 19.

The following sections delve into the details of CONST-λ (Section 4.1) and DECOUPLE (Section 4.2), as well as a fast method to compute $|\text{dom}(a)|$ (Section 4.3).

4.1 const-λ Algorithm

The rationale behind the const-λ algorithm is that an upper bound for $\text{alloc}(k, K)$ is the optimal allocation of the problem where all the λ values are replaced with

$$\lambda_{k,\max} = \begin{cases} \max\{\lambda_k, \dots, \lambda_{K-1}\} & \text{if } k < K, \\ 1 & \text{otherwise.} \end{cases}$$

This value can be exactly computed in $O(NK)$ time by first sorting ads in decreasing order in $\bar{v}/(1 - \lambda_{k,\max} c)$, as described by Kempe and Mahdian (2008).

Lemma 21. *The* const-λ *algorithm has a runtime complexity of* $O(N \log N + NK)$ *and a memory complexity of* $O(NK)$.

4.2 decouple Algorithm

The main idea behind the decouple algorithm is to repeatedly use the rearrangement inequality in order to upper-bound the value of the social welfare of the optimal allocation, given the problem instance, as shown in Lemma 22.

Lemma 22. *Given a problem instance* Γ*, we can upper-bound the value of the social welfare that can be obtained in any valid allocation by the quantity*

$$\bar{v}_1^* + \bar{v}_2^* \, c_1^* \, \lambda_1 + \cdots + \bar{v}_{K-k+1}^* \, (c_1^* \dots c_{K-1}^*) \, (\lambda_1 \dots \lambda_{K-1}),$$

where the $\bar{v}^*$ *values represent the* $\bar{v}$ *values of the ads, ordered in decreasing order (i.e.* $\bar{v}_1^*$ *is the maximum among all the* $\bar{v}$ *values); same for the* c^* *values.*[5]

Proof. Without loss of generality, assume $\theta = (1, \dots, K)$ is the optimal allocation. Then, we have:

$$\text{SW}(\theta) = \bar{v}_1 + \bar{v}_2 \, c_1 \, \lambda_1 + \cdots + \bar{v}_K \, (c_1 \, \dots \, c_{K-1}) \, (\lambda_1 \, \dots \, \lambda_{K-1}).$$

Since

$$c_1 \, \lambda_1 \geq (c_1 \, c_2) \, (\lambda_1 \, \lambda_2) \geq \cdots \geq (c_1 \, \dots \, c_{K-1}) \, (\lambda_1 \, \dots \, \lambda_{K-1}),$$

by the rearrangement inequality (Hardy, Littlewood, & Polya, 1952) we have

$$\text{SW}(\theta) \leq \bar{v}_1^* + \bar{v}_2^* \, c_1 \, \lambda_1 + \cdots + \bar{v}_K^* \, (c_1 \, \dots \, c_{K-1}) \, (\lambda_1 \, \dots \, \lambda_{K-1}).$$

Now, we have

$$\bar{v}_1^* \geq \bar{v}_2^* \, \lambda_1 \geq \cdots \geq \bar{v}_K^* \, (\lambda_1 \, \dots \, \lambda_{K-1}),$$

so that by using the rearrangement inequality once more we conclude

$$\begin{aligned} \text{SW}(\theta) &\leq \bar{v}_1^* + \bar{v}_2^* \, c_1 \, \lambda_1 + \cdots + \bar{v}_K^* \, (c_1 \, \dots \, c_{K-1}) \, (\lambda_1 \, \dots \, \lambda_{K-1}) \\ &\leq \bar{v}_1^* + \bar{v}_2^* \, c_1^* \, \lambda_1 + \cdots + \bar{v}_K^* \, (c_1^* \, \dots \, c_{K-1}^*) \, (\lambda_1 \, \dots \, \lambda_{K-1}). \end{aligned}$$

□

5. To avoid possible misunderstandings, let us notice that the two orders over $\bar{v}$ and c are taken independently. This means that, given ad i with parameters $(\bar{v}_i, c_i)$, $\bar{v}_i$ and c_i can be in different positions (or ranks) of the two respective orders.

Corollary 1. *We have, with the same symbols of Lemma 22:*

$$\text{alloc}(k, K) \leq \bar{v}_1^* + \bar{v}_2^*\, c_1^*\, \lambda_k + \cdots + \bar{v}_K^*\, (c_1^* \,\ldots\, c_{K-1}^*)\, (\lambda_k \,\ldots\, \lambda_{K-1}).$$

For our convenience, we introduce the symbol $\text{UB}_{\text{decouple}}(k)$, $k \in \mathcal{K}$, defined as

$$\text{UB}_{\text{decouple}}(k) = \bar{v}_1^* + \bar{v}_2^*\, c_1^*\, \lambda_k + \cdots + \bar{v}_K^*\, (c_1^* \,\ldots\, c_{K-1}^*)\, (\lambda_k \,\ldots\, \lambda_{K-1}),$$

so that we can generally write

$$\text{alloc}(k, K) \leq \text{UB}_{\text{decouple}}(k)$$

for all $k \in \mathcal{K}$.

Remark 1. *Notice that we can easily compute $\bar{v}_1^*, \ldots, \bar{v}_K^*$ in $O(K \log K + N)$ time using a linear selection algorithm—such as the Introselect algorithm (Musser, 1997)—and then sorting the topmost K values by an efficient sorting algorithm (the same holds for the c^* values).*

Once we have the values of $\text{UB}_{\text{decouple}}(k)$ for all $k = 1, \ldots, K$, we can simply plug them in Equation 3 obtaining an upper bound for $\tilde{B}$, i.e. a suitable value for B.

We now show that it is possible to compute the values $\text{UB}_{\text{decouple}}(1)$, …, $\text{UB}_{\text{decouple}}(K)$ in time sub-quadratic in N. We define

$$a_k = \bar{v}_k^*\, (c_1^* \,\ldots\, c_{k-1}^*), \quad b_k = \lambda_1 \,\ldots\, \lambda_{k-1}.$$

In particular, $a_1 = \bar{v}_1^*$ and $b_1 = 1$. Notice that we can assume without loss of generality that $b_k \neq 0$ for every $k \in \{1, \ldots, K\}$. We also let

$$d_k = b_k\, \text{UB}_{\text{decouple}}(k).$$

It is easy to prove that the vector $\mathbf{d} = (d_K, \ldots, d_1)$ is the (discrete) convolution of vectors $\mathbf{a} = (a_1, \ldots, a_K)$ and $\mathbf{b} = (b_K, \ldots, b_1)$. Indeed, for every $k \in \{1, \ldots, K\}$:

$$\begin{aligned}
(\mathbf{a} * \mathbf{b})_k &= \sum_{i=1}^{k} (\mathbf{a})_i\, (\mathbf{b})_{k-i+1} = \sum_{i=1}^{k} a_i\, b_{K-k+i} \\
&= \sum_{i=1}^{k} \bar{v}_i^*\, (c_1^* \,\ldots\, c_{i-1}^*)\, (\lambda_1 \,\ldots\, \lambda_{K-k+i-1}) \\
&= b_{K-k+1}\, \text{UB}_{\text{decouple}}(K - k + 1) \\
&= d_{K-k+1} \\
&= (\mathbf{d})_k
\end{aligned}$$

where $(\mathbf{z})_k$ is the k-th element of a generic vector $\mathbf{z}$. This means that we can determine $\mathbf{c}$ in $O(K \log K)$ time using the FFT algorithm (Cooley & Tukey, 1965). Once we have $\mathbf{c}$, it is trivial to compute $\text{UB}_{\text{decouple}}(k)$ for $k = 1, \ldots, K$ in $O(K)$ time. The final complexity for the algorithm is then $O(K \log K + N)$.

Lemma 23. *The* DECOUPLE *algorithm has a runtime complexity of $O(N + K \log K)$, and a memory complexity of $O(N + K)$.*

4.3 Fast Dominance Computation

A naïve implementation of Line 3 in Algorithm 1 tests every ad a against all the other ads, keeping track of the number of dominators of a. This algorithm requires $O(N^2)$ time and is likely to be efficient enough only for instances up to a few thousand ads. As we now show, it is possible to compute the number of dominators of an ad a in $O(\log N)$ amortized time using the dynamic fractional cascading technique (Mehlhorn & Näher, 1990), leading to an overall complexity $O(N \log N)$.

In order to efficiently compute $|\mathrm{dom}(a)|$ for each ad a, we begin with this simple lemma:

Lemma 24. *For any fixed $(x, y) \in \mathcal{D}$, the order $\prec_{(x,y)}$ over the set of ads, defined as*

$$a \prec_{(x,y)} b \iff w_{a,b}(x, y) > 0,$$

is a total order.[6]

Proof. We have, by the properties of the determinant:

$$w_{a,b}(x,y) = \det \begin{pmatrix} x & -y & 1 \\ 1 & \bar{v}_b & c_b \\ 1 & \bar{v}_a & c_a \end{pmatrix} = \det \begin{pmatrix} x & -y & 1 \\ 1 - c_b\, x & \bar{v}_b + c_b\, y & 0 \\ 1 - c_a\, x & \bar{v}_a + c_a\, y & 0 \end{pmatrix} =$$
$$= (1 - c_b\, x)(\bar{v}_a + c_a\, y) - (1 - c_a\, x)(\bar{v}_b + c_b\, y).$$

Hence, $w_{a,b}(x, y) > 0$ if, and only if,

$$\frac{1 - c_b\, x}{\bar{v}_b + c_b\, y} > \frac{1 - c_a\, x}{\bar{v}_a + c_a\, y}.$$

As we assumed x and y to be fixed parameters, we see that $\prec_{(x,y)}$ defines a total order over the set of ads. □

Lemmas 18 and 24 together prove that the domination partial order, $\prec$, is the intersection of the four total orders $\prec_{(0,0)}$, $\prec_{(0,B)}$, $\prec_{(\lambda_{\max},0)}$ and $\prec_{(\lambda_{\max},B)}$. This immediately suggests the idea of working with ranks; in particular, we define $\rho_{(x,y)}(a)$ as the rank of ad a in the total order $\prec_{(x,y)}$, and introduce the 4-dimensional vector:

$$\mathbf{rk}(a) = \begin{pmatrix} \rho_{(0,0)}(a) \\ \rho_{(0,B)}(a) \\ \rho_{(\lambda_{\max},0)}(a) \\ \rho_{(\lambda_{\max},B)}(a) \end{pmatrix}.$$

The condition of Lemma 18 is then equivalent to the following:

Lemma 25. *Given two ads a and b, $\mathbf{rk}(a) < \mathbf{rk}(b)$ iff $a \prec b$.*

6. We assume a deterministic tie-breaking condition is used to handle the case when $w_{a,b}(x, y) = 0$. Hence, from now on we will assume without loss of generality that $w_{a,b}(x, y) \neq 0$ for all $(x, y) \in \mathcal{D}$ and $a, b \in \mathcal{A}$.

Here $<$ denotes component-wise comparison. Using Lemma 25, we see that

$$\mathrm{dom}(a) = \{b \in \mathcal{A} : \mathbf{rk}(b) < \mathbf{rk}(a)\}.$$

In other words, we can cast the problem of counting the number of ads dominating a to a geometric dominance counting problem in four dimensions. This problem is widely studied in literature, with many results showing algorithms computing dominance counts for every point in the point set in $O(N \operatorname{poly}\log N)$ time. For instance, we could use a 4-dimensional range tree (Bentley, 1979) to calculate $|\mathrm{dom}(a)|$ for every ad a in batch in $O(N \log^4 N)$ time.

