ECONOMIC MECHANISMS FOR ONLINE PAY–PER–CLICK ADVERTISING: COMPLEXITY, ALGORITHMS AND LEARNING

Doctoral Dissertation of:

Marco Rocco

Supervisor:
Prof. Nicola Gatti

Tutor:
Prof. Cesare Alippi

The Chair of the Doctoral Program:
Prof. Carlo Ettore Fiorini

2014 – XXVII
Auctions for online pay–per–click advertising (e.g., Sponsored Search Auctions) constitute one of the most successful applications of microeconomic mechanisms, producing a revenue of about 42.8 billions in the U.S. alone in 2014. A crucial issue is the study of accurate models of the user attention and their effective exploitation in the economic mechanisms. A famous example widely studied in the literature is the Cascade Model, in which a user is assumed to scan the ads in a Markovian way. Such a model exhibits several limitations, not being suitable for a large number of scenarios, e.g., in mobile advertising scenarios. In this dissertation, we provide a hierarchy of user models including as special cases the models currently adopted in literature and we theoretically analyze our models along four perspectives that are all necessary for the success of an economic mechanism: computational complexity of finding the best allocation, exact and theoretically bounded approximation algorithms, online learning algorithms with theoretical bounds over the regret, and incentive compatibility. The ideal goal is the design of a user models admitting the best results in each the four perspectives. In the dissertation, we show that this is not the case and that, when we select a user model, it is necessary to handle a trade–off among the four perspectives.
## Contents

1 Introduction ........................................... 1

2 Preliminaries ........................................... 7
   2.1 Mechanism Design .................................. 7
       2.1.1 Notions of truthfulness ....................... 8
       2.1.2 Additional desirable properties ............. 10
       2.1.3 Known implementability results ............ 10
   2.2 Multi–Armed Bandit ................................ 14
   2.3 Sponsored Search Auction models ............... 15
       2.3.1 SSA and Mechanism Design ................ 18
       2.3.2 SSA and MAB ................................ 23

3 Models and Summary of the Results .................. 27
   3.1 Models ........................................... 27
       3.1.1 Three new models: CFNE_{aa}, CFNE_{sa}, CFNE_{qi,m} ........ 30
       3.1.2 Two new dimensions: user memory and a⊥ .... 31
       3.1.3 The last new model: Mobile Advertising .... 32
   3.2 Summary of Results ................................ 35
       3.2.1 Computational Complexity and Mechanism Design ........ 35
       3.2.2 SSA and MAB ................................ 37

4 CFNE_{sa}(c) and CFNE_{aa}(c) with \( c < K \) .......... 39
   4.1 CFNE_{sa}(c) is in \( P \) for constant \( c \) .......... 39
   4.2 CFNE_{aa}(1)\( \{nr,r\} \) is APX–hard .............. 44
   4.3 Approximation algorithms for CFNE_{aa}(c) ....... 48
<table>
<thead>
<tr>
<th>Section Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9 Learning: Numerical Simulations</td>
<td>137</td>
</tr>
<tr>
<td>9.1 Learning with Separable CTR</td>
<td>138</td>
</tr>
<tr>
<td>9.1.1 Unknown $\Lambda_m, m \in \mathcal{K}$</td>
<td>138</td>
</tr>
<tr>
<td>9.1.2 Unknown $\Lambda_m, m \in \mathcal{K}$, and $q_i, i \in \mathcal{N}$</td>
<td>139</td>
</tr>
<tr>
<td>9.2 Learning with Cascade Model</td>
<td>140</td>
</tr>
<tr>
<td>10 Conclusions and Future Works</td>
<td>143</td>
</tr>
<tr>
<td>Bibliography</td>
<td>151</td>
</tr>
</tbody>
</table>
CHAPTER 1

Introduction

Sponsored search auctions (SSAs) constitute one of the most successful applications of microeconomic mechanisms, producing a revenue of about $42.8 billions in the U.S. alone in 2014, dominating display ads, the second-largest revenue source [31]. In a SSA, a number of advertisers bid to have their sponsored links (from here on ads) displayed in some slot alongside the search results of a keyword. Sponsored search auctions currently adopt a pay–per–click scheme, which requires positive payments from an advertiser only when its ad is clicked. Given an allocation of ads over the available slots, each ad is associated with a click–through rate (CTR) corresponding to the probability that such ad will be clicked by the user. CTRs are estimated by the auctioneer and play a crucial role in the definition of the auction, since they are used by the auctioneer to compute the optimal allocation (in expectation) and to compute the payments for each ad.

In microeconomic literature, SSAs have been formalized as a mechanism design problem [43], where the objective is to design an auction mechanism that incentivizes advertisers to bid their truthful valuations (needed for economic stability) and that assures both the advertisers and the auctioneer to have a non–negative utility. The most common SSA mechanism is the generalized second price (GSP) auction [21][53]. As shown in [21], this
mechanism is not truthful and advertisers may implement bidding strategies that gain more than bidding their truthful valuations. While in complete information settings the worst Nash equilibrium in the GSP gives a revenue to the auctioneer equal to the revenue given by the Vickrey–Clarke–Groves (VCG) equilibrium [21], in Bayesian settings the worst Bayes–Nash equilibrium in the GSP can provide a much smaller revenue than the VCG—a lower bound of \(\frac{1}{8}\) is provided in [37]. The implementation of the VCG mechanism (assuring truthfulness) for SSAs has been investigated in [43]. Although the VCG mechanism is not currently adopted by the search engines (but it is used, e.g., by Facebook), a large number of scientific theoretical results build upon it.

A crucial issue in SSAs is the study of effective models of the user attention and their exploitation in the auction mechanism. Recently, a number of works showed that externalities play an important role in the user behavior [2, 22, 35]. On the other hand, externalities may make the problem of finding the optimal allocation intractable, even when approximated, e.g., it is APX–hard in [22]. The most widely adopted user model is the Cascade Model [2, 35], in which a user is assumed to scan the ads sequentially from the top slot to the bottom slot with a probability to observe the subsequent slot that depends on the last observed ad (ad–dependent externality) and on its position (position–dependent externality) and with the remaining probability the user stops to observe the ads. Although several experimental activities [19, 33] confirmed that the behavior of the users is described better when modelled by the Cascade Model than with the simpler separable Click–Through–Rate model that takes care only of the position–dependent externalities, the Cascade Model has clearly limitations in the way the externalities are represented. We will overcome some of them in this thesis.

While designing new models we will also touch another interesting topic strictly related to SSAs. The topic is still advertising, but from a different point of view: Mobile geo–location advertising [52], specifically, we consider an environments where mobile ads are targeted based on a user’s location (e.g., streets or squares within a city or a district). This field has been identified as a key growth factor for the mobile market. Growing at a compound annual growth rate of 31 percent, the mobile ad market is forecasted to be worth 19.7 billion Euros in 2017—corresponding to 15.5 percent of the total digital advertising market [13].

This problem has not been widely studied and in particular has not been studied from a mechanism design point of view, while, as in the case of SSAs, a crucial ingredient for its success is the development of effective economic mechanisms.
Finally, we will study situations where mechanism design and online learning mash up. Specifically, in practical applications, it is not true that all the parameters of the model designed for the SSAs are known. This fact opens the problem of studying stable mechanism while estimating parameters during the repetitions of the auction. The problem is challenging since it represents one of the first examples where online learning theory and mechanism design are paired to obtain effective methods to learn under equilibrium constraints (notably the truthfulness property). Another application domain where these ideas have been used is crowdsourcing [51].

From a more general point of view, we can summarise this dissertation in the following way, we have studied online pay–per–click/visit auctions (e.g., SSAs and Mobile geo–location advertising) along the four main perspectives that are crucial for the success of an economic mechanism:

- **computational complexity of finding the best allocation**: studying the computational complexity of a problem is important to deeply understand the hardness of the problem. Specifically, this aspect could be crucial in the choice of the model to adopt in order to represent reality and to establish which algorithmic tools are required (are exact algorithms applicable in the real world?);

- **incentive compatibility**: an economic mechanism often is composed of agents that interact. Agents that are usually rational (selfish). Rules are required in order to handle the interaction and to guide it to a stable outcome. Otherwise, the market could become unstable and unpredictable. For this reason, in order to guarantee the stability it is necessary to design incentive compatible mechanisms. This constraint influences also the study of the algorithms;

- **exact and theoretically bounded approximation algorithms**: once the hardness of a problem is known, the problem has to be solved. This requires the design of algorithms. In the case the optimal allocation can be found in polynomial time, the study of algorithm can focus on the exact ones, while in the other situations a study of approximation algorithms is required, otherwise the problem cannot be solved in practical situations. The study of the computational complexity, e.g., providing impossibility results, could guide the choice of which approximation algorithm has to be studied, e.g., with which approximation guarantee. At the same time the requirement of incentive compatibility is strictly related to properties of the allocation algorithms and it could be the case that there is gap between the best approximation guarantee of
Chapter 1. Introduction

an algorithm and the the bound on the best approximation guarantee under the constraint of incentive compatibility;

• online learning: in real world environments, often, we face situations of lack of information, i.e., we do not have all the information required to make the best decision. Thus, in order to apply, in practice, our algorithms and mechanisms it is also important to study ways to handle this uncertainty, e.g., adopting online learning tools. Of particular interest is the study of worst–case bounds over the loss (regret) due to the lack of information w.r.t. the ideal case in which all the information is known. The incentive compatibility requirement influences these bounds.