To lower the exponent of the log factor, we exploit the structure of the problem a bit further. We define the two sets:

$$\mathrm{dom}^-(a) = \{b \in \mathcal{A} : b \prec a \ \wedge \ c_b \leq c_a\}$$
$$\mathrm{dom}^+(a) = \{b \in \mathcal{A} : b \prec a \ \wedge \ c_b > c_a\}.$$

It is trivial to verify that $\mathrm{dom}^-(a) \cup \mathrm{dom}^+(a) = \mathrm{dom}(a)$ and $\mathrm{dom}^-(a) \cap \mathrm{dom}^+(a) = \emptyset$, i.e. that $\mathrm{dom}^-(a)$ and $\mathrm{dom}^+(a)$ are a partition of $\mathrm{dom}(a)$. Hence,

$$|\mathrm{dom}(a)| = |\mathrm{dom}^-(a)| + |\mathrm{dom}^+(a)|.$$

Notice that

$$\frac{\partial}{\partial y} w_{a,b}(x, y) = c_a - c_b.$$

This means that when $c_b \leq c_a$, we have $w_{(a,b)}(x, y) \geq w_{(a,b)}(x, 0)$, for all $(x, y) \in \mathcal{D}$. Analogously, when $c_b > c_a$, we have $w_{(a,b)}(x, y) \geq w_{(a,b)}(x, B)$ for all $(x, y) \in \mathcal{D}$. As a consequence, once we focus on one of these two cases (i.e., $c_b \leq c_a$ or $c_b > c_a$), the problem of counting the number of ads dominating a can be cast to a geometric dominance counting problem in two dimensions. We introduce the following restricted ranks:

$$\mathbf{rk}^-(a) = \begin{pmatrix} \rho_{(0,0)}(a) \\ \rho_{(\lambda_{\max},0)}(a) \end{pmatrix}, \ \mathbf{rk}^+(a) = \begin{pmatrix} \rho_{(0,B)}(a) \\ \rho_{(\lambda_{\max},B)}(a) \end{pmatrix}.$$

We formalize the observation above in the next lemma whose proof follows from the above observation:

Lemma 26. *Given two ads a and b:*

- $b \in \mathrm{dom}^-(a) \iff \mathbf{rk}^-(b) < \mathbf{rk}^-(a)$*;*
- $b \in \mathrm{dom}^+(a) \iff \mathbf{rk}^+(b) < \mathbf{rk}^+(a)$.

Assume we have a data structure RT supporting each of the following operations in $O(\log N)$ time:

- RT.INSERT($\mathbf{v}$) insert the 2-dimensional point $\mathbf{v}$ in the structure; point $\mathbf{v}$ is not already present.
- RT.ERASE($\mathbf{v}$) deletes the 2-dimensional point $\mathbf{v}$ from the structure; point $\mathbf{v}$ is guaranteed to be present. Furthermore, after the first call to SC.ERASE is made, the structure no longer accepts calls to DS.INSERT.

- RT.DOMINANCE($\mathbf{v}$) counts the number of points $\mathbf{w}$ in the structure, satisfying $\mathbf{w} < \mathbf{v}$.

Such a data structure is well-known in literature. It basically consists of a 2-dimensional range tree combined with the technique of dynamic fractional cascading (Mehlhorn & Näher, 1990). The RT data structure can be used to compute $|\mathrm{dom}(a)|$ for every ad a in batch in $O(N \log N)$ time, as streamlined in Algorithm 2.

Algorithm 2 FAST-DOMINATED-ADS

```
1: procedure FAST-DOMINATED-ADS(ads, slots)
2:     Determine 𝒟, as defined in Lemma 17
3:     RT⁺ ← new RT
4:     RT⁻ ← new RT
5:     for each ad a ∈ 𝒜 do
6:         RT⁻.INSERT(rk⁻(a))
7:     for each ad a ∈ 𝒜, sorted according to decreasing c, do
8:         |dom⁺(a)| ← RT⁺.DOMINANCE(rk⁺(a))
9:         |dom⁻(a)| ← RT⁻.DOMINANCE(rk⁻(a))
10:        |dom(a)| ← |dom⁺(a)| + |dom⁻(a)|
11:        RT⁺.INSERT(rk⁺(a))
12:        RT⁻.ERASE(rk⁻(a))
13:    Discard all ads a having |dom(a)| ≥ K
```

Hence, the following lemma holds:

Lemma 27. *The* FAST-DOMINATED-ADS *algorithm has a runtime complexity of* $O(N \log N + T(N,K))$*, where* $T(N,K)$ *is the runtime complexity of computing* $\mathcal{D}$ *on Line 2, and a memory complexity of* $O(N + M(N,K))$*, where* $M(N,K)$ *is the memory complexity of computing* $\mathcal{D}$ *on Line 2.*

5. COLORED-ADS Algorithm

We present an algorithm that can determine an optimal allocation in time polynomial in the number of ads N and exponential in the number of slots K. As K is generally small (typically ≤ 10), this algorithm is likely to be suitable for real-world applications. The key insight for the algorithm is that the problem of finding the allocation maximizing the social welfare can be cast to an instance of the well-known problem of finding a maximal K-vertex weighted simple path in a undirected graph. Efficient algorithms for the latter problem can, therefore, be used to solve our problem.

We introduce the definition of a *right-aligned allocation.*

Definition 8. *Given an ordered sequence* $S : a_1, \ldots, a_n$ *of ads of length* $n \leq K$*, we say that the* right-aligned allocation *of* S *is the allocation mapping the ads in* S *to the last* n *slots, in order, leaving the first* $K - n$ *slots vacant.*

Consider the complete undirected graph $\mathcal{G}$ having the available ads for vertices; every simple path $\pi : a_1, \ldots, a_n$ of length $n \leq K$ in $\mathcal{G}$ can be uniquely mapped to the right-aligned allocation of π, and *vice versa.* Following this correspondence, we define the *value*

of a simple path in $\mathcal{G}$ as the social welfare of the corresponding right-aligned allocation. In order to prove that the problem of finding a maximal K-vertex weighted simple path in $\mathcal{G}$ is equivalent to that of finding an optimal allocation, it is sufficient to prove that there exists (at least) one optimal allocation leaving no slot vacant, i.e. using all of the K slots. To this end, we introduce the following lemma.

Lemma 28. *In at least one optimal allocation all the available slots are allocated.*

Proof. Let θ be any optimal allocation for the problem at hand. If it satisfies the condition of the lemma, the lemma is trivially true. Otherwise, we show that it is possible to build an optimal allocation θ^* from θ, respecting the condition.

Let k^* the maximum index of any empty slot of θ. If it is the last slot, take any non-allocated ad, and allocate it to k^*. Such ad must exist because we always assume $N \geq K$. As the social welfare of the new allocation is not less than that of θ, we conclude that the new allocation is optimal.

If, on the other hand, $k^* \neq K$, we act as follows: we move all the ads allocated in slots $k^* + 1$ to K one slot above. This operation does not decrease the social welfare of the allocation, and at the same time leaves slot K empty. We then operate as in the previous case, filling slot K with any non-allocated ad.

We repeat this algorithm until no slot is vacant. □

The problem of finding a maximal K-vertex weighted simple path in $\mathcal{G}$ can be solved by means of the Color Coding technique (Alon et al., 1994). The basic idea is as follows: in a single iteration, vertices are assigned one of K random colors drawn with uniform probability; then, the best path visiting every color exactly once is found using a dynamic programming approach; finally, this loop is repeated a certain amount of times, depending on the type of algorithm (randomized or derandomized). We call the resulting algorithm COLORED-ADS, and show the pseudocode in Algorithm 3.

Algorithm 3 COLORED-ADS

```
1: procedure COLORED-ADS(ads, slots)
2:     repeat e^K log 2 times                          ▷ 50% success probability
3:         Assign random colors ∈ {1,...,K} to G's vertices
           (note: each color must be used at least once)
       ▷ We now construct a memoization table MEMO[C], reporting, for each color subset C of
         {1,...,K} the maximum value of any simple path visiting colors in C exactly once.
4:         MEMO[∅] ← 0
5:         P ← powerset of {1,...,K}, sorted by set size
6:         for C ∈ P − {∅}, in order do
7:             λ̃ ← λ_{K−|C|+1}
8:             for each color c ∈ C do
9:                 for each vertex (ad) a in G of color c do
10:                    VALUE ← v̄_a + c_a λ̃ · MEMO[C − {c}]
11:                    MEMO[C] ← max{MEMO[C], VALUE}
12:    return MEMO[{1,...,K}]
```

We state the following lemma, whose proof directly follows from the properties of the Color Coding algorithm (Alon et al., 1994).

Lemma 29. *The* colored-ads *algorithm finds the optimal solution with probability* $1 - e^{-\frac{\text{restarts}}{e^K}}$*, where* restarts *is the number of random restarts, with a runtime complexity of* $O(2^K N)$ *per restart and an overall memory complexity of* $O(2^K + N)$.

It can be easily observed that an exponential number of restarts is necessary to assure a strictly positive success probability. Furthermore, notice that in the derandomized version, the algorithm is exact and requires $O((2e)^K K^{O(\log K)} N \log N)$ time when paired with the derandomization technique presented by Naor, Schulman, and Srinivasan (1995). In terms of worst-case approximation, the colored-ads algorithm cannot provide a guarantee better than $\frac{1}{K}$ as stated in the following lemma.

Lemma 30. *The* colored-ads *algorithm, when executed in the randomized version, provides a guarantee of* $\frac{1}{K}$ *and this is tight.*

Proof. It can be easily observed that it always possible to find a $\frac{1}{K}$-approximation by placing the ad with the largest value $\overline{v}$ in the first slot. As well, it can be easily observed that in the worst case for all the random restarts all the ads are assigned with the same color and therefore the allocation retuned by the algorithm is composed by a single ad. □

Note that Algorithm 3 is just a simplified, randomized, with a success probability of 50%, and non-parallelized sketch of the algorithm we test in the experimental section of this paper; also, for the sake of presentation, in Algorithm 3 we only return the social welfare of the optimal allocation and not the allocation itself. We investigated also the idea of using $1.3K$ different colors, as suggested by Hüffner, Wernicke, and Zichner (2008), but we found out that for this particular problem the improvement is negligible.

We conclude this subsection with some remarks about the above algorithm. First, we remark that, given the nature of the operations involved (mostly arithmetic operations and accesses to one array), the algorithm proves to be efficient in practice, with only a small constant hidden in the big-oh notation. Furthermore, it is worth noting that the iterations of the main loop (Lines 2 to 15) are independent; as such, the algorithm scales well horizontally in a parallel computing environment. Second, we point out an important economic property of the algorithm, that makes colored-ads appealing: it allows the design of truthful mechanisms when paired with VCG-like payments.

Lemma 31. *When* colored-ads *is paired with VCG-like payments, it leads to a DSIC mechanism in the following versions:*

- *derandomized;*
- *randomized in which the number of random restarts is decided before the reporting phase of the agents;*
- *anytime randomized in which the time allocated for each random restart is the worst-case one (i.e., the time required by the longest execution).*

Proof. In all the three cases above, colored-ads is maximal-in-its-range. More precisely, when the algorithm is derandomized, the proof is trivial, since the algorithm always returns the optimal allocation. When the number of random restarts is fixed *a priori*, the algorithm

Algorithm 4 SORTED-ADS

procedure SORTED-ADS(ads, slots, $\prec_{\text{ads}}$)
  Sort the ads according to $\prec_{\text{ads}}$, so that ad 1 is the minimum ad w.r.t. the given order

  ▷ We now construct a memoization table $\mathcal{T}[n,k]$, reporting, for each $1 \leq n \leq N$ and $1 \leq k \leq K$, the social welfare of the best allocation that uses only ads $n,\dots,N$ and slots $k,\dots,K$ (i.e. no ads get allocated to slots $1,\dots,k-1$ and $\lambda_i = 1, \forall i < k$).

  $\mathcal{T}[N,k] \leftarrow \bar{v}_N, \quad k = 1,\dots,K$ ▷ Base case
  for $n = N-1$ **downto** 1 **do**
    for $k = 1,\dots,K$ **do**
      if $k < K$ **then**
        VALUE $\leftarrow \bar{v}_n + \lambda_k\, c_n\, \mathcal{T}[n+1,k+1]$
        $\mathcal{T}[n,k] \leftarrow \max\{\text{VALUE}, \mathcal{T}[n+1,k]\}$
      else
        $\mathcal{T}[n,k] \leftarrow \max\{\bar{v}_n, \mathcal{T}[n+1,K]\}$
  return $\mathcal{T}[1,1]$

is maximal-in-its-range, as the range is decided independently of the agents' reports and the algorithm returns the best allocation among all those available given the range. Finally, when executed in anytime fashion, in order to preserve the property that the range does not depend on agents' reports, it is sufficient to force that the time spent to solve each single restart is the same and does not depend on the agents' reports. This can be obtained by allocating the worst-case execution time to each random restart and, if the execution time of some random restart terminates prematurely, the algorithm must wait for the remaining time. □

6. SORTED-ADS Algorithm

While the problem of finding an optimal allocation is difficult when no restriction is posed, polynomial time algorithms are easy to derive—as we show below—when we restrict the set of feasible allocations to those respecting a given total order $\prec_{\text{ads}}$ defined on the ads set. This suggests the following simple approximation algorithm: first, T total orders $\prec_{\text{ads},1}$ $,\dots,\prec_{\text{ads},T}$ over the set of ads are chosen; then, the optimal allocation satisfying $\prec_{\text{ads},i}$ is computed, for each i; finally, the social welfare of the optimal allocation for the original unrestricted problem is approximated with the best allocation found over all the T orders.

In order to find the optimal allocation respecting the total order $\prec_{\text{ads}}$, we propose a simple $O(NK)$ time dynamic programming algorithm, which we name SORTED-ADS, described in Algorithm 4. The idea behind the algorithm is to find, for each $n = 1,\dots,N$ and $k = 1,\dots,K$, the value $\mathcal{T}[n,k]$ of the social welfare of the best allocation for the problem having $\mathcal{A}_n = \{1,\dots,n\} \subseteq \mathcal{A}$ as ads set and $\mathcal{K}_k = \{k,\dots,K\} \subseteq \mathcal{K}$ as slots set. The values of $\mathcal{T}[n,k]$ can be computed inductively, noticing that the associated subproblems share the same optimal substructure. As before, in Algorithm 4 we only show how to find the social welfare of the optimal allocation, and not the allocation itself. Also, for the sake of simplicity, the algorithm we present uses $O(NK)$ memory, but we remark that it is possible to bring

the required memory down to $O(N + K) = O(N)$ while letting fast (i.e. $O(NK)$ time) reconstruction of the optimal allocation, using a technique similar to that one presented by Hirschberg (1975).

6.1 Memory-Efficient Algorithm

We modify the idea by Hirschberg (1975) to find a $O(NK)$ time, $O(N)$ memory implementation of SORTED-ADS. In particular, we need to show that there exists a $O(NK)$ time and $O(N)$ memory algorithm able to find the least slot k^*, respecting Definition 9.

Definition 9. *Slot k is a* split-point *if there exists an optimal allocation such that slots $1, \ldots, k-1$ are occupied only by ads a with $a \leq N/2$ and slots $k, \ldots, K$ are occupied only by ads a with $a > N/2$.*

Conceptually, we scan all the possible values of k, one by one from 1 up until K until we find the first split-point. Once a tentative value for k^* has been chosen, it is enough to test whether the best allocation satisfying the condition in Definition 9 is an optimal allocation for the original problem. The split-point k breaks the optimal allocation fulfilling the condition into two natural pieces: the first using only ads in $\{1, \ldots, N/2\}$ and slots $1, \ldots, k-1$, and the second piece using ads in $\{N/2+1, \ldots, N\}$ and slots $k, \ldots, K$. We let the social welfare of the first piece be W_1, and that of the second piece be W_2; also, we let C_1 be the product of the continuation probabilities of the ads in the first piece. The value SW of the optimal allocation satisfying the split-point condition is therefore

$$\mathrm{SW} = W_1 + \Lambda_{k-1} C_1 W_2.$$

Since by definition the allocation is optimal, the second piece of the allocation must be itself an optimal allocation for the problem having $\mathcal{A} = \{N/2+1, \ldots, N\}$, $\mathcal{K} = \{k, \ldots, K\}$. Notice that we already know how to find W_2, for all the possible values of k, in $O(NK)$ time and $O(N)$ memory. We can therefore treat W_2 as a known constant and focus on the problem of determining the allocation of the first piece maximizing SW. The crucial point is that we can find the maximum of W_1 for each k in dynamic programming. Indeed, the objective function is the same we would have in the allocation problem in which we have to allocate ads $\{1, \ldots, N/2\} \cup \{X\}$ in slots $1, \ldots, k$ under the constraints:

- $\bar{v}_X = W_2$ (notice that W_2 is known, while the values c_X and λ_k can be disregarded not affecting the objective function);
- ad X must be allocated in slot k.

6.2 Economic Properties

We note that, if the total orders used do not depend on the reported types of the agents, SORTED-ADS is maximal-in-its-range, leading thus to a truthful mechanism when paired with VCG-like payments. This is true also when the algorithm is executed on a subset T' of the total orders T set to satisfy a given time-limit constraint (the choice of the subset must be independent of the agents' reports). Furthermore, the resulting mechanism is computationally tractable, requiring polynomial time both for finding the allocation and the

payments. Indeed, the allocation can be found by means of Algorithm 4, that, as discussed above, requires polynomial compute time, and the payment of agent i can be found by calling Algorithm 4 to the subproblem in which agent i is removed from the market. We can state the following lemma whose proof is similar to the proof of Lemma 31.

Lemma 32. *The* SORTED-ADS *algorithm, when paired with VCG-like payments, leads to a DSIC mechanism in the following versions:*

- *the set of orders $\prec_{ad}$ is fixed before the agents report their types;*
- *the set of orders $\prec_{ad}$ is generated in anytime fashion independently of agents' reports and the time allocated for the execution time of the algorithm for each order must be the same equal to the worst-case time.*

6.3 Approximation Guarantees — Constant λ

It is possible to prove a 1/2-approximation ratio for the SORTED-ADS algorithm in specific settings, e.g. when all the λ are constant. The proof in the general case, though, is left here as an open conjecture.

Definition 10. *Given a total order $\prec_{ads}$, we define $\Theta_{\prec_{ads}}$ as the set of allocations θ where:*

- *for every $a, a' \in \theta$, $\text{slot}_\theta(a) < \text{slot}_\theta(a')$ only if $a \prec_{ads} a'$ (notice that the constraint is only over the allocated ads, each non-allocated ad may appear everywhere in $\prec_{ads}$);*
- *there are no empty slots before an allocated slot.*

Given a $\prec_{\text{ads}}$, the set $\Theta_{\prec_{\text{ads}}}$ constitutes the range of our allocation algorithm.

Definition 11. *We define $\theta^*_{\prec_{ads}}$ as the optimal allocation (maximizing the social welfare) among all the allocations in $\Theta_{\prec_{ads}}$.*

Definition 12. *We define θ^* as the optimal allocation (maximizing the social welfare) among all the allocations in Θ and $\prec^*_{ads}$ as $\prec^*_{ads} = \{\prec_{ads} : \theta^*_{\prec_{ads}} = \theta^*\}$.*

Definition 13. *We define $\prec^N_{ads}$ as: for every $a, a' \in A$, $a \prec^N_{ads} a'$ if and only if $\frac{\overline{v}_a}{1-c_a} \geq \frac{\overline{v}_{a'}}{1-c_{a'}}$ (ties are broken lexicographically).*

When $\lambda_k = 1$ for every k, we have $\prec^N_{\text{ads}} \in \prec^*_{\text{ads}}$ for every value of the parameters as shown by Kempe and Mahdian (2008).

Definition 14. *We define $\prec^R_{ads}$ as the reverse of $\prec^N_{ads}$.*

We introduce a lemma that we use in our main result.

Lemma 33. *When $\lambda_k = 1$ for every k, once A is restricted to the ads appearing in θ^* and the ads are labeled such that a is the a-th ad in the allocation, then for every a we have:*

$$\frac{\overline{v}_a}{1-c_a} \geq \sum_{b=a}^{K} \overline{v}_b \prod_{h=i}^{b-1} c_h.$$

Proof. The proof is by mathematical induction.

Basis of the induction. The basis of the induction is for slots k and $k-1$. We need to prove that:

$$\frac{\overline{v}_{k-1}}{1-c_{k-1}} \geq \overline{v}_{k-1} + c_{k-1}\overline{v}_k. \tag{4}$$

Equation 4 can be rewritten as:

$$\frac{\overline{v}_{k-1}}{1-c_{k-1}} \geq \overline{v}_k,$$

that holds since

$$\frac{\overline{v}_{k-1}}{1-c_{k-1}} \geq \frac{\overline{v}_k}{1-c_k}$$

(by definition of $\prec^N_{\text{ads}}$) and

$$\frac{\overline{v}_k}{1-c_k} \geq \overline{v}_k$$

(by $c_k \in [0,1]$).

Inductive step. We assume that following inequality holds

$$\frac{\overline{v}_{i+1}}{1-c_{i+1}} \geq \sum_{j=i+1}^{k} \overline{v}_j \prod_{h=i+1}^{j-1} c_h,$$

and we want to prove

$$\frac{\overline{v}_{a_i}}{1-c_{a_i}} \geq \sum_{j=i}^{k} \overline{v}_{a_j} \prod_{h=i}^{j-1} c_{a_h}. \tag{5}$$

Equation 5 can be written as:

$$\frac{\overline{v}_i}{1-c_i} \geq \sum_{j=i+1}^{k} \overline{v}_j \prod_{h=i+1}^{j-1} c_h,$$

that holds since

$$\frac{\overline{v}_i}{1-c_i} \geq \frac{\overline{v}_{i+1}}{1-c_{i+1}}$$

(by definition of $\prec^N_{\text{ads}}$) and

$$\frac{\overline{v}_{i+1}}{1-c_{i+1}} \geq \sum_{j=i+1}^{k} \overline{v}_j \prod_{h=i+1}^{j-1} c_h$$

(by assumption of the inductive step). This completes the proof. □

We use the above lemma to derive the bound.

Theorem 1. *When $\lambda_k = 1$ for every k, we have*

$$r = \frac{\mathrm{SW}(\theta^*_{\prec_{\mathrm{ads}}})}{\mathrm{SW}(\theta^*)} \geq \frac{1}{2}$$

for every $\prec_{\mathrm{ads}}$.

Proof. For the sake of presentation, we split the proof in several parts.

Ads ranking. Among all the possible $\prec_{\mathrm{ads}}$, we can safely focus only on $\prec^R_{\mathrm{ads}}$ given that this order is the worst for our problem. This property follows from the results discussed by Kempe and Mahdian (2008). More precisely, the authors show that, given any allocation θ, for every a, a' if $\mathrm{slot}_\theta(a) < \mathrm{slot}_\theta(a')$ and $\frac{\overline{v}_a}{1-c_a} > \frac{\overline{v}_{a'}}{1-c_{a'}}$ then switching a with a' never reduces the social welfare of the allocation. Therefore, the social welfare with $\prec^R_{\mathrm{ads}}$ is the minimum w.r.t. that one with all the $\prec_{\mathrm{ads}}$. (Interestingly, when $\prec_{\mathrm{ads}} \notin \prec^*_{\mathrm{ads}}$, $\theta^*_{\prec_{\mathrm{ads}}}$ may prescribe that the number of allocated ads is strictly smaller than the number of ads allocated in θ^*.)