The ideal goal is the design of very expressive user models admitting very efficient allocation algorithms that can be used in incentive compatible mechanisms with the minimum online learning regret. In this thesis, we show that this is never the case in practice. Indeed, each user model provides a different trade–off in terms of expressiveness, economic stability, approximation bounds, and online regret bounds and therefore there is not the best user model for each scenario, but each scenario potentially requires a different model.

This thesis is the product the work of 3 years as PhD student in Politecnico di Milano. These results are also partially presented in three published papers \[23, 25, 42\]:


• Nicola Gatti, Marco Rocco, Paolo Serafino, and Carmine Ventre: Cascade Model with Contextual Externalities and Bounded User Memory for Sponsored Search Auctions. In Proceedings of AAMAS ’15, 14th AAMAS Conference. Istanbul, Turkey, May, 2015,

and in additional two papers currently under review.
Thesis organization and original contributions In Chapter 2, we provide an introduction to the main topics of the thesis, i.e., mechanism design, online learning (specifically, multi–armed bandit) and Sponsored search auctions. We also review the state–of–the–art of the applications of the two first topics to the latter.

Then we start introducing the original contributions of the thesis. In particular, in Chapter 3 we introduce new models extending the Cascade Model.

The thesis can be divided in two main blocks:

- Chapters 4, 5 and 6 provide theoretical and experimental results over the new models;

- Chapters 7, 8, 9 deal with the mechanism design and learning problems when the CTRs are not known and need to be estimated.

The first three models we introduce in Chapter 3 are: CFNE\textsubscript{sa}, CFNE\textsubscript{aa} and CFNE–\textsubscript{q}i,m. In the thesis we mainly develop the study of the first two models, also considering additional dimensions like the size of the user memory, i.e., for how many slots an ad influences the CTR of the others. Notice that in the Cascade Model the length of the user memory is always considered equal to the number of slots. In the same chapter, we also introduce the first model for Mobile geo–location advertising, which calculates an advertising plan based on the path followed so far by the user and predicted future path. Finally, in Sec. 3.2, we provide a more detailed summary of the results obtained in this thesis.

In the following two chapters we present the theoretical results obtained on the CFNE\textsubscript{sa} and CFNE\textsubscript{aa} models. Specifically, we divide the results on the basis of the length of the user memory: in Chapter 4 we study the case the length of the memory is constant and strictly lower than the number of the slots, while in Chapter 5 we deal with the situation where the length of the memory is equal to the number of slots.

Finally, we close the first part of the thesis with Chapter 6 devoted to the algorithmic study of the model for the mobile geo–location advertising environment.

The second part, the one devoted to lack of information and online learning, begins with Chapter 7 where we study stable mechanisms under different condition of uncertainty when the separable Click–Through–Rate model is adopted. In Chapter 8 we move to the more challenging problem of combining mechanism design and online learning when the Cascade Model is adopted. In both Chapters 7 and 8, we provide bounds on the loss, on the revenue of the auctioneer or on the social welfare, due to the lack of
Chapter 1. Introduction

information, w.r.t. the situation where all the information is known. Finally, in Chapter 9 we experimentally validate our bounds.

The last chapter of the thesis, Chapter 10, is devoted to the conclusions.
CHAPTER 2

Preliminaries

In this chapter we introduce all the main notions on which we will base our work. Given this thesis combines notions of mechanism design together with online learning applied to SSAs, in Sec. 2.1 we introduce the main definitions of mechanism design, then, in Sec. 2.2 we describe the field of online learning focusing on multi–armed bandit algorithms. Finally, in Sec. 2.3 we introduce the SSAs and two main related models, providing also the state–of–the–art of mechanism design and online leaning applied to the SSAs.

2.1 Mechanism Design

The aim of mechanism design \[38\] is to design an economic mechanism composed of an allocation function and a payment function satisfying some desirable properties when agents are rational and have private information representing their preferences—also referred to as the type of the agent. An agent is rational when he acts with the goal of maximising his own welfare. W.l.o.g., mechanism design focuses on mechanisms said direct, in which the only action available to the agents is to report their (potentially non–truthful) type (the revelation principle \[38\]). On the basis of the agents’
reports the mechanism determines the allocation (of resources) to agents and the agents’ payments.

A generic direct-revelation mechanism $M$ is usually described as a tuple $M = (\mathcal{N}, \Theta, V, f, P, U)$:

- $\mathcal{N} = \{1, \ldots, N\}$: the set of rational agents,
- $\Theta$: the set of possible outcomes $\theta$ of the interaction,
- $V = \{V_1, \ldots, V_N\}$: $V_i$ is the set of types of agent $i$, where the type represents the private information,
- $f$: the social choice function $f : V \to \Theta$ that, given the declarations of the agents, produces the outcome, i.e., the social choice,
- $P = \{p_1, \ldots, p_N\}$: the payment rule $p_i : V \to \mathbb{R}$ for agent $i$ determines how much an agent has to pay or is paid on the basis of the agents’ declarations,
- $U = \{u_1, \ldots, u_N\}$: the utility function $u_i : V_i \times \Theta \to \mathbb{R}$ of agent $i$, that defines how good is an outcome for agent $i$, considering also the payments.

Given that the preferences of the agents are private information, one of the main goal of mechanism design is to handle the interaction in order to elicit this information and generating stable interactions. Therefore, the main desirable property of a mechanism is truthfulness, often referred to as incentive compatibility (IC), which requires that reporting the true types constitutes an equilibrium strategy profile for the agents.

When a mechanism is not truthful, agents should find their (untruthful) best strategies on the basis of some possible model about the opponents’ behavior, but, in absence of common information, no normative model for rational agents exists. This leads the mechanism to be economically unstable, given that the agents continuously change their strategies. As it is customary in game theory, there are different solution concepts and consequently there are different notions of truthfulness.

### 2.1.1 Notions of truthfulness

The three most important notions of truthfulness are: dominant strategy incentive compatibility (DSIC), ex post incentive compatibility (ex post IC) and Bayesian incentive compatibility (BIC).

**Property 1.** A mechanism $M$ is DSIC if $u_i(v_i, f(v_i, v_{-i})) \geq u_i(v_i, f(v'_i, v_{-i})) \forall v_{-i} \in V_{-i}, v'_i \in V_i$. 
2.1. Mechanism Design

Notice that we denote by $bv = (v_1, \ldots, v_N) \in V$ the profile of types of the agents, where $v_i$ is the type of agent $i$, and by $v_{-i} \in V_{-i}$ the profile of the types of the agents other than $i$.

A mechanism is *ex post* IC when reporting the true types is a Nash equilibrium. Formally,

**Property 2.** A mechanism $M$ is *ex post* IC iff $u_i(v_i, f(v_i, w_d v_{-i})) \geq u_i(v_i, f(v_i', v_{-i})) \forall v_i' \in V_i$.

Interestingly, DSIC and *ex post* IC are equivalent notions of truthfulness in absence of interdependencies among the types of the agents, that means the type of an agent does not influence the type of the others [46].

Finally, if an agent has a Bayesian prior over the types of the others, we can introduce the notion of BIC. A mechanism is BIC when reporting the true types is a Bayes–Nash equilibrium.

**Property 3.** A mechanism $M$ is BIC iff $E_{v_{-i}}[u_i(v_i, f(v_i, v_{-i}))|v_i] \geq E_{v_{-i}}[u_i(v_i, f(v_i', v_{-i}))|v_i] \forall v_i' \in V_i$.

Since this notion requires that every agent has a Bayesian prior over the types of the other agents, the BIC property is weaker than DSIC and IC is defined in expectation w.r.t. the prior.

In this thesis we will also consider environments where the mechanism design problem includes a source of randomness, not due to the distribution of probabilities over the types of the agents, e.g., random components of the mechanism or the realization of events. We denote with $\omega \in \Omega$ a specific realization of events. In this context, for each notion of truthfulness, we distinguish the case the property holds in expectation, i.e., it holds in expectation w.r.t. the realization of the events and *a posteriori*, i.e., it holds for every possible realization. Obviously the notion of truthfulness in expectation is weaker than the one *a posteriori*.

Given that in this thesis we will not deal with situations where there are interdependences among the types of the agents and we suppose each agent has no prior over the other types, we introduce the definitions of in expectation and *a posteriori* only for the DSIC property.

**Property 4.** A mechanism $M$ is DSIC in expectation iff $\mathbb{E}_\omega[u_i(v_i, f(v_i, v_{-i}), \omega)] \geq \mathbb{E}_\omega[u_i(v_i, f(v_i', v_{-i}), \omega)] \forall v_{-i} \in V_{-i}, v_i' \in V_i$.

**Property 5.** A mechanism $M$ is DSIC *a posteriori* iff $u_i(v_i, f(v_i, v_{-i}), \omega) \geq u_i(v_i, f(v_i', v_{-i}), \omega) \forall v_{-i} \in V_{-i}, v_i' \in V_i \forall \omega \in \Omega$.

Notice that, for simplicity, we will refer to DSIC *a posteriori* only as DSIC. This holds also for the following properties.
2.1.2 Additional desirable properties

The notion of IC, introduced in the previous section, is not the only property to be kept into account when designing an economic mechanism. Three additional properties play a crucial role: *allocative efficiency*, *individual rationality* and *weak budget balance*. Each property, in presence of sources of randomness, can be *in expectation* w.r.t. all the possible realizations of events, or *a posteriori* if it holds for every possible realization.

**Allocative efficiency** A mechanism is Allocative Efficient (AE) when the outcome chosen by the allocation function is the one maximising the social welfare. Formally,

**Property 6.** A mechanism $M$ is AE in expectation iff $f(v) \in \arg \max_{\theta \in \Theta} \mathbb{E}_{\omega}[SW(\theta, \omega)] \ \forall v \in V$.

**Property 7.** A mechanism $M$ is AE a posteriori iff $f(v) \in \arg \max_{\theta \in \Theta} SW(\theta, \omega) \ \forall v \in V \ \forall \omega \in \Omega$.

**Individual rationality** A mechanism is Individually Rational (IR) when each agent is guaranteed to have no loss when reporting truthfully. Formally,

**Property 8.** A mechanism $M$ is IR in expectation iff $\mathbb{E}_{\omega}[u_i(v_i, f(v), \omega)] \geq 0 \ \forall v \in V$ and $\forall i \in N$.

**Property 9.** A mechanism $M$ is IR a posteriori iff $u_i(v_i, f(v)) \geq 0 \ \forall v \in V$ and $\forall i \in N$.

**Weak budget balance** A mechanism is Weakly Budget Balanced (WBB) when the auctioneer is guaranteed to have no loss.

**Property 10.** A mechanism is WBB in expectation iff $\mathbb{E}_{\omega}[\sum_{i \in N} p_i(v, \omega)] \geq 0 \ \forall v \in V$.

**Property 11.** A mechanism is WBB a posteriori iff $\sum_{i \in N} p_i(v) \geq 0 \ \forall v \in V$.

2.1.3 Known implementability results

In a mechanism design problem, the utility function $u_i$ induces a preference relation $\prec$ over the outcomes of the interaction. When the induced preference relation is strictly–total, i.e., the relation is reflexive, complete transitive and antisymmetric, an important negative result is known in literature:
2.1. Mechanism Design

**Theorem 1.** (Gibbard–Satterthwaite Impossibility Theorem) Suppose that the set of outcomes $\Theta$ is finite and $|\Theta| \geq 3$, the preference relations induced by the utility function $u_i, \forall i \in N$, are strictly–total and the allocation function $f$ is onto. Then, the allocation function $f$ can be part of a DSIC mechanism iff it is dictatorial.

**Definition 1.** A function $f$ is onto (a.k.a. surjective) if for every element $y$ of the image $Y$, there exists an element $x$ of the domain $X$ s.t. $f(x) = y$.

**Definition 2.** An allocation function $f$ is dictatorial when the outcome selected by $f$ is s.t. $\forall v f(v) = \theta$, where $u_i(v_i, \theta) > u_i(v_i, \theta') \ \forall \theta' \in \Theta \setminus \{\theta\}$. Agent $i$ is called the dictator.

Notice that there is a strict relation between the Gibbard–Satterthwaite impossibility theorem and the Arrow’s one, the former being a special case of the latter.

Given the negative result provided by the Gibbard–Satterthwaite impossibility theorem, in order to overcome this limitation, one idea is to focus on mechanisms that do not fulfill one of the three hypothesis of the theorem. One possibility is to consider utility functions inducing rational preference relations, i.e., reflexive, complete and transitive, but not antisymmetric, relations, for example considering the quasi–linear environment.

In a quasi–linear environment the utility functions are quasi linear \cite{38}:

**Definition 3.** Utilities are quasi linear when the utility function $u_i$ is defined as $u_i(v_i, f(v)) = val(v_i, f(v)) - p_i(v)$. Where $val(v_i, f(v))$ is the value obtained by agent $a_i$ when his type is $v_i$ and the outcome of the mechanism is $f(v)$.

In this context, if the allocation function $f$ is AE, we can design DSIC mechanisms adopting the Groves payments:

**Theorem 2.** (Groves’ theorem) Given an AE allocation function $f$. We can design a DSIC mechanism if we adopt the following payment scheme:

$$ p_i(v) = h_i(v_{-i}) - \sum_{j \in N; j \neq i} val_j(v_j, f(v)) $$

where $h_i(\cdot)$ is a function that does not depend on the declaration of agent $i$.

When the functions $val_i$ are s.t. $val_i : V_i \times \Theta \rightarrow \mathbb{R}$, applying the Green–Laffont theorem \cite{29}, the if of the previous theorem becomes an if and only if.
In particular, the most famous payment among the family of Groves’ one is the payment adopting the Clarke pivot, i.e., the Vickrey–Clarke–Groves (VCG) payments:

\[
p_i(v) = \sum_{j \in N: j \neq i} \text{val}_j(v_j, f(v_{-i})) - \sum_{j \in N: j \neq i} \text{val}_j(v_j, f(v))
\]

\[
= SW(f_{-i}(v_{-i}), v_{-i}) - SW_{-i}(f(v), v)
\]

Where \(SW(f_{-i}(v_{-i}), v_{-i})\) is the social welfare, i.e., how good the solution is for the population of agents, of the allocation returned by the allocation function when agent \(i\) does not participate to the interaction, while \(SW_{-i}(f(v), v)\) is the social welfare of the allocation obtained when \(i\) participates to the interaction, the reports of the agents are \(v\), but the contribution of \(i\) to the social welfare is not taken into account.

Removing the requirement that the allocation function \(f\) is AE, we can characterise additional DSIC mechanisms based on the two concepts of \textit{affine maximiser} allocation function \cite{46} and \textit{maximal in range} (MIR) \cite{44}.

**Definition 4.** An allocation function \(f\) is called an affine maximizer if for some subrange \(\Theta' \subseteq \Theta\), for some weights \(w_i \in \mathbb{R}^+\), \(i \in N\), depending only on the agent, and for some constants \(c_\theta \in \mathbb{R}\) associated to the outcome \(\theta\), then \(f(v) \in \arg \max_{\theta \in \Theta'} (c_\theta + \sum_{i \in N} w_i \text{val}_i(v_i, \theta))\).

**Definition 5.** Given an allocation function \(f\) that is an affine maximiser, we can design a DSIC mechanism adopting the following payment scheme:

\[
p_i(v) = h_i(v_{-i}) - \sum_{j \in N: j \neq i} \frac{w_j}{w_i} \text{val}_j(v_j, f(v)) - \frac{cf(v)}{w_i}
\]

where \(h_i(\cdot)\) is a function that do not depend on the declaration of agent \(i\).

Moreover, Roberts’ theorem states that for unrestricted domains with at least 3 possible outcomes, an incentive compatible mechanisms must satisfy the characteristics identified in Def. 4 and Def. 5.

**Theorem 3.** (Roberts) If \(|\Theta| \geq 3\), \(f\) is onto in \(\Theta\), \(v_i \in \mathbb{R}^{|\Theta|}\) for every \(i \in N\), and a mechanism \(M\) is DSIC, then its allocation function \(f\) is an affine maximizer.