Allocations set. Given $\prec^R_{\mathrm{ads}}$, we use a lower bound over the value $\theta^*_{\prec^R_{\mathrm{ads}}}$ focusing only on a specific subset of allocations defined below. This subset may not contain the optimal allocation $\theta^*_{\prec^R_{\mathrm{ads}}}$ and therefore we obtain a lower bound for r that may be not tight. However, in the following, we complete the result showing that the bound is tight. The allocations we focus on are of the form: $\langle z, z-1, \ldots, 1\rangle$ with $z \in \{1, \ldots, K\}$ where the ads are indexed on the basis on $\prec^N_{\mathrm{ads}}$ (i.e., $1 \prec^N_{\mathrm{ads}} 2 \prec^N_{\mathrm{ads}} \ldots$). We study the value of r for each possible value of z under the constraints that, given z, it holds:

$$\sum_{i=1}^{z} \left(\overline{v}_i \prod_{j=i+1}^{z} c_j \right) \geq \sum_{i=1}^{z+1} \left(\overline{v}_i \prod_{j=i+1}^{z+1} c_j \right) \tag{6}$$

$$\sum_{i=1}^{z} \left(\overline{v}_i \prod_{j=i+1}^{z} c_j \right) \geq \sum_{i=1}^{h} \left(\overline{v}_i \prod_{j=i+1}^{h} c_j \right) \qquad \forall h : 1 \leq h < z \tag{7}$$

with the exception that when $z = K$, only Constraints (7) are required. We notice that, for every auction instance, there always exists a value of z such that the above constraints hold. This value can be found by starting from $z = 1$ and increasing it until the constraints are satisfied.

Ratio. Once z is fixed, a lower bound over r can be obtained by using the following upper bound over the value of θ^*: we compute θ^* as if $K = N$. Thus, the ratio r is:

$$r \geq \frac{\sum_{i=1}^{z} \left(\overline{v}_i \prod_{j=i+1}^{z} c_j \right)}{\sum_{i=1}^{N} \left(\overline{v}_i \prod_{j=1}^{i-1} c_j \right)}.$$

Bound with $z = 1$. In this case, we have:

$$z = 1$$
$$\overline{v}_1 \geq \overline{v}_2 + c_2\overline{v}_1 \tag{8}$$
$$r \geq \frac{\overline{v}_1}{\sum_{i=1}^{N}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right)} \tag{9}$$

by applying Lemma 33 and Equation 8, we can write r as:

$$r \geq \frac{\overline{v}_1}{\sum_{i=1}^{N}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right)} \geq \frac{\overline{v}_1}{\overline{v}_1 + c_1\frac{\overline{v}_2}{1-c_2}} \geq \frac{\overline{v}_1}{\overline{v}_1 + c_1\overline{v}_1}.$$

The minimization of $\frac{\overline{v}_1}{\overline{v}_1+c_1\overline{v}_1}$ can be achieved by maximizing c_1 (i.e., when $c_1 = 1$), obtaining $r \geq \frac{1}{2}$.

Bound with $K > z > 1$. In this case, we can write r as:

$$r \geq \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{N}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right)} = \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \sum_{i=z+1}^{N}\left(\overline{v}_i \prod_{j=z+1}^{i-1} c_j\right)},$$

by applying Lemma 33, we have:

$$r \geq \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \sum_{i=z+1}^{N}\left(\overline{v}_i \prod_{j=z+1}^{i-1} c_j\right)} \geq \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \frac{\overline{v}_{z+1}}{1-c_{z+1}}},$$

by using Equation 6 we have:

$$r \geq \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \frac{\overline{v}_{z+1}}{1-c_{z+1}}} \geq \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}.$$

We derive r w.r.t. $\overline{v}_z$ and we show that the derivative is always positive:

$$\frac{\partial r}{\partial \overline{v}_z} = \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) - \prod_{j=1}^{z-1} c_j \sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)}{\left(\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)\right)^2}$$
$$= \frac{\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j \left(1 - \prod_{j=i}^{z-1} c_j \prod_{j=i+1}^{z} c_j\right)\right)}{\left(\sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z} c_j \sum_{i=1}^{z}\left(\overline{v}_i \prod_{j=i+1}^{z} c_j\right)\right)^2} \geq 0,$$

given that $\left(1 - \prod_{j=i}^{z-1} c_j \prod_{j=i+1}^{z} c_j\right) \geq 0$. Thus, we minimize r by minimizing $\overline{v}_z$. Due to Constraint (7) with $h = z-1$, the minimum feasible value of $\overline{v}_z$ is:

$$\overline{v}_z = (1 - c_z)\left(\sum_{i=1}^{z-1}\left(\overline{v}_i \prod_{j=i+1}^{z-1} c_j\right)\right),$$

substituting $\overline{v}_z$ in r we have:

$$r \geq \frac{\sum_{i=1}^{z-1}\left(\overline{v}_i \prod_{j=i+1}^{z-1} c_j\right)}{\sum_{i=1}^{z-1}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right) + \prod_{j=1}^{z-1} c_j \sum_{i=1}^{z-1}\left(\overline{v}_i \prod_{j=i+1}^{z-1} c_j\right)}.$$

By applying iteratively the above steps for $\overline{v}_h$ with h from $h = z-2$ to $h = 2$, we obtain the same bound of the case with $z = 1$:

$$r \geq \frac{\overline{v}_1}{\overline{v}_1 + c_1 \overline{v}_1} \geq \frac{1}{2}.$$

Bound with $z = K$. Differently from the previous cases, here we do not use an upper bound over θ^*, but the exact value, and for the lower bound over $\theta^*_{\prec^R_{\text{ads}}}$ we consider only the ads appearing in θ^*. In this case, we have:

$$r \geq \frac{\sum_{i=1}^{K}\left(\overline{v}_i \prod_{j=i+1}^{K} c_j\right)}{\sum_{i=1}^{K}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right)},$$

we derive r w.r.t. $\overline{v}_K$, obtaining:

$$\frac{\partial r}{\partial \overline{v}_K} = \frac{\sum_{i=1}^{K}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j \left(1 - \prod_{j=i}^{K-1} c_j \prod_{j=i+1}^{K} c_j\right)\right)}{\left(\sum_{i=1}^{K}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j\right)\right)^2} \geq 0,$$

we minimize r by minimizing $\overline{v}_K$ as:

$$\overline{v}_K = (1 - c_K)\left(\sum_{i=1}^{K-1}\left(\overline{v}_i \prod_{j=i+1}^{K-1} c_j\right)\right),$$

obtaining:

$$r \geq \frac{\sum_{i=1}^{K-1}\left(\overline{v}_i \prod_{j=i+1}^{K-1} c_j\right)}{\sum_{i=1}^{K-1}\left(\overline{v}_i \prod_{j=1}^{i-1} c_j \left(1 + (1 - c_K)c_i \prod_{j=i+1}^{K-1} c_j^2\right)\right)}.$$

By applying iteratively the above steps for $\overline{v}_h$ with h from $h = K-2$ to $h = 2$—the derivative is always positive—, we obtain:

$$r \geq \frac{\overline{v}_1}{\overline{v}_1 + \overline{v}_1 c_1\left(1 - \prod_{i=2}^{K} c_j\right)} = \frac{1}{1 + c_1\left(1 - \prod_{i=2}^{K} c_j\right)},$$

r is minimized when $c_1 = 1$ and $\prod_{i=2}^{K} c_j = 0$, obtaining $r \geq \frac{1}{2}$. □

The extension of the above result to the case in which $\lambda_k = \lambda$ for every k is straightforward. Indeed, it is sufficient to observe that in this case we can redefine the continuation probabilities as $c'_i = c_i\,\lambda$ for every i and subsequently redefine $\lambda_k = 1$ for every k. Finally, the proof of the theorem above provides an example that proves the following lemma.

Lemma 34. *When the orders $\prec_{\text{ad}}$ are chosen randomly, the approximation bound of $\frac{1}{2}$ is tight.*

6.4 Approximation Guarantees — Small K

The extension of the proof of Theorem 1 to the general case in which λ_k are arbitrary in $(0, 1)$ and K can be any is not straightforward. More precisely, the proof is trivial when $K = 2$. When $K = 3$, the proof can be obtained by enumerating a large number of conditions. The crucial issue is that these conditions rise exponentially in K, and it seems no condition can be safely discarded. This makes the derivation of a proof when K can be any difficult, requiring one to enumerate an intractable number of conditions. Nevertheless, we believe the following conjecture to hold in general:

Definition 15. *Given a total order $\prec_{ads}$, let $\Psi_{\prec_{ads}}$ be the set of all $\psi > 0$ such that, for any valid allocation $\theta \in \Theta$, there exists $\theta_{\prec_{ads}} \in \Theta_{\prec_{ads}}$ with the property that* $\mathrm{val}(\theta_{\prec_{ads}}) \geq \psi\, \mathrm{val}(\theta)$.

Conjecture 1. *$\frac{1}{2} \in \Psi_{\prec_{ads}}$ for all $\prec_{ads}$.*

Notice that Conjecture 1 is equivalent to the fact that the SORTED-ADS algorithm always returns an $\frac{1}{2}$-approximation of the social welfare of the optimal allocation for the problem at hand for any total order. We provide some arguments supporting our conjecture. By using global optimization tools, able to return the optimal solution even of non-convex non-linear optimization problems within a given finite accuracy (set, in our case, to 10^{-4}), it is possible to verify the conjecture above for a large number of significant cases. More precisely, we consider the following feasibility problem:

$$\mathcal{P}_{\prec_{\text{ads}}}(K,\pi,\psi): \begin{cases} \text{Variables:} \\ \qquad \bar{v}_1,\dots,\bar{v}_K \text{ (ad parameters)} \\ \qquad c_1,\dots,c_K \text{ (ad parameters)} \\ \qquad \lambda_1,\dots,\lambda_{K-1} \text{ (slot parameters)} \\ \\ \text{Constraints:} \\ \qquad \textcircled{1}\ \text{Permutation } \pi \text{ of } \{1,\dots,N\} \text{ coincides with } \prec_{\text{ads}} \\ \qquad \textcircled{2}\ \mathrm{val}(\theta^*) \geq \mathrm{val}(\theta), \quad \forall \theta \in \Theta \\ \qquad \textcircled{3}\ \psi\, \mathrm{val}(\theta^*) > \mathrm{val}(\theta), \quad \forall \theta \in \Theta_{\prec_{\text{ads}}} \end{cases} \tag{10}$$

where, without loss of generality:

- we set $N = K$;
- θ^* can be taken as the identity allocation, i.e. the allocation mapping ad i in slot i for each $i = 1,\dots,K$.

We state the following lemma.

Lemma 35. *If $\mathcal{P}_{\prec_{ads}}(K,\pi,\psi)$ is infeasible for all permutations π of $\{1,\dots,K\}$, then $\psi \in \Psi_{\prec_{ads}}$ for the given number of available slots K.*

Proof. By contradiction, suppose there exists a problem instance where the best allocation θ^* has a value $\mathrm{val}(\theta^*) > (1/\psi)\mathrm{val}(\theta^*_{\prec_{\text{ads}}})$, where $\theta^*_{\prec_{\text{ads}}}$ is the best allocation respecting the given order $\prec_{\text{ads}}$. Let $\tilde{\pi}$ be the permutation induced by $\prec_{\text{ads}}$. Since $\mathcal{P}_{\prec_{\text{ads}}}(K,\pi,\psi)$ is infeasible for all π, it is infeasible for $\tilde{\pi}$ as well. In other words, it is impossible to assign values to the variables in such a way that constraints ①, ② and ③ all hold at the same time. This means that for any possible assignment of values that satisfies ① and ②, there always exists θ satisfying $\tilde{\pi}$ such that $\mathrm{val}(\theta) \geq \psi\, \mathrm{val}(\theta^*)$. Because θ respects π, it is $\theta \in \Theta_{\prec_{\text{ads}}}$, and hence $\mathrm{val}(\theta^*_{\prec_{\text{ads}}}) \geq \mathrm{val}(\theta) \geq \psi\, \mathrm{val}(\theta^*)$. This contradicts our hypothesis, showing that $\psi \in \Psi_{\prec_{\text{ads}}}$. □

Remark 2. *In order to prove that $\psi \in \Psi_{\prec_{ads}}$ for all $\prec_{ads}$, one can drop constraint ① in (10).*

Remark 3. *Our method requires the solution of $K!$ non-linear optimization problems, each having $\approx K! + 2^K$ constraints and $3K - 1$ variables. As such, it becomes impractical even for very modest numbers, such as $K = 7$.*

Our experimental verification of the conjecture above shows the following results:[7]

- when $1 \leq K \leq 4$, we observe $\frac{1}{2.1} \in \Psi_{\prec_{\text{ads}}}$ for all $\prec_{\text{ads}}$, showing that, for every given total order, SORTED-ADS returns a solution with an approximation ratio of at least $\frac{1}{2.1}$; the need for using $\frac{1}{2.1}$ in place of $\frac{1}{2}$ is due to accuracy issues of the global optimization solver: when $\frac{1}{2}$ is used, the solver requires an extremely long time; by the way, no case with an approximation ratio $< \frac{1}{2}$ has been found;
- when $K = 5$, we observe $\frac{1}{2.1} \in \Psi_{\prec^c_{\text{ads}}}$, where $\prec^c_{\text{ads}}$ represents any total order of ads respecting $a \prec^c_{\text{ads}} b \implies c_a \geq c_b$; this shows that for such total order SORTED-ADS returns a solution with an approximation ratio of at least $\frac{1}{2.1}$ (by the way, no case with an approximation ratio $< \frac{1}{2}$ has been found); our experimental analysis is limited only to total order $\prec^c_{\text{ads}}$ due to computational issues: global optimization tools are very fast with $\prec^c_{\text{ads}}$ thanks to the presence of additional constraints that speedup the resolution process, while they require an extremely long time with the other total orders, not allowing us to provide a complete verification of the claim with these total orders in a reasonable time (by the way, even with total orders different from $\prec^c_{\text{ads}}$ no case with approximation ratio $< \frac{1}{2}$ has been found);
- when $K = 6$, we observe $\frac{1}{3} \in \Psi_{\prec^c_{\text{ads}}}$, showing that, for such total order, SORTED-ADS returns a solution with an approximation ratio of at least $\frac{1}{3}$ (by the way, no case with an approximation ratio $< \frac{1}{2}$ has been found); the need for using $\frac{1}{3}$ in place of $\frac{1}{2}$ is due to computational issues, for the same reason we limit the analysis to the total order $\prec^c_{\text{ads}}$.
- when $K \geq 7$, global optimization tools require excessively long computation times and therefore they cannot be used unless exploiting a massive parallelization.

7. Experimental Evaluation

For a better comparison of the results, we adopt the same experimental setting used by Gatti and Rocco (2013). The experimental setting is based on *Yahoo! Webscope A3* dataset. Each bid is drawn from a truncated Gaussian distribution, where the mean and standard deviation are taken from the dataset, while quality is drawn from a beta distribution. The prominence values Λ_s for the first 10 slots are $\{1.0, 0.71, 0.56, 0.53, 0.49, 0.47, 0.44, 0.44, 0.43, 0.43\}$ and they were estimated from the dataset by means of least squares method. Instead, the values of c_a cannot be estimated from the dataset, since the dataset does not provide any information on the displayed allocations, but only information on the single ads. In practice, this does not allow one to estimate how an ad can influence the CTR of the ads displayed below. Thus, we use synthetic values of c sampling uniformly from $[0, 1]$. Let us notice that Craswell et al. (2008) propose to adopt $c = 1 - q$. However, in this case, every c has a value very close to 1 and thus the APDC model is very similar to the PDC model.

7. In our experimental analysis we used the BARON solver (Sahinidis, 2014).

Empirically, the approximation ratios obtained with our experimental setting are worse than those obtained with the setting proposed by Craswell et al.. Furthermore, we consider two scenarios, one having $K = 5$ and one having $K = 10$. In both cases we let $N \in \{50, 60, \ldots, 100\} \cup \{200, 300, \ldots, 1000\}$. For each pair (K, N), 20 instances were generated. The implementation of our algorithms is in the C++11 language. We ran the algorithms on the OSX 10.10.3 operating system. The main memory is a 16GB 1600MHz DDR3 RAM, while the processor is an Intel Core i7-4850HQ CPU. The source is compiled by GNU `g++` version 4.9.1. Parallelization is achieved by using OpenMP version 4.0.

7.1 DOMINATED-ADS Algorithm

We study the average number of ads that survive DOMINATED-ADS in Figure 1. In addition to the average, the figure also shows the boxplot of the 20 repetitions for each (N, K) pair.[8] The upper-bounding strategy needed in Lemma 17 is implemented using a $O(NK)$ time algorithm.

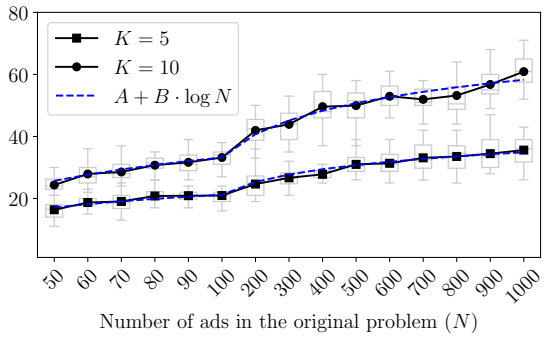

Figure 1: Average number of ads after running the DOMINATED-ADS algorithm.

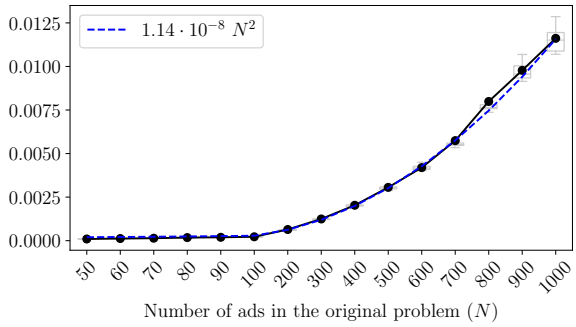

Figure 2: Average DOMINATED-ADS running time (seconds). This is independent of K and grows as $\approx 1.2 \cdot 10^{-8} N^2$ ($R^2 > 0.99$).

Notice that the number of removed ads is already considerable when N is small (e.g., $N \approx 50$), and that the prune ratio increases dramatically as N grows (for instance, the prune ratio is approximately 96% on average when $N = 1{,}000$ and $K = 5$). Experiments show that the number of surviving ads is of the form $\tilde{N} = A + B \cdot \log N$ for suitable values of A and B. In Figure 3, the boxplots of the number of ads surviving DOMINATED-ADS and of the prune ratio are reported.[9,10] They show the high statistical significance of the results, all the boxplots presenting a very strict range, outliers included.

8. A boxplot is composed of a box and whiskers. The box specifies the first quartile (Q1), the median, and the third quartile (Q3). The distance between Q1 and Q3 is called the interquartile range (IQR). The upper whisker extends to last datum less than Q3 + 1.5 IQR. Similarly, the lower whisker extends to the first datum greater than Q1 − 1.5 IQR. Beyond the whiskers, data are considered outliers and are plotted as individual points.
9. For $K = 5$, we have $A \approx -6.1, B \approx 5.9$ and $R^2 > 0.98$, where R^2 is the coefficient of determination of the fit.
10. For $K = 10$, we have $A \approx -16.9, B \approx 10.9$ and $R^2 > 0.98$.

In Figure 2 we report the average and the boxplot of the running time for DOMINATED-ADS. The graph shows that the running time depends quadratically on the original problem size N, but it remains negligible (shorter than 20 milliseconds) even when $N = 1{,}000$. These results are also confirmed by the boxplots reported in Figure 3, where the ranges of all the boxplots are very strict.

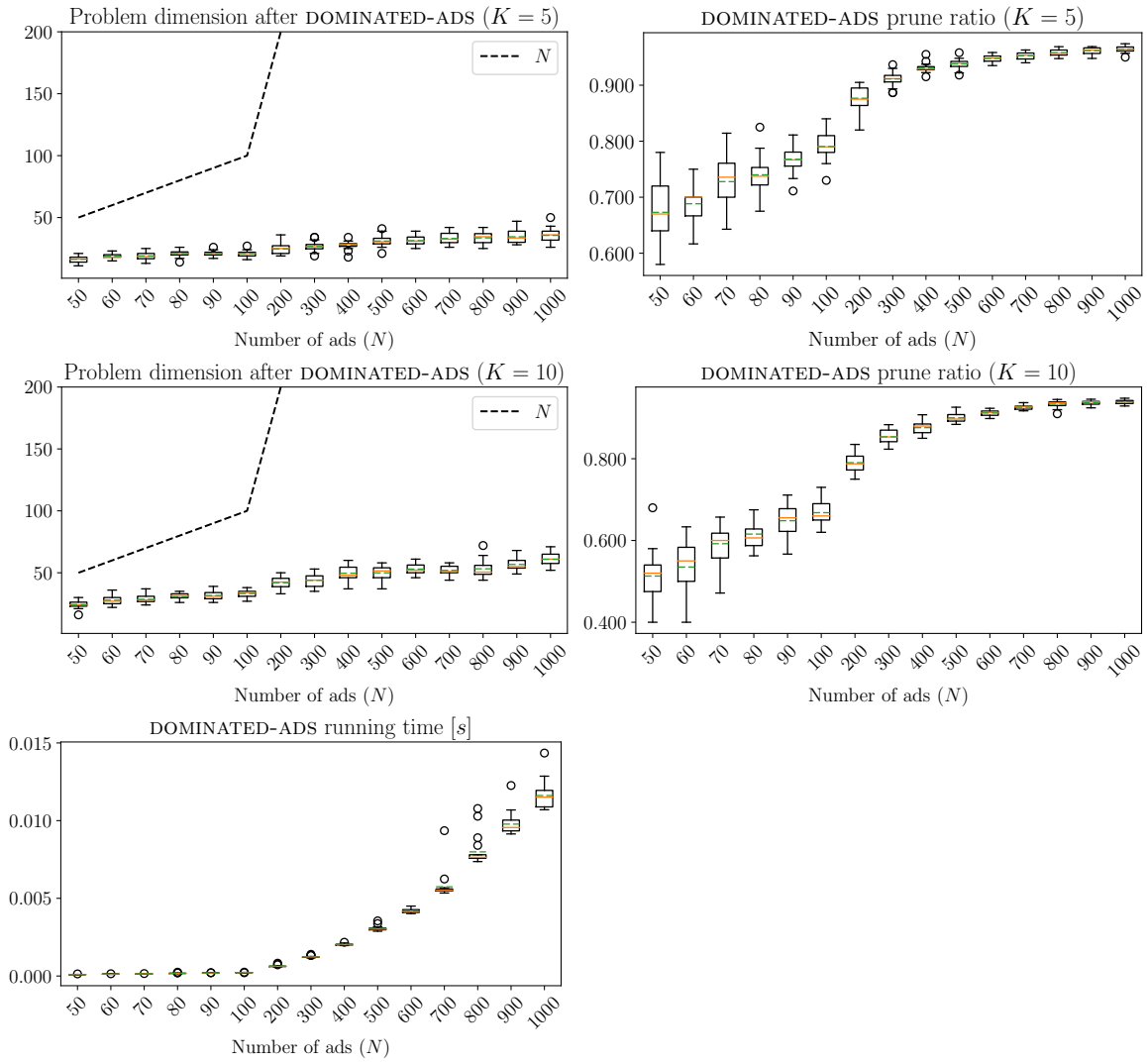

Figure 3: Boxplots of the experimental results of DOMINATED-ADS. The average value is drawn as a dashed green line, while the median as an orange solid line.

7.2 COLORED-ADS Algorithm

Figure 4 shows the average and the boxplot of the running time for COLORED-ADS, implemented in its randomized version. The number of iterations is set to $e^K \log 2$, assuring to have a probability of at least 50% of finding the optimal allocation. The algorithm is run on

the reduced instances produced by DOMINATED-ADS. As a result of the shrinking process, we point out that the running times for $N = 50$ and $N = 1{,}000$ are comparable. We also remark that, in order for COLORED-ADS to be applicable in real-time contexts when $K = 10$, some sort of hardware scaling is necessary, as a running time of ≈ 0.5 seconds per instance is likely to be too expensive for many practical applications. When $K = 5$, though, no scaling is necessary, and the algorithm proves to be extremely efficient.