In particular, when an allocation function \(f\) is an affine maximiser s.t. \(\Theta' \subset \Theta\) and \(w_i = 1\), \(\forall i \in N\), we say that the allocation function \(f\), \(f(v) \in \arg \max_{\theta \in \Theta'} \sum_{i \in N} \text{val}_i(v_i, \theta)\), is \textit{maximal in range} and, in order to
obtain a DSIC mechanism, it can be coupled with the following VCG–like payments:

\[ p_i(v) = h_i(v_{-i}) - \sum_{j \in N : j \neq i} val_j(v_j, f(v)). \]

Finally, we focus on a subset of quasi–linear environment, the single–parameter linear environment:

**Property 12.** A mechanism \( M \) is single–parameter linear when a single parameter \( v_i \in \mathbb{R} \) describes the type of the agent and the utility function \( u_i \) can be written as \( u_i(v_i, v) = load_i(f(v))v_i - p_i(v) \). Where \( load_i(\theta) \) is the load of agent \( i \) when the outcome of the mechanism is \( \theta \).

Given a single–parameter linear mechanism, it is possible to design payments \( p_i(v) \) s.t. the resulting mechanism is DSIC, if and only if, the \( load_i(f(v_i, v_{-i})) \) is monotone in the report \( v_i \) of agent \( i \), i.e., \( load_i(f(v_i, v_{-i})) \) is an increasing (or decreasing) function of \( v_i \). In the case of increasing function, the DSIC mechanism is obtained imposing payments [5, 41]:

\[
    p_i(v) = h(v_{-i}) + v_i \cdot load_i(f(v)) - \int_0^{v_i} load_i(f(x, v_{-i}))dx \tag{2.1}
\]

where \( h(v_{-i}) \) is an arbitrary function.

One of the problems of Payment 2.1 is that they include an integral that may be not easily computable. However, by adopting DSIC in expectation (w.r.t. the randomness of the mechanism), such integral can be easily estimated by using samples [6]. Another drawback of the payments described in [5, 41] is that they require the off–line evaluation of the social welfare of the allocations for some agents’ types different from the reported ones and this may be not possible in many practical situations. A way to overcome this issue is to adopt the result presented in [10], in which the authors propose an implicit way to calculate the payments. More precisely, given an allocation function in input, a random component is introduced such that with a small probability the reported types of the agents are modified to obtain the allocations that are needed to compute the payments in [5, 41]. The resulting allocation function is less efficient than the allocation function given in input, but the computation of the payments is possible and it is executed online. More details on this technique are provided when we will adopt it in Sec. 7.2.2.
2.2 Multi–Armed Bandit

The multi–arm bandit (MAB) [48] is a simple yet powerful framework formalizing the online decision–making problem under uncertainty. Historically, the MAB framework finds its motivation in optimal experimental design in clinical trials, where two new treatments, say $A$ and $B$, need to be tested. In an idealized version of the clinical trial, $T$ patients are sequentially enrolled in the trial, so that whenever a treatment is tested on a patient, the outcome of the test is recorded and it is used to choose which treatment to provide to the next patient. The objective is to provide the best treatment to the largest number of patients. This raises the challenge of balancing the collection of information and the maximization of the performance, a problem usually referred to as the exploration–exploitation trade–off. In fact, on the one hand, it is important to gather information about the effectiveness of the two treatments by repeatedly providing them at different patients (exploration). On the other hand, in order to meet the objective, as an estimation of effectiveness of the two treatments is available, the (estimated) best treatment should be selected more often (exploitation). This scenario matches with a large number of applications, such as online advertisements, adaptive routing, cognitive radio.

In general, the MAB framework can be adopted whenever a set of $N$ arms (e.g., treatments, ads) is available and the rewards (e.g., effectiveness of a treatment, click–through rate of an ad) associated with each of them are random realizations from unknown distributions. Although this problem can be solved by dynamic programming methods and notably by using the Gittins index solution [27], this requires a prior over the distribution of the reward of the arms and it is often computationally heavy (high–degree polynomial in $T$). More recently, a wide range of techniques have been developed to solve the bandit problem. In particular, these algorithms formalize the objective using the notion of regret, which corresponds to the difference in performance over $T$ steps between an optimal selection strategy which knows in advance the performance of all the arms and an adaptive strategy which learns over time which arms to select. Although a complete review of the bandit algorithms refer to [16], here, we only discuss two results which are relevant to the rest of the thesis. The exploration–separated algorithms solve the exploration–exploitation trade–off by introducing a strict separation between the exploration and the exploitation phases. While during the exploration phase all the arms are uniformly selected, in the exploitation phase only the best estimated arm is selected until the end of the experiment. The length $\tau$ of the exploration phase is critical to guarantee the
success of the experiment and it is possible to show that if properly tuned, the worst–case cumulative regret scales as $O(T^{\frac{2}{3}})$.

Another class of algorithms interleaves exploration and exploitation and rely on the construction of confidence intervals for the reward of each arm. In particular, the upper–confidence bound (UCB) algorithm \cite{9} gives an extra exploration bonus to arms which have been selected only few times in the past and it achieves a worst–case cumulative regret of order $O(T^{\frac{1}{2}})$. Although this represents a clear improvement over the exploration–separated algorithms in some web advertising applications considered in this paper, it is not possible to preserve incentive compatibility when exploration and exploitation are interleaved over time.

### 2.3 Sponsored Search Auction models

We now introduce a problem which importance increased a lot of in the recent years, i.e., Sponsored Search Auctions (SSAs). In SSAs a publisher selects ads to be placed in a number of slots on a Web page and an advertiser pays the publisher only when its ad is clicked.

The main idea of the problem is that when a user submits a query on a search engine, the search engine returns links related to the query, but it embeds into the results some sponsored links, i.e., advertising (ads). Given the importance of SSA in the revenue of a search engine, it is important, for an algorithm, to select the best way the ads can be allocated into the slots (position of the webpage) devoted to sponsored links.

A lot of papers deal with this problem, trying to identify the best way to model the behaviour of the user when looking at the ads.

We introduce the two following models representing in different ways the behaviour of the user:

- *separable Click–Through–Rate model* \cite{3, 21}, i.e., the most simple model that, with some variations, is still currently used by some search engines;

- *Cascade Model* \cite{2, 35} where the user behaviour is Markovian, observing the slots from the top one to the bottom one.

Even if the two models represent in a different way the behaviour of the user, they share most of the parameters:

- $\mathcal{N} = \{1, \ldots, N\}$ is the set of indices of the ads $a_i$ ($i \in \mathcal{N}$). We define also an additional fictitious ad: $a_{\perp}$. W.l.o.g. we assume each
advertiser to have a single ad, so each advertiser \( i \) can be identified with ad \( a_i \);

- \( K = \{1, \ldots, K\} \) is the set of indices of the slots \( s_m (m \in K) \) ordered from the top to the bottom. We define \( K_c = K \cup \{K + 1, \ldots, N\} \) (when \( N > K \));

- \( q_i \in [0, 1] \) is the quality of ad \( a_i \) (i.e., the probability a user will click ad \( a_i \) when observed). We define \( q_\perp = 0 \);

- \( v_i \in V_i \subseteq \mathbb{R}^+ \) is the value obtained by advertiser \( i \) when ad \( a_i \) is clicked by a user, \( v = (v_1, \ldots, v_K) \) is the value profile. We define \( v_\perp = 0 \);

- \( \hat{v}_i \in V_i \) is the value reported by advertiser \( i \), \( \hat{v} = (\hat{v}_1, \ldots, \hat{v}_K) \) is the reported value profile;

- \( \Theta \) is the set of allocations of ads to slots, where each ad, except \( a_\perp \), can be allocated in at most one slot; we assign \( a_\perp \) to \( s_m \) to denote that no ad is displayed in \( s_m \), and we assign \( a_i \) to a slot \( s_m, m > K \), to denote that \( a_i \) is not displayed.

A generic allocation \( \theta \in \Theta \) is denoted as \( \theta = (a_1, \ldots, a_K) \), where the ads are ordered from the one allocated in the first slot to the one allocated in the last one. In the case we are interested also to the order of the non–allocated ads, with abuse of notation, \( \theta \) will denote an allocation \( \theta = (a_1, \ldots, a_N) \).

Finally, we define two maps \( \pi : N \times \Theta \to K_c \) and \( \alpha : K_c \times \Theta \to N \cup \{\perp\} \) such that \( \pi(i; \theta) \) returns the index of the slot in which \( a_i \) is displayed in allocation \( \theta \) and \( \alpha(m; \theta) \) returns the index of the ad displayed in slot \( s_m \) in allocation \( \theta \). Given \( \theta \in \Theta \), we have that \( \pi(i; \theta) = m \) if and only if \( \alpha(m; \theta) = i \).

Separable Click–Through–Rate model The main idea of the separable Click–Through–Rate model is that the probability ad \( a_i \) is clicked when allocated in position \( s_m \) can be represented as the product of two quantities: one that is only an ad–specific term, i.e., the quality \( q_i \) of the ad \( a_i \), and the second that is position–dependent, i.e., the factorised prominence \( \lambda_m \) of the slot \( s_m \). \( \lambda_m \) represents the probability the user observes slot \( s_m \) after having observed slot \( s_m \). The prominence \( \Lambda_m \) is the probability that slot \( s_m \) is observed by the user and is defined in the following way: \( \Lambda_m = \prod_{k=1}^{m-1} \lambda_k \).

Notice that \( \Lambda_m = 0 \) when \( m > K \) and w.l.o.g. \( \Lambda_1 = 1 \).

The probability ad \( a_i \) is clicked by the user is called click–through rate \( \text{(CTR)} \) and, in an allocation \( \theta \) is formally defined as: \( CTR_i(\theta) = \Lambda_{\pi(i; \theta)} q_i \).
The total value of an allocation \( \theta \) is called Social Welfare (SW): 
\[
SW(\theta, v) = \sum_{i \in N} CTR_i(\theta)q_iv_i = \sum_{m \in K} \Lambda_m q_{\alpha(m;\theta)}v_{\alpha(m;\theta)}.
\]

We consider the optimal allocation as the one maximising the SW, thus the optimal allocation function \( f \) is the one s.t. \( f(v) \in \arg\max_{\theta \in \Theta} SW(\theta, v) \). The computational complexity of finding the optimal allocation when the separable Click–Through–Rate model is adopted is polynomial in the size of the problem, indeed it only requires to allocate the ads in decreasing order of \( q_i v_i \) from the top to the bottom slot.

**Cascade Model** One of the weaknesses of the separable Click–Through–Rate model is that the ad the user observes does not influence the probability the user will observe the following slots. This seems unrealistic, e.g., an ad with very low quality could convince the user to completely disregard the other ads on the page. Thus, the Cascade Model proposed in [2, 35] introduces a new parameter called continuation probability: \( \gamma_i, \forall i \in N \). This parameter represents the probability the user will move to the next slot after having observed ad \( a_i \), independently on whether the user clicked on \( a_i \) or not. The cumulative continuation probability is defined as 
\[
\Gamma_i(\theta) = \prod_{m < \pi(i;\theta)} \gamma_{\alpha(m;\theta)}.
\]

Specifically, in the Cascade Model the user’s behaviour is represented as a Markov chain where states correspond to the slots, which are observed sequentially from the top to the bottom. The transition probability from a state to another (from a slot to the next one) corresponds to the probability the user observes ad \( a_i \) displayed in the next slot; with the remaining probability the user stops observing the ads. The transition probability is given by the product of two terms: one depending on the starting slot (i.e., \( \lambda_{\pi(i;\theta)−1} \)), position–dependent externality, and the second depending on the ad that precedes \( a_i \) in the current allocation \( \theta \) (i.e., \( \gamma_{\alpha(\pi(i;\theta)−1;\theta)} \)), ad–dependent externality. Given an allocation \( \theta \), the CTR of \( a_i \) is computed as 
\[
CTR_i(\theta) = \Lambda_{\pi(i;\theta)}\Gamma_i(\theta)q_i,
\]
thus 
\[
SW(\theta, v) = \sum_{i \in N} CTR_i(\theta)q_iv_i = \sum_{m \in K} \Lambda_m \Gamma_{\alpha(m;\theta)}q_{\alpha(m;\theta)}v_{\alpha(m;\theta)}.
\]

The computational complexity of finding the optimal allocation when the Cascade Model is adopted is unknown, as well as the complexity of the decision problem that is commonly believed, but not proved, to be NP–hard [35]. However, the allocation problem can be solved in reasonable time adopting a branch–and–bound algorithm (details in the next section) for instances with \( K \leq 7 \) [23].
2.3.1 SSA and Mechanism Design

In this section, we study SSA as a strategic problem, indeed, being rational, the agents could misreport the value they communicate whether this allows the advertiser to gain more utility than declaring the true value $v_i$ (the private information of the agent). Therefore, it is necessary to study the problem into the mechanism design framework.

We now provide a brief review of the state–of–the–art of mechanism design applied to the two previously introduced user’s models.

**Separable Click–Through–Rate model** The algorithm that allocates the ads in decreasing order of $q_i v_i$ finds in polynomial time the allocation $\theta^*$ that maximises the SW. When coupled with the VCG payments, the obtained mechanism is DSIC in expectation, IR in expectation and WBB a posteriori. The problem is that with VCG payments an advertiser could pay even when its ad has not been clicked.

In order to satisfy DSIC and IR a posteriori, we have to resort a payment scheme called pay–per–click, i.e., an agent pays only when its ad is clicked. The new payments are defined in the following way:

$$ p^*_i, c_i(\hat{v}, click^i_{\pi(i; \theta^*)}) = \frac{SW(\theta^{-i}, \hat{v}^{-i}) - SW_{-i}(\theta^*, \hat{v})}{\Gamma_{\pi(i; \theta^*)}(\theta^*) q_i} \times click^i_{\pi(i; \theta^*)} $$

where we denote by $click^i_{s_m} \in \{0, 1\}$ the no–click/click event for ad $a_i$ allocated in slot $s_m$. Notice that these payments are such that $E[p^*_i, c_i(\hat{v}, click^i_{\pi(i; \theta^*)})]$ are equal to the VCG payments. The expectation is w.r.t. the click event, which is distributed as a Bernoulli random variable with parameter coinciding with the CTR of ad $a_i$ in allocation $\theta^*$.

**Cascade Model** The computational complexity class of finding the optimal allocation of an SSA problem when the user’s behaviour is modelled through the Cascade Model is still unknown. Currently, literature provides only algorithms with exponential computational complexity. Obviously, coupling these algorithms that are AE and the pay–per–click VCG payments, the obtained mechanisms are DSIC, IR and WBB a posteriori.

In [23], we proposed a simple algorithm based on branch and bound for computing the efficient allocation returned by $f_E$ (i.e., the efficient allocation function) to be employed in the VCG mechanism. It is reported in Algorithm 1. Basically, it is a recursive backtracking algorithm. Initially, the algorithm is called with $(\theta_0, 1, 0)$ where in $\theta_0$ all the slots are associated with $a_\perp$. Then, at each call, the algorithm, at Step 3, adds a new ad $a_i$ to the
partial current allocation \( \theta \) unless all the slots are already assigned (Step 2), checks at Step 5, by using an admissible heuristic \( \text{heu}(\theta) \), whether the new partial allocation can lead to an allocation strictly better than the best allocation \( \theta^* \) (whose value is stored in \( \text{best} \)) found so far and, in the affirmative case, the algorithm is recursively called from the new allocation \( \theta \) (Step 6).

We define \( \text{heu}(\theta) \) as the minimum between:

- the optimal total expected value given by the remaining slots when \( \gamma_i = 1 \) for all the \( a_i \) that are not allocated yet,
- the optimal total expected value given by the remaining slots when for all \( \lambda_m \) it holds \( \lambda_m = \lambda_k \) for every \( m > k \), where \( s_k \) is the last allocated slot in \( \theta \).

Both above values are computable in polynomial time. Heuristic \( \text{heu}(\theta) \) obviously provides an overestimation of the optimal value of the ads that can be allocated in the remaining slots of \( \theta \) and therefore the algorithm is sound. The algorithm runs with \( O\left( N^K \left( N \log N + N K \right) \right) \), where \( O\left( N \log N + N K \right) \) is the complexity of computing the heuristic. The VCG payments require the determination of the best allocation in \( K \) subproblems, as described in [43]. Therefore the complexity of computing \( f_E \) and the VCG payments with our algorithm is \( O\left( K N^K \left( N \log N + N K \right) \right) \). The experimental evaluation we performed in [23] shows that with \( K > 7 \) and a large number of ads the algorithm does not terminate in reasonable time, thus, approximations are needed.

**Algorithm 1** \( f_E(\text{Allocation}(\theta, j, \text{best})) \)

1. \( \theta^* \leftarrow \theta \)
2. if \( m \leq K \) then
3.   for all \( i \neq \alpha(k; \theta) \) \( \forall k < m \) do
4.     \( \alpha(k; \theta) \leftarrow i \)
5.   if \( \sum_{i \in N} \text{CTR}_i(\theta) \cdot \hat{v}_i + C_{\theta(s_j)}(\theta) \cdot \text{heu}(\theta) > \text{best} \) then
6.     \( (\theta', \text{best}') \leftarrow f_E(\text{Allocation}(\theta, j + 1, \text{best})) \)
7.   if \( \text{best}' > \text{best} \) then
8.     \( \text{best} \leftarrow \text{best}' \)
9.     \( \theta^* \leftarrow \theta' \)
10. end if
11. end if
12. end for
13. end if
14. return \( (\theta^*, \text{best}) \)

On the contrary, a result in [35] shows that, restricting the attention to
those instances of the problem where all the $\Lambda_m, m \in \mathcal{K}$, are equal to 1, i.e., ignoring the position–dependent externality, finding the optimal allocation is a polynomial task that can be solved with Dynamic Programming (DP) techniques. We call this environment only ad–dependent model. Obviously, in this scenario, it is possible to design a DSIC mechanism adopting the pay–per–click VCG payments.