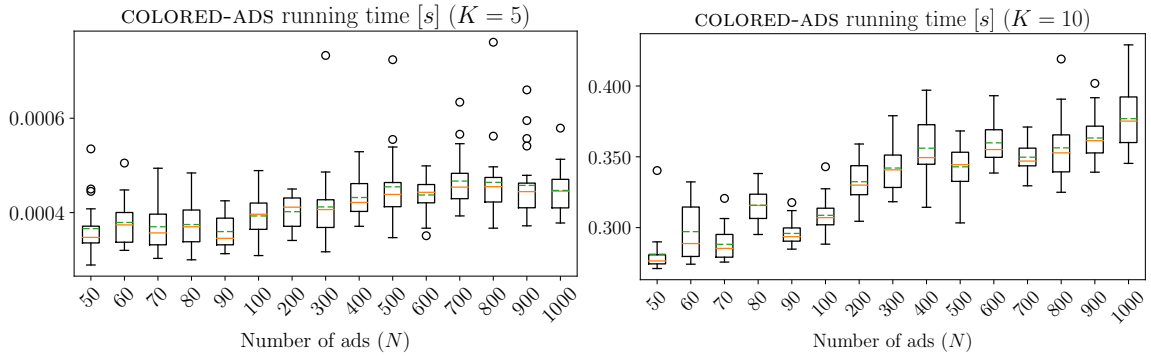

Figure 4: Average COLORED-ADS running time (seconds, $e^K \log 2$ runs).

Furthermore, we experimentally investigate the approximation ratio of COLORED-ADS as the number of random restarts varies. The social welfare of the optimal allocations is computed by using an exponential branch-and-bound algorithm, similar to that of Gatti and Rocco (2013). Experimental results are reported in Figure 5. It can be observed that with $K = 10$ even the minimum approximation ratio achieved is larger than 0.98, requiring a running time that is negligible for a number of recolorings smaller than 2,000. When the number of random restarts is exponential (consider that $e^K \log 2$ is larger than $10,000$) the minimum approximation ratio is larger than 0.999.This shows that COLORED-ADS can be successfully applied in real-wold settings with a small number of recolorings.

7.3 SORTED-ADS Algorithm

In Figure 6 we study the approximation ratio of SORTED-ADS. For each problem instance shrunk by DOMINATED-ADS, we generate $T = 2K^3$ random orders, and use SORTED-ADS to approximate the optimal allocation. Surprisingly, even though the number of iterations is relatively low, approximation ratios demonstrate to be very high, with values $> 97\%$ in the worst case, with both medians and means always $> 99\%$. In Figure 7 we report the corresponding running times. These are in the order of a handful of milliseconds (one order of magnitude smaller than the running time of COLORED-ADS when run for a sub-exponential number of random restarts), thus making SORTED-ADS suitable in real-time contexts even for a large number of ads.

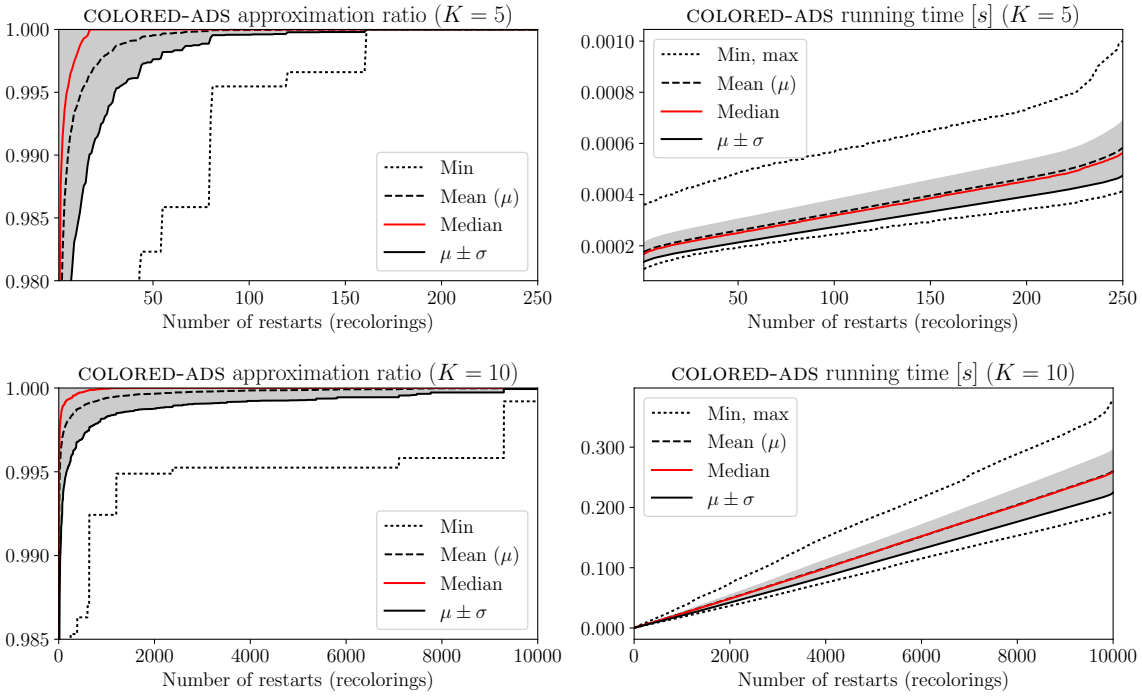

Figure 5: Approximation ratio and running time of COLORED-ADS when run for a sub-exponential number of iterations. The standard deviation σ of the outcomes is computed for each fixed number of restarts.

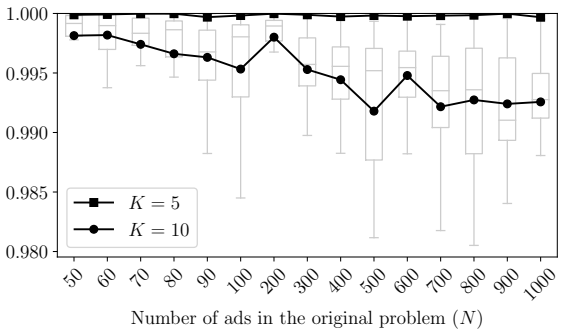

Figure 6: Average SORTED-ADS approximation ratio ($T = 2K^3$ orders).

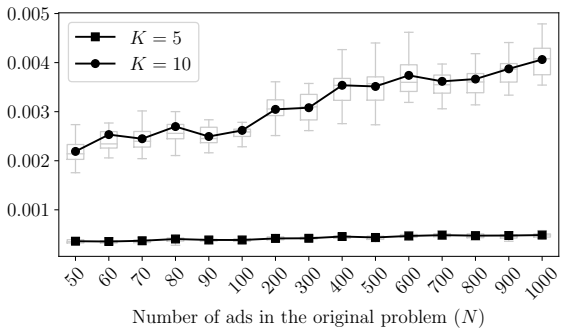

Figure 7: Average SORTED-ADS running time (seconds, $T = 2K^3$ orders).

In Figure 8, we report the boxplots related to the above experimental results. They show that the range is very strict, and therefore the standard deviation is extremely small. It can be observed that the running time required by the execution of DOMINATED-ADS together with SORTED-ADS in the worst case for $N = 1,000$ is less than 20 milliseconds, with an approximation ratio in the worst case larger than 0.98. Summarily, SORTED-ADS provides

an empirical approximation ratio slightly smaller than that of COLORED-ADS, but it requires less running time.

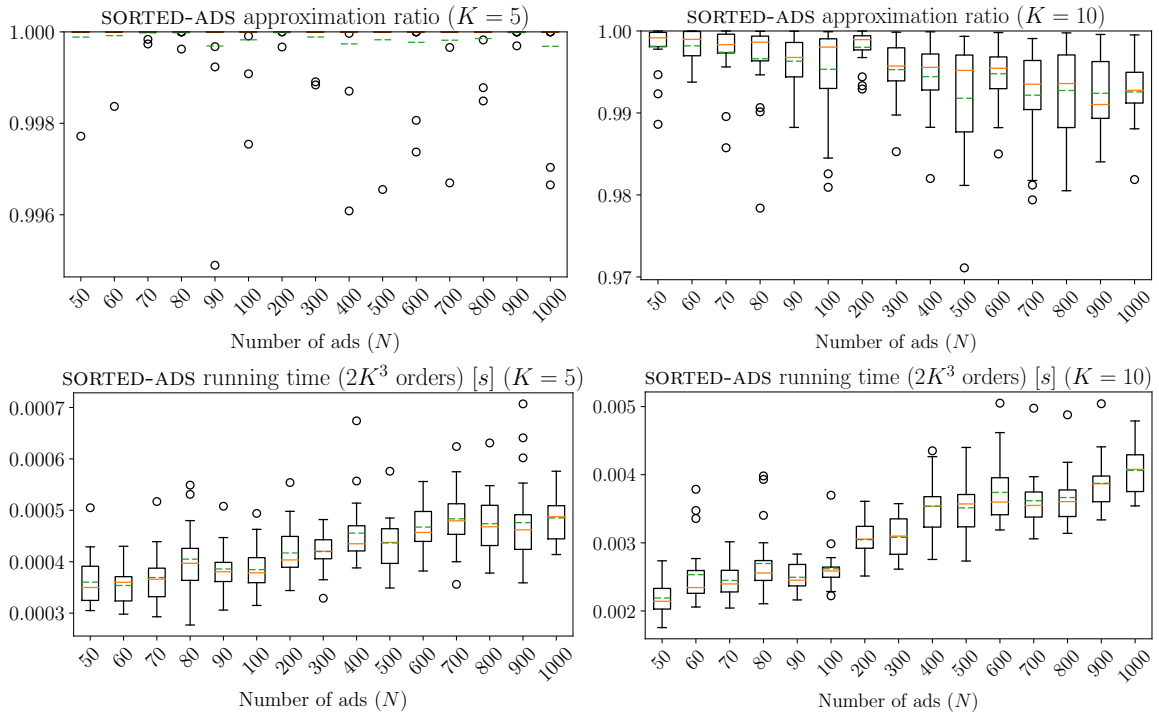

Figure 8: Boxplots of the results of SORTED-ADS.

7.4 Discussion

We compare here the proposed algorithms in terms of theoretical and empirical properties. A brief summary is provided in Table 2.

Approximation guarantees. COLORED-ADS does not provide any theoretical guarantee over the worst-case approximation better than $\frac{1}{K}$. When COLORED-ADS is executed with an exponential number of restarts, the probability to find the optimal solution is strictly larger than 0, while when the number of restarts is sub-exponential the probability goes to zero in K (however, it is not negligible with $K = 10$ and $2,000$ restarts, being about 9%). When derandomized, COLORED-ADS is the most efficient known algorithm finding the optimum allocation within the APDC model. Instead, SORTED-ADS provides some guarantees in terms of approximation in the worst case, while the probability of finding the optimal solution is negligible. The empirical approximation of the algorithms with $K = 10, N = 1,000$ is > 0.999 on average for COLORED-ADS with exponential restarts, 0.997 on average for COLORED-ADS with polynomial restarts (precisely, $2,000$ restarts) and 0.990 on average for SORTED-ADS (precisely, $2,000$ restarts).

Running time. In COLORED-ADS, the asymptotic running time needed for each restart is exponential in K (in terms of both time and memory), while in SORTED-ADS is polynomial in K (all the algorithms are polynomial in N). The actual running times of the three algorithms with $K = 10, N = 1,000$ are: about 0.5 seconds to have 50% success probability with

colored-ads, about 0.05 seconds with a polynomial number of restarts with colored-ads (with $2,000$ restarts), about 0.005 seconds with sorted-ads (with $2,000$ restarts).