In the general case, where $\Lambda_m$ are not constrained, literature focused on approximation algorithms, i.e., algorithms not returning the optimal allocation $\theta^*$, but returning, in polynomial time, solutions $\theta$ with bounds on their quality ($\frac{\text{SW}(\theta,v)}{\text{SW}(\theta^*,v)}$).

In [23] we showed that the optimal allocation of the Cascade Model cannot be approximated by using algorithms returning the optimal allocations for the only ad–dependent model or separable CTR model. We initially focus on the relation between the only separable CTR model and the Cascade Model.

**Theorem 4.** The optimal allocation of the separable CTR model ($\gamma_i = 1$ $\forall i \in \mathcal{N}$) is not a constant approximation of the optimal allocation in the Cascade Model.

**Proof.** The proof is by counterexample. Consider the following setting (where $\epsilon > 0$ is an arbitrarily small parameter):

- ad $a_1$ with $v_1q_1 = \overline{v} + \epsilon$ and $\gamma_1 = 0$;
- ads $a_2, \ldots, a_{K+1}$ with $v_iq_i = \overline{v}$ and $\gamma_1 = 1$;
- $\Lambda_1 = 1, \Lambda_2, \ldots, \Lambda_K = 1 - \epsilon$.

The optimal allocation when of the separable CTR model is: $(a_1, a_2, \ldots, a_K)$, corresponding to assigning the highest–valuation ad to the top slot, while assigning the other ads arbitrarily to the other slots. This allocation, when the Cascade Model is adopted, has a value of $\overline{v} + \epsilon$. The value of the allocation $(a_2, a_3, \ldots, a_{K+1})$ with the Cascade Model is $\overline{v} + (1 - \epsilon)(1 - K)\overline{v}$. The ratio is $\frac{\overline{v} + \epsilon}{\overline{v} + (1 - \epsilon)(1 - K)\overline{v}}$ that is asymptotically $O\left(\frac{1}{K}\right)$. Therefore, the value of the optimal allocation of the separable CTR model does not provide any constant approximation of the Cascade Model.

We have the same result when focusing on the relation between the only ad–dependent model and the Cascade Model.

**Theorem 5.** The optimal allocation of the only ad–dependent model ($\Lambda_m = 1 \forall m \in \mathcal{K}$) is not a constant approximation of the optimal allocation of the Cascade Model.
2.3. Sponsored Search Auction models

Proof. The proof is by counterexample. Consider the following setting (where \( \epsilon > 0 \) is an arbitrarily small parameter):

- ad \( a_1 \) with \( v_1 q_1 = \epsilon \) and \( \gamma_1 = 1 \);
- ad \( a_2 \) with \( v_2 q_2 = \overline{v} \) and \( \gamma_2 = 1 - \epsilon \);
- \( \Lambda_1 = 1 \) and \( \Lambda_2 = \epsilon \).

The optimal allocation when only ad–dependent externalities are present is: \((a_1, a_2)\). This allocation, with the Cascade Model, has a value of \( \epsilon + \epsilon \overline{v} \). The value of the allocation \((a_2, a_1)\) with the Cascade Model is \( \overline{v} + (1 - \epsilon) \epsilon \). The ratio is \( \frac{\epsilon + \epsilon \overline{v}}{\overline{v} + (1 - \epsilon) \epsilon} \) that is asymptotically \( O(\epsilon) \). Therefore, the value of the optimal allocation of the only ad–dependent model does not provide any constant approximation for the Cascade Model given that \( \epsilon \) can be arbitrarily small.

The above negative results push for the need of more sophisticated algorithms approximating the Cascade Model.

In [35] the authors propose an approximation algorithm with constant bound: \( \frac{1 - \epsilon}{4} \), but in our paper [23], we proved that the pseudopolynomial–time \( \frac{1}{4} \)–approximation algorithm from which the authors derived their \( \frac{1 - \epsilon}{4} \) approximation algorithm is not monotone. Thus, it does not lead to a DSIC mechanism. The proof can be extended to the \( \frac{1 - \epsilon}{4} \)–approximation algorithm.

The \( \frac{1}{4} \)–approximation algorithm works as follows, it finds the allocation \( \theta \) maximizing \( \sum_{i \in N} \Lambda_{x(i; \theta)} q_i \hat{v}_i \) under the constraints that:

- all the allocated ads, except the last one, are such that \( q_{\alpha(m, \theta)} \hat{v}_{\alpha(m, \theta)} \geq q_{\alpha(m+1, \theta)} \hat{v}_{\alpha(m+1, \theta)} \);
- letting \( s_k \) be the last slot with an ad different from \( a_{\perp} \), \( \Gamma_{\alpha(k; \theta)}(\theta) \geq \frac{1}{2} \).

The last constraint can be also represented as \( \sum_{m=1}^{k-1} \log_2 \frac{1}{\gamma_{\alpha(m; \theta)}} \leq 1 \).

The property of monotonicity of an allocation function can be easily captured in the case of the sponsored search auctions with externalities observing Fig. [2.1] increasing \( \hat{v}_i \) and keeping fix all the others \( \hat{v}_{-i} \) the allocation \( \theta \) changes and the \( CTR_{i}(\theta) \) monotonically increases. We state the following.

Proposition 1. The allocation function induced by the above \( \frac{1}{4} \)–approximation algorithm is not monotone.
Proof. Consider an auction with 5 ads and 4 slots, with the parameters reported in Fig. 2.2.

We can show that the allocation function is not monotonic as \( x \) varies. The allocations chosen by the allocation function when \( x \in [1.554, 2.500] \) are:

- \( 1.554 \leq x \leq 2.23 \): the allocation is, from the first slot to the last allocated one, \((a_2, a_3, a_4, a_5)\) and \( CTR_4(\theta) = \Lambda_3 \gamma_2 \gamma_3 q_4 = 0.081 \),

- \( 2.23 < x \leq 2.5 \): the allocation is \((a_1, a_4, a_5)\) and \( CTR_{a4}(\theta) = \Lambda_2 \gamma_1 q_4 = 0.060 \).

Therefore, increasing \( x \) along \([1.554, 2.5]\), the \( CTR_4(\theta) \) decreases from 0.081 at \( x \leq 2.23 \) to 0.06 at \( x > 2.23 \). This shows that the allocation function is not monotonic. By changing the values of the parameters, there are cases in which the non–monotonicity is related to each possible advertiser.

As shown in [5], with linear utility functions, the monotonicity of the allocation function is necessary to have incentive compatibility. Thus, no

\[
\begin{array}{|c|c|c|c|}
\hline
a_i & q_{a_i} & \hat{v}_{a_i} & c_{a_i} \\
\hline
a_1 & 0.500 & 2.0 & 0.50 \\
\hline
a_2 & 0.300 & 3.0 & 0.90 \\
\hline
a_3 & 0.250 & 2.0 & 0.90 \\
\hline
a_4 & 0.200 & x & 1.00 \\
\hline
a_5 & 0.111 & 10.0 & 0.10 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
\Lambda & \\
\hline
s_1 & 1.00 \\
\hline
s_2 & 0.60 \\
\hline
s_3 & 0.50 \\
\hline
s_4 & 0.36 \\
\hline
\end{array}
\]

Figure 2.1: Relation between \( CTR_{a_i} \) and \( \hat{v}_{a_i} \).

Figure 2.2: Parameters of the counterexample used in the proof of Proposition 1.
incentive compatible mechanism can be designed when the above allocation function is used. In a similar way it is possible to produce a proof of non–monotonicity for the $\frac{1-\epsilon}{4}$–approximation algorithm described in [35].

2.3.2 SSA and MAB

We begin this new section with a question: is it true, in real applications, that all the parameters, except $v$, are known to the auctioneer? In some environments clearly it is not the case, indeed in many practical problems, the parameters (i.e., $q_i$ and $\gamma_i$) are not known in advance by the auctioneer and must be estimated at the same time as the auction is deployed. This section is devoted to the introduction of the state–of–the–art in situations where there is lack of information. Specifically, in this environment it is necessary to consider an iterative process where the auction is repeated over $T$ steps, and at each step different estimates of the CRTs are used. Thus, we are moving the problem into the field of online learning. The fact of using estimates introduces the need of the definition of a tradeoff between exploring different possible allocations so as to collect information about the parameters and exploiting the estimated parameters so as to implement a truthful high–revenue auction (i.e., a VCG mechanism). This problem could be easily casted as a multi–armed bandit problem [48] and standard techniques could be used to solve it, e.g., [8]. Nonetheless, such an approach would completely overlook the strategic dimension of the problem: advertisers may choose their reported values at each step $t$ to influence the outcome of the auction at $t$ and/or in future steps after $t$, in order to increase the cumulative utility over all the steps of the horizon $T$. Thus, in this context, a new definition of truthfulness is required. A mechanism is truthful when reporting the truthful valuation maximizes the cumulative utility over all the horizon $T$. A mechanism is defined DSIC if it is truthful even supposing that the advertisers know everything (including, e.g., the ads that will be clicked at each step $t$ if displayed) or it can be DSIC in expectation. Hence we study the worst case in which only the auctioneer has lack of information. We adopt three forms of DSIC in expectation: DSIC in expectation w.r.t. the click realizations and a posteriori w.r.t. the realizations of the random component of the mechanism (if such a component is present), DSIC in expectation w.r.t. the realizations of the random component of the mechanism and a posteriori w.r.t. the click realizations, and, finally, DSIC in expectation w.r.t. both randomizations. We consider DSIC in expectation w.r.t. the click realizations weaker than DSIC in expectation w.r.t. the realizations of the randomness mechanism since each advertiser
could control the clicks by using software bots.

The problem we are now facing is more challenging than other learning problems, indeed the exploration–exploitation dilemma must be solved so as to maximize the revenue of the auctioneer under the hard constraint of incentive compatibility. We are thus combining the two field of mechanism design and online learning.

**Model**  Let $M$ be a DSIC mechanism running over $T$ steps. We assume, as it is common in practice, that the advertisers’ reports can change during these $T$ steps. At each step $t$, $M$ defines an allocation $\theta_t$ and prescribes an expected payment $p_{i,t}(\hat{v})$ for each ad $a_i$. In the thesis we will analyse mechanisms $M$ with two different objectives: minimization of the loss of the auctioneer and minimization of the loss in the social welfare.

We first introduce the problem where the objective of $M$ is to minimize the loss of the auctioneer w.r.t. the revenue provided by the VCG mechanism computed on the basis of the actual parameters preserving the properties of IR and WBB. More precisely, we measure the performance of $M$ as its cumulative regret over $T$ steps:

$$ R_T(M) = T \sum_{i=1}^{N} p^*_i(\hat{v}) - \sum_{t=1}^{T} \sum_{i=1}^{N} p_{i,t}(\hat{v}). $$

where with $p^*_i$ we are referring to the VCG payments. We remark that the regret is not defined on the basis of the pay–per–click payments asked on a specific sequence of clicks, but on the expected payments $p_{i,t}(\hat{v})$. Furthermore, since the learning mechanism $M$ estimates the CTRs from the observed (random) clicks, the expected payments $p_{i,t}(\hat{v})$ are random as well. Thus, in this dissertation we will study the expected regret:

$$ R_T(M) = \mathbb{E}[R_T(M)], \quad (2.2) $$

where the expectation is taken w.r.t. random sequences of clicks and possibly the randomness of the mechanism. The mechanism $M$ is a no–regret mechanism if its per–step regret $R_T(M)/T$ decreases to 0 as $T$ increases, i.e., $\lim_{T \to \infty} R_T(M)/T = 0$.

We now focus on the case where the performance [11][28] is evaluated on the basis of the social welfare regret, that measures the performance of $M$ as follows:

$$ R^{SW}_T(M) = T \cdot SW(\theta^*, \hat{v}) - \sum_{t=1}^{T} SW(\bar{\theta}_t, \hat{v}), $$
where $\tilde{\theta}_t$ is the allocation prescribed by the learning mechanism at time step $t$. As before, since the learning mechanism $M$ estimates the parameters from the observed (random) clicks, we will study the expected regret:

$$R_T^{SW}(M) = \mathbb{E}[R_T^{SW}(M)].$$

(2.3)

We notice that minimizing the social welfare regret $R_T^{SW}(M)$ does not coincide with minimizing $R_T(M)$. In fact, once the quality estimates are accurate enough, such that $\theta_t$ is equal to $\theta^*$, the social welfare regret drops to zero. On the other hand, since $p_{i,t}(\hat{v})$ is defined according to the estimated qualities, $R_T(M)$ might still be positive even if $\theta_t = \theta^*$.

**State-of-the-art** The study of the problem when $K = 1$, thus when separable CTR model and Cascade Model are equivalent, is well established in literature. More precisely, the properties required to have a DSIC mechanism are studied in [20] and it is shown that any learning algorithm must split the exploration and the exploitation in two separate phases in order to incentivize the advertisers to report their true values. This condition has a strong impact on the regret $R_T(M)$ of the mechanism. In fact, while in a standard bandit problem the distribution–free regret is of order $\Omega(T^{1/2})$, in single–slot auctions ($K = 1$), DSIC mechanisms have a regret $\Omega(T^{2/3})$. In [20] a truthful learning mechanism is designed with a nearly optimal regret of order $\tilde{O}(T^{2/3})$. Similar structural properties for DSIC mechanisms are also studied in [11] and similar lower–bounds are derived for the social welfare regret. The authors show in [10] that, by introducing a random component in the allocation function and resorting to truthfulness in expectation w.r.t. the realizations of the random component, the separation of exploration and exploitation phases can be avoided. In this case, the upper bound over the regret of the social welfare is $O(T^{1/2})$ matching the best bound of standard distribution–free bandit problems. However, the payments of this mechanism suffer of potentially high variance. Although it is expected that with this mechanism also the regret over the auctioneer revenue is of the order of $O(T^{1/2})$, no formal proof is known.

On the other hand, the study of the problem when $K > 1$ is still mostly open. In this case, a crucial role is played by the user model. Two papers deal with this problem: [49] and [24]. In the first, the authors characterize DSIC mechanisms and provide theoretical bounds over the social welfare regret. More precisely, the authors assume a simple user model in which

\[ \tilde{O}/\tilde{\Omega}/\tilde{\Theta} \] notation hides both constant and logarithmic factors, e.g., in the case of $\tilde{O}$ we say the regret is $\tilde{O}(T^{2/3})$ if there exist $a$ and $b$ such that the regret is $\leq aT^{2/3} \log^b T$. 


the CTR itself depends on the ad \(a_i\) and the slot \(s_m\). This model differs from the Cascade Model. It can be easily shown that the model studied in [49] does not include and, at the same time, is not included by the Cascade Model. However, the two models correspond when the CTRs are separable in two terms in which the first is the agents’ quality and the second is a parameter in \([0, 1]\) monotonically decreasing in the slots (i.e., separable CTR model). In [49], the authors show that when the CTRs are unrestricted (e.g., they are not strictly monotonically decreasing in the slots), and the parameters \(CTR_{i,m}\) (i.e., the CTR of ad \(a_i\) depends only on its position) have to be estimated, then the regret over the social welfare \(R_{SW}^T(M)\) is \(\Theta(T)\) and therefore at every step (or repetition of the auction) a non–zero regret is accumulated. In addition, the authors provide necessary and, in some situations, sufficient conditions to have DSIC in restricted environments (i.e., higher slot higher click probability, separable CTRs in which only ads qualities need to be estimated), without presenting any bound over the regret (except for reporting an experimental evidence that the regret is \(\Omega(T^{2/3})\) when the CTRs are separable).

In [24], the authors study multi–slot auctions when the separable CTR model is adopted and the quality \(q_i\) considered unknown, designing exploration–exploitation mechanism based on MAB and studying the regret over the revenue of the auctioneer. Specifically, the obtained regret is \(\tilde{O}(T^{2/3}N^{1/3}K^{2/3})\). Moreover, they provide results over multi–slot auctions when the Cascade Model is adopted and \(q_i\) is unknown. The designed mechanism has a regret bound that scales as \(\tilde{O}(T^{2/3}K^{2/3}N)\).
CHAPTER 3

Models and Summary of the Results

In Sec. 2.3, we introduced two models used in SSAs. The Cascade Model, introducing an ad–dependent externality, better fits the user’s behaviour than the separable CTR model. But, even this model has been criticised, e.g. because it does not consider the way adjacent ads interacts among them, indeed the continuation probability depends only on a single ad an not on a couple of adjacent ads.