Truthfulness. All the algorithms are truthful when executed with a given number of restarts and even when executed in an anytime fashion.

Summary. The choice of the algorithm to adopt strictly depends on the available technology. Our recommendation is that, with the current technology, the sorted-ads algorithm should be adopted. It provides theoretical guarantees that make the performance of the algorithm robust even in non-average cases, its average performance is close to that of colored-ads, and its running time is extremely low. Considering the huge number of simultaneous queries a search engine has to manage, we believe that requiring a very low running time is crucial. However, we envision that in the future the growth in available computing power will make the computational cost of the derandomized colored-ads algorithm affordable. In this case, this latter approach guarantees that the best allocation is always found.

Algorithm	**colored-ads** exp(K) restarts	**colored-ads** poly(K) restarts	**sorted-ads** poly(K) restarts
Theoretical approximation guarantee	$\frac{1}{K}$	$\frac{1}{K}$	$\frac{1}{2}$ in special cases (numerical) $\frac{1}{2}$ up to $K=5$ (numerical) $\frac{1}{3}$ up to $K=6$ (conjectured) $\frac{1}{2}$
Empirical approximation with $K=10, N=1,000$	>0.999 on average	0.997 on average (2,000 restarts)	0.990 on average (2,000 restarts)
Theoretical bound on optimal solution probability	$1-e^{-\frac{\text{restarts}}{e^K}}$ (50% with $e^K \log 2$ restarts)	$1-e^{-\frac{\text{restarts}}{e^K}}$ (9% with $K=10$ and 2,000 restarts)	$1-\frac{(N-K)!}{N!}$ ($\sim 0\%$ with $K=10$, $N=1{,}000$ and 2,000 restarts)
Asymptotic running time	restarts $\cdot\, 2^K$	restarts $\cdot\, 2^K$	restarts $\cdot$ poly(K,N)
Actual running time with $K=10$, $N=1{,}000$	about 0.5 seconds ($e^K \log 2$ restarts)	about 0.05 seconds (2,000 restarts)	about 0.005 seconds (2,000 restarts)
Anytime truthfulness	yes	yes	yes

Table 2: A comparison of the theoretical and empirical properties of the proposed algorithms.

8. Conclusions and Future Work

In this paper, we provide several contributions about the ad/position dependent cascade model for ad auctions introduced by Kempe and Mahdian (2008). This user model is more accurate than the separable model that is currently used. However, it is not know

whether there is an efficient algorithm for the winner determination problem and no (even exponential) algorithm suitable for real-world applications is known. Initially, we study the inefficiency of the GSP auction—currently adopted for sponsored search auctions—and the PDC-VCG auction—currently adopted for contextual advertising—with respect to the APDC-VCG auction. We analyze upper bounds over the PoA and PoS both of the social welfare and the auctioneer's revenue. Our results show that the inefficiency is in most cases unbounded and that the dispersion over the equilibria is extremely large, suggesting that the GSP auction, as well as the PDC-VCG auction, presents severe limits. Subsequently, we provide three algorithms and we experimentally evaluate them with a real-world dataset. Our first algorithm reduces the size of the instances, discarding ads that surely will not be allocated in the optimal allocation. Empirically, the number of non-discarded ads is logarithmic. Our second algorithm finds the optimal allocation with high probability, requiring a short time (less than 1 millisecond when $K = 5$ even with 1,000 ads), but too long for real-time applications when $K = 10$. Our third algorithm approximates the optimal allocation in very short time (less than 5 milliseconds even with 1,000 ads) providing very good approximations on average (>0.99). This shows that the ad/dependent cascade model can be effectively used in practice. Furthermore, we prove that our approximation algorithm has a tight theoretical bound of 1/2 in restricted domains and we conjecture that the bound holds in general settings.

In the future, we will extend our analysis and our algorithms to models in which there is a contextual graph among the ads, and in which there are multiple slates. Furthermore, we will explore novel advertising scenarios, such as, e.g., mobile advertising.

Acknowledgments

A preliminary version of this paper appeared at AAAI (Farina & Gatti, 2016). The present paper extends the preliminary version by including all missing proofs, providing additional inefficiency results, reporting a theoretical analysis of the approximation guarantees of the SORTED-ADS algorithm, and presenting more computationally efficient variants of the algorithms.

Appendix A. Missing Proofs

Lemma 1. *Both in the restricted and in the unrestricted case, the $\overline{PoA}$ of the auctioneer's revenue in the GSP when users behave as prescribed by the APDC model is unbounded.*

Proof. Consider a problem with $N > K \geq 2$ having:

- $v_1 = 1$, $v_2 = \ldots = v_N = 1 - \epsilon$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = 0$, $c_2 = \ldots = c_N = 1$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$,
- $\hat{v}_1 = 1$, $\hat{v}_2 = \ldots = \hat{v}_N = 0$.

The allocation mapping ad i to slot i for each $i = 1, \ldots, K$ is optimal for the GSP. The revenue of the auctioneer is 0, each agent being charged a payment of 0. Furthermore, the above strategy profile is a Nash equilibrium, as the first agent obviously has no incentive in decreasing her reported type, while each other agent can be allocated in the first slot only

bidding a value larger than or equal to 1, leading to a payment of $1 > 1-\epsilon$, hence generating a strictly negative utility. The APDC-VCG auction admits an optimal allocation featuring ad 1 in slot K. In this case, agent 1 is charged a payment $1-\epsilon$, leading the auctioneer to have a strictly positive revenue. This concludes the proof. □

Lemma 2. *In the restricted case, the $\overline{PoS}$ of the social welfare in the GSP auction when users behave as prescribed by the APDC model is $\geq K/2$.*

Proof. Consider a problem with $N = K \geq 3$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = 0, c_2 = \ldots = c_N = 1$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$.

We need to characterize all Nash equilibria of the GSP auction when the correct externality model is that provided by the APDC model. We assume w.l.o.g. a tie-breaking mechanism breaking ties in favor of the auctioneer with the lowest index. We split the analysis into cases:

- $\hat{v}_1$ is the maximum among $\{\hat{v}_1, \ldots, \hat{v}_N\}$. By the tie-breaking rule, this means that ad 1 is assigned to slot 1. Now, if an equilibrium exists, $\hat{v}_{\mathrm{ad}_\theta(2)} \leq v_1 = 1$ (otherwise, agent 1 would be better off lowering their bid to the point where they get assigned to the last slot). But this means that, if a Nash equilibrium for this case exists, the revenue of the system is $\leq \hat{v}_{\mathrm{ad}_\theta(2)} \leq 1$, since $c_1 = 0$. In other words, whatever the equilibrium, the PoS, in this case, is not smaller than K.

- $\hat{v}_1$ is (strictly) the second maximum among the bids, i.e. there exists exactly one bid (w.l.o.g. that of agent 2) such that $\hat{v}_2 > \hat{v}_1$. Iterating the reasoning above, we conclude that $\hat{v}_1 \leq 1$ and $\hat{v}_{\mathrm{ad}_\theta(3)} \leq 1$. Therefore, in this case, the equilibrium, if it exists, has a PoS on the social welfare not smaller than $K/2$. Unlike the previous case, we prove that such equilibrium exists. Indeed, consider the case where $\hat{v}_2 = 2$ and all other bids are 1. All agents have a utility equal to 0, and it is easy to verify that nobody has an incentive to deviate from the profile.

- $\hat{v}_1$ is (strictly) neither the first nor the second maximum among the bids, i.e. there exist at least two distinct bids (w.l.o.g., e.g. that of agent 2 and agent 3 respectively) such that $\hat{v}_2 \geq \hat{v}_3 > \hat{v}_1$. The utility of agent 2 is $u_2 \leq 1 - \hat{v}_3$. However, by bidding a value close enough (yet greater than) $\hat{v}_1$, agent 2 would increase their utility by at least $\hat{v}_3 - \hat{v}_1 > 0$. This means that there cannot be any Nash-compatible bidding strategy in which agent 1 is assigned to a slot lower than the second.

The analysis above shows that the $\overline{\mathrm{PoS}}$ on the social welfare is $\geq K/2$. □

Lemma 3. *In the restricted case, the $\overline{PoS}$ of the auctioneer's revenue in the GSP auction when users behave as prescribed by the APDC model is $\geq K/2$.*

Proof. The thesis follows from the proof of Lemma 2. □

Lemma 4. *Both in the restricted and in the unrestricted case, the $\underline{PoS}$ of the social welfare in the GSP when users behave as prescribed by the APDC model is 1.*

Proof. Trivially, by definition of PoS over the social welfare, PoS is always larger than or equal to 1. In order to show that PoS may be equal to 1, it is enough to show that in at least one circumstance the allocation of GSP and APDC-VCG auction coincide. To this end, consider a problem with $N > K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = \ldots = c_N = 1$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$,
- $\hat{v}_1 = \ldots = \hat{v}_N = 1$.

It is easy to see that the allocation mapping ad i to slot i for each $i = 1, \ldots, K$ is optimal with both the mechanisms. Furthermore, it is a Nash equilibrium in the GSP, since each agent whose ad is displayed evaluates the allocation 1, pays 1 to the mechanism, and thus has a utility of 0, while all the other agents have a utility of 0. It is straightforward to see that no agent benefits from misreporting their true type. This concludes the proof. □

Lemma 5. *Both in the restricted and in the unrestricted case, the PoS of the auctioneer's revenue in the GSP when users behave as prescribed by the APDC model is 0.*

Proof. Consider a problem with $N = K = 2$ having:

- $v_1 = 1$, $v_2 = 1/3$,
- $q_1 = q_2 = 1$,
- $c_1 = 1, c_2 = 1/2$,
- $\lambda_1 = 1$,
- $\hat{v}_1 = 1$, $\hat{v}_2 = 1/3$.

The GSP allocates ad 1 before ad 2, with a revenue of 1/3. Notice that the above strategy profile constitutes a Nash equilibrium, as agent 1 can only decrease her utility from 2/3 to 1/2 when ad 1 is allocated below ad 2, while agent 2 can only decrease her utility from 1/3 to $-2/3$ if ad 2 is allocated in the first slot. While the APDC-VCG auction allocates ad 1 before ad 2 as well, the revenue for the system is 0, since agent 2 is not charged any payment (not being subject to competition for the second slot), and agent 1 is charged $v_2 - \lambda_1 c_1 v_2 = 0$. This concludes the proof. □

Lemma 6. *In the unrestricted case, the PDC-VCG auction may not admit any (even mixed-strategy) Nash equilibrium when users behave as prescribed by the APDC model.*

Proof. Consider a problem with $N = K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = \ldots = c_N = 1/2$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$.

It can be observed that each agent is charged a payment of 0 independently of the reported values $\hat{v}_i$. This is due to the fact that $N = K$ and $\lambda_s = 1$ for every s. However, each agent aims at having their ad displayed in the first position. Thus, given any profile $\hat{v}_{-i}$, agent i will bid $\epsilon + \max_{j \neq i}\{\hat{v}_j\}$. This shows that the agents keep increasing their bids and therefore no equilibrium exists. Notice that the non-existence of a mixed-strategy equilibrium is due to the fact that the game is not finite, but the set of actions is $[0, +\infty)$. □

Lemma 8. *In the unrestricted case, when a Nash equilibrium exists, the $\overline{PoA}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Proof. Consider a problem with $N = K = 2$ having:

- $v_1 = 0, v_2 = 1$,
- $q_1 = q_2 = 1$,
- $c_1 = 0$, $c_2 = 1$,
- $\lambda_1 = 1/2$,
- $\hat{v}_1 = 4, \hat{v}_2 = 0$.

The utilities for the agents are both 0 in the PDC-VCG auction. The PDC-VCG auction allocates ad 1 before ad 2, as $q_1\hat{v}_1 + \lambda_1 q_2\hat{v}_2 > q_2\hat{v}_2 + \lambda_1 q_1\hat{v}_1$. The above strategy profile is a Nash equilibrium, as agent 1 does not gain more by reducing her bid, while, if agent 2 made any bid > 4, her utility would be $\bar{v}_2 - (1 - \lambda_1)q_1\hat{v}_1 < 0$. The social welfare produced by the PDC-VCG auction is 0. On the other hand, the APDC-VCG auction allocates ad 2 before ad 1, producing a strictly positive social welfare. This concludes the proof. □

Lemma 9. *In the restricted case, the $\overline{PoA}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is $\geq K$.*

Proof. Consider a problem with $N > K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = 0$, $c_2 = \ldots = c_N = 1$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$,
- $\hat{v}_1 = 1, \hat{v}_2 = \ldots = \hat{v}_N = 0$.

The allocation mapping ad i to slot i for each $i = 1, \ldots, K$ is optimal for the PDC-VCG auction, resulting in a revenue of 1. Under the assumption that ties are broken lexicographically, this strategy profile is also a Nash equilibrium. Indeed, agent 1 has a utility of 1 and cannot improve it; agents i with $i \geq 2$ have a utility of 0 since their value from the allocation is zero and their payments are zero. They cannot improve their utility. Every optimal allocation for APDC-VCG auction allocates ad 1 either in slot K or it does not display such an ad, resulting in a payment of K. This concludes the proof. □

Lemma 10. *In the unrestricted case, when a Nash equilibrium exists, the $\overline{PoA}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Proof. Consider a problem with $N = K \geq 2$ having:

- $v_1 = 1, v_2 = \ldots = v_N = 1/2 - \epsilon$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = 0, c_2 = \ldots = c_N = 1$,
- $\lambda_1 = 1/2, \lambda_2 = \ldots = \lambda_{K-1} = 1$,
- $\hat{v}_1 = 1, \hat{v}_2 = \ldots = \hat{v}_N = 0$.

The optimal allocation for the PDC-VCG auction is to display ad i in slot i. The payments are 0 for every agent. The above strategy profile is a Nash equilibrium: agent 1 cannot gain more than in the current strategy profile, while the other agents would have a strictly

negative utility if they overbid to be displayed in the first position since they would pay 1/2 to the mechanism. As a result, the mechanism has a revenue of 0. In the APDC-VCG auction the revenue is strictly positive. This concludes the proof. □

Lemma 11. *In the restricted case, the $\overline{PoA}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Proof. Consider a problem with $N = K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = \ldots = c_N = 1/2$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$,
- $\hat{v}_1 = \ldots = \hat{v}_N = 1$.

Any allocation is optimal for the PDC-VCG auction, charging each agent a payment of 0 since $N = K$. Furthermore, the above strategy profile is a Nash equilibrium for the PDC-VCG auction in the restricted case in which no agent overbids. Thus, the revenue of the auctioneer with PDC-VCG auction is 0. On the other hand, in the APDC-VCG auction, all the agents except the one allocated in slot K are charged a strictly positive payment. This concludes the proof. □

Lemma 12. *In the restricted case, the $\overline{PoS}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is $\geq K$.*

Proof. Consider a problem with $N = K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = 0, c_2 = \ldots = c_N = 1 - \epsilon$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$.

We need to characterize all Nash equilibria of the PDC-VCG auction when the correct externality model is that provided by the APDC model. The existence of at least an equilibrium is guaranteed by Lemma 7. We assume w.l.o.g. a tie-breaking mechanism breaking ties in favor of the auctioneer with the lowest index. Since $N = K$ and $\lambda = 1$ for all slots, all payments are 0. Notice that ad 1 will always be allocated in slot 1, for otherwise agent 1 could bid $\hat{v}_1 = v_1 = 1$ and secure the first slot for themselves, strictly increasing their utility. But since $c_1 = 0$, all equilibria of the PDC-VCG auction mechanism produce a social welfare of 1, while the APDC-VCG auction would produce a social welfare of $1 + (1-\epsilon) + \cdots + (1-\epsilon)^{K-1} \approx K$. Hence, the PoS for this scenario is K, showing that the $\overline{\text{PoS}}$ of the mechanism is $\geq K$. □

Lemma 13. *In the restricted case, the $\overline{PoS}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is unbounded.*

Proof. Consider a problem with $N = K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = \ldots = c_N = 1/2$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1$.

We need to characterize all Nash equilibria of the PDC-VCG auction when the correct externality model is that provided by the APDC model. The existence of at least an equilibrium is guaranteed by Lemma 7. We assume w.l.o.g. a tie-breaking mechanism breaking ties in favor of the auctioneer with the lowest index. Since $N = K$ and $\lambda_i = 1$ for all slots i, all payments are 0. This implies that whatever the equilibrium considered, the revenue for the system is 0. On the other hand, the APDC-VCG auction auction would have a strictly positive revenue, as the continuation probabilities are not all 1. All equilibria of this instance have an unbounded inefficiency, showing that the $\overline{\text{PoS}}$ of the revenue in the PDC-VCG auction may be unbounded. □

Lemma 14. *Both in the restricted and unrestricted case, the $\underline{\text{PoS}}$ of the social welfare in the PDC-VCG auction when users behave as prescribed by the APDC model is 1.*

Proof. Follows by considering the same setting as in the proof of Lemma 4. □

Lemma 15. *In the unrestricted case, when a Nash equilibrium exists, the $\underline{\text{PoS}}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC is 0.*

Proof. Consider a problem with $N = K = 2$ having:

- $v_1 = 1, v_2 = 0$,
- $q_1 = q_2 = 1$,
- $c_1 = 1$, $c_2 = 1/2$,
- $\lambda_1 = 1/2$,
- $\hat{v}_1 = 1, \hat{v}_2 = 1/2$.

Both PDC-VCG auction and APDC-VCG auction allocate ad 1 before ad 2. With PDC-VCG auction, agent 1 has a utility of 3/4, while agent 2 has a utility of 0. Notice that the above strategy profile is a Nash equilibrium of the PDC-VCG auction: if agent 1 made a bid $< 1/2$, her utility would be 1/4; if agent 2 made a bid > 1, her utility would be $-1/2$. The revenue for PDC-VCG auction is 1/4, while that of APDC-VCG auction is 0. This concludes the proof. □

Lemma 16. *In the restricted case, the $\underline{\text{PoS}}$ of the auctioneer's revenue in the PDC-VCG auction when users behave as prescribed by the APDC model is ≤ 1.*

Proof. Consider a problem with $N > K \geq 2$ having:

- $v_1 = \ldots = v_N = 1$,
- $q_1 = \ldots = q_N = 1$,
- $c_1 = \ldots = c_N = 1$,
- $\lambda_1 = \ldots = \lambda_{K-1} = 1/2$,
- $\hat{v}_1 = \ldots = \hat{v}_N = 1$.

Notice that the strategy is a Nash equilibrium of the PDC-VCG auction, as no agent can overbid, and no agent benefits from lowering her reported type. Furthermore, since all the ad continuation probabilities are equal to 1, the revenues of the two systems are equal, thus proving the lemma. This concludes the proof. □

References

Aggarwal, G., Feldman, J., Muthukrishnan, S., & Pál, M. (2008). Sponsored search auctions with Markovian users. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pp. 621–628.

Alon, N., Yuster, R., & Zwick, U. (1994). Color-coding: A new method for finding simple paths, cycles and other small subgraphs within large graphs. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pp. 326–335.

Bachrach, Y., Ceppi, S., Kash, I. A., Key, P., & Kurokawa, D. (2014). Optimising trade-offs among stakeholders in ad auctions. In *Proceedings of the International ACM Conference on Economics and Computation (EC)*, pp. 75–92.

Bentley, J. L. (1979). Decomposable searching problems. *Information Processing Letters*, *8*(5), 244 – 251.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, *19*(90), 297–301.

Craswell, N., Zoeter, O., Taylor, M., & Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 87–94.

Edelman, B., Ostrovsky, M., & Schwarz, M. (2007). Internet advertising and the Generalized Second-Price auction: Selling billions of dollars worth of keywords. *American Economic Review*, *97*(1), 242–259.

Evangelos, E., & Telelis, O. (2010). Discrete strategies in keyword auctions and their inefficiency for locally aware bidders. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pp. 523–530.

Farina, G., & Gatti, N. (2016). Ad auctions and cascade model: GSP inefficiency and algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 489–495.

Fotakis, D., Krysta, P., & Telelis, O. (2011). Externalities among advertisers in sponsored search. In *Proceedings of the International Symposium on Algorithmic Game Theory (SAGT)*, pp. 105–116.

Gatti, N., Lazaric, A., Rocco, M., & Trovò, F. (2015). Truthful learning mechanisms for multi-slot sponsored search auctions with externalities. *Artificial Intelligence*, *227*, 93–139.

Gatti, N., & Rocco, M. (2013). Which mechanism in sponsored search auctions with externalities?. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 635–642.

Gatti, N., Rocco, M., Serafino, P., & Ventre, C. (2015). Cascade model with contextual externalities and bounded user memory for sponsored search auctions. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 1831–1832.

Giotis, I., & Karlin, A. R. (2008). On the equilibria and efficiency of the GSP mechanism in keyword auctions with externalities. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pp. 629–638.

Goldman, M., & Rao, J. M. (2014). Experiments as instruments: Heterogeneous position effects in sponsored search auctions. Working papers, SSRN. Available online at `http://dx.doi.org/10.2139/ssrn.2524688`.

Gomes, R., Immorlica, N., & Markakis, E. (2009). Externalities in keyword auctions: An empirical and theoretical assessment. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pp. 172–183.

Gunawardana, A., & Meek, C. (2008). Aggregators and contextual effects in search ad markets. In *Proceedings of the Workshop on Targeting and Ranking for Online Advertising at the International World Wide Web Conference (WWW)*.

Hardy, G., Littlewood, J., & Polya, G. (1952). *Inequalities*. Cambridge Mathematical Library.

Hegeman, J. (2010). *Facebook's ad auction. Talk at Ad Auctions Workshop*.

Hirschberg, D. S. (1975). A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, *18*(6), 341–343.

Hüffner, F., Wernicke, S., & Zichner, T. (2008). Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica*, *52*(2), 114–132.

IAB (2017). *IAB Internet Advertising Revenue Report. 2016 Full year results*.

Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., & Gay, G. (2007). Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems*, *25*(2), 7.

Kempe, D., & Mahdian, M. (2008). A cascade model for externalities in sponsored search. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pp. 585–596.

Kuminov, D., & Tennenholtz, M. (2009). User modeling in position auctions: re-considering the GSP and VCG mechanisms. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 273–280.

Leme, R. P., & Tardos, E. (2010). Pure and Bayes-Nash price of anarchy for Generalized Second Price auction. In *Proceedings of Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 735–744.

Lucier, B., Leme, R. P., & Tardos, E. (2012). On revenue in the generalized second price auction. In *Proceedings of International World Wide Web Conference (WWW)*, pp. 361–370.

McLaughlin, K., & Friedman, D. (2016). Online ad auctions: An experiment. Working papers 16-05, Chapman University, Economic Science Institute.

Mehlhorn, K., & Näher, S. (1990). Dynamic fractional cascading. *Algorithmica*, *5*(1-4), 215–241.

Musser, D. R. (1997). Introspective sorting and selection algorithms. *Software: Practice and Experience*, *27*(8), 983–993.

Naor, M., Schulman, L. J., & Srinivasan, A. (1995). Splitters and near-optimal derandomization. In *Proceedings of Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 182–191.

Narahari, Y., Garg, D., Narayanam, R., & Prakash, H. (2009). *Game Theoretic Problems in Network Economics and Mechanism Design Solutions*. Springer.

Nisan, N., & Ronen, A. (2000). Computationally feasible VCG mechanisms. In *Proceedings of the International ACM Conference on Electronic Commerce (EC)*, pp. 242–252.

Sahinidis, N. V. (2014). *BARON 14.3.1: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual.

Varian, H. R., & Harris, C. (2014). The VCG auction in theory and practice. *American Economic Review*, *104*(5), 442–445.