The goal of this chapter is to extend the Cascade Model, proposing new models that extend it along different ways, overcoming its limitations. In particular, in Fig. 3.1, we propose a graph representing the relations among models already proposed in literature and our new ones. In the first part of this chapter we will introduce the models, while in the second one we provide a summary of the results presented in the following chapters.

3.1 Models

When designing the new models extending the Cascade Model, we were driven by two main motivations:

- better characterize the behaviour of the user in order to improve the efficiency of SSAs;
Chapter 3. Models and Summary of the Results

Figure 3.1: Summary of computational results

Parameters: $\phi_t, \phi_t^m$ (c)

Mobile advertising (c)

Only ad-dependent (c)

Separable CTR (c)

Cascade Model (c)

Complete

Complete

Complete

Complete

$\phi_t, \phi_t^m$ (c)

$\phi_t, \phi_t^m$ (c)

$\phi_t, \phi_t^m$ (c)
• design model that fits for more general environments.

In particular, in relation to the second goal, the new models can be adopted in the position auction environment, i.e., auctions where a user, with a Markovian behaviour, physically moves among positions and not just moves his eyes like in SSAs. Anyway, except for a model specifically designed for a different application (Geo–location advertising), in order to ease the presentation and comprehension, we will introduce the other new models in the environment of SSAs, their extension to the case of position auction being straightforward.

All the models in Fig. 3.1 share most of the parameters of the Cascade Model. We recall the shared ones:

- $\mathcal{N} = \{1, \ldots, N\}$ is the set of indices of the ads $a_i$ ($i \in \mathcal{N}$). An additional fictitious ad is $a_\perp$. W.l.o.g. we assume each advertiser to have a single ad, so each advertiser $i$ can be identified with ad $a_i$;

- $\mathcal{K} = \{1, \ldots, K\}$ is the set of indices of the slots $s_m$ ($m \in \mathcal{K}$) ordered from the top to the bottom. We define $\mathcal{K}_c = \mathcal{K} \cup \{K + 1, \ldots, N\}$ (when $N > K$);

- $v_i \in V_i \subseteq \mathbb{R}^+$ is the value for advertiser $i$ when ad $a_i$ is clicked by a user, $v = (v_1, \ldots, v_K)$ is the value profile. We define $v_\perp = 0$;

- $\hat{v}_i \in V_i$ is the value reported by advertiser $i$, $\hat{v} = (\hat{v}_1, \ldots, \hat{v}_K)$ is the reported value profile;

- $\Theta$ is the set allocations of ads to slots, where each ad except $a_\perp$ can be allocated in at most one slot; we assign $a_\perp$ to $s_j$ to denote that no ad is displayed in $s_j$, and we assign $a_i$ to $s_\perp$ to denote that $a_i$ is not displayed.

The first dimension in which we extend the cascade model is the way the externalities are represented. This is the criterion that guided us in the definition of the graph in Fig. 3.1. An arrow, starting from a model and pointing to an other, expresses the fact that the source model generalises the sink one. The relation is transitive.

We start the analysis of the graph from the terminal nodes at the bottom left of it, i.e., the separable CTR model and the only ad–dependent model. These are the two simplest models proposed in literature and we described them in Sec. 2.3 in this chapter we recall that the main difference among the two models is that the first one (separable CTR model) considers only the position–dependent externalities, i.e., the only externalities modelled
are related to where the ads are allocated and not to which ads are allocated, while the second considers only the ad–dependent externalities.

It is easy to see that both these models are a specialisation of the Cascade Model, where both the position– and ad–dependent externalities influence the allocation. In particular, as described in Sec. 2.3, these two externalities are factorized in two groups of parameters: the position–dependent externalities represented by parameters $\lambda_m \in [0, 1], \forall m \in K$, and the ad–dependent externalities modelled by $\gamma_i \in [0, 1], \forall i \in N$.

### 3.1.1 Three new models: CFNE$_{aa}$, CFNE$_{sa}$, CFNE–$q_{i,m}$

In this thesis we design three new models extending the Cascade Model.

The first model we propose is named CFNE$_{aa}$ (Cascade Forward Negative Externalities). It preserves the factorization of position– and ad–dependent externalities, but it redefines the second ones as $\gamma_{i,j} \in [0, 1]$, where $a_j$ is the ad that is displayed in the slot just below the slot in which $a_i$ is displayed. Parameters $\gamma_{i,j}$ can be seen as the weights of a directed complete contextual graph $G = (N, E)$ where the direct edges $(i, j)$ represent the way an ad influences the other. $CTR_i(\theta)$ is defined as $CTR_i(\theta) = q_i \Lambda_{\pi(i;\theta)} \Gamma_i(\theta)$ where $\Lambda_m$ is defined as in the Cascade Model and $\Gamma_i(\theta) = \prod_{l=1}^{\pi(i;\theta)-1} \gamma_{\alpha(l;\theta),\alpha(l+1;\theta)}$. The fact that in the Cascade Model each ad influences the following in the same way, ignoring which is the following displayed ad, was considered a weakness of the model, being not realistic. Indeed, if we consider two car makers attracting the high end of market, their ads will have a strong externality, while the externality existing between two makers in a different price bracket is arguably much less strong. CFNE$_{aa}$ overcomes this limitation capturing the situation in which each ad can affect each other ad in a different way.

The second proposed model is named CFNE$_{sa}$. It extends the Cascade Model removing the factorization of ad and position–dependent externalities substituting the parameters $\lambda_m$ and $\gamma_i$ with the single position/ad–dependent parameters $\gamma_{m,j} \in [0, 1], m \in K$ and $j \in N$. $CTR_i(\theta)$ is defined as $CTR_i(\theta) = q_{m,i} \Gamma_i(\theta)$ where $\Gamma_i(\theta) = \prod_{m=1}^{\pi(i;\theta)-1} \gamma_{m,\alpha(m;\theta)}$. This models captures the situation in which an ad can affect the ads displayed below in a different way according to the position in which it is displayed.

Finally, we introduce the model named CFNE–$q_{i,m}$. Differently w.r.t. the CFNE$_{sa}$ model, where the position dependent externality is merged with the ad dependent parameter $\gamma_i$, in CFNE–$q_{i,m}$, we remove the parameter $\lambda_m$, merging the position–dependent component with the other ad–dependent parameter, the quality $q_i$. The new parameter is identified as
3.1. Models

$q_{m,i}$, the quality of ad $a_i$ when it is allocated in slot $s_m$. It is easy to imagine situations where the quality is a function of the slot. Think for example to applications related to position auctions where the user does not have just to click on the ad, like in SSA, but the slots are places distributed in an area and the user, after having received the ad, has to move to the shop that is advertised (e.g., in order to use coupons). The distance between the place where the user receives the ad and the place where the corresponding shop is located can influence the probability the user will go to the shop, i.e., the quality of the ad. We will discuss more in depth this aspect when we will introduce the Mobile Advertising model (bottom right of Fig. 3.1).

For all these three new models, CFNE$_{aa}$, CFNE$_{sa}$ and CFNE$_{q_im}$, we have identified a common ancestor that generalises them all. We called it CFNE. This model is characterised by having two parameters representing the ad/position–dependent externalities:

- $q_{i,m}$: the quality of ad $a_i$ when it is allocated in slot $s_m$;
- $\gamma_{m,i,j}$: the probability the user will move from slot $s_m$ to $s_{m+1}$ when $a_i$ is allocated in slot $s_m$ and $a_j$ in $s_{m+1}$.

With this model $CTR_i(\theta)$ is computed as $CTR_i(\theta) = q_{\pi(i;\theta),i} \Gamma_i(\theta)$ where $\Lambda_m$ is defined as in the Cascade Model and $\Gamma_i(\theta) = \prod_{l=1}^{\pi(i;\theta)-1} \gamma_{l,\alpha(l;\theta),\alpha(l+1;\theta)}$. In this dissertation we will focus mainly on the CFNE$_{aa}$ and CFNE$_{sa}$ models.

3.1.2 Two new dimensions: user memory and $a_\perp$

The second dimension we used to distinguish the models concerns the user memory. We denote by $c$ the number of ads displayed above $a_i$, from $s_{\pi(i;\theta)-1}$ to $s_{\pi(i;\theta)-c}$, that affect $CTR_i$. While in the Cascade Model, $c$ is equal to $K$, i.e., each allocated ad influences the CTR of all the ads allocated after it, here we consider also the cases where the memory of the user is limited and therefore $c \leq K$. This is supported by some experimental studies [32].

We show how the limited user memory affects the models CFNE$_{aa}$ and CFNE$_{sa}$. It is easy to extend the same reasoning to the other models.

We now identify the models explicitly expressing the length $c$ of the user memory: CFNE$_{aa}(c)$ and CFNE$_{sa}(c)$. In both the models the unique change resides in the definition of $\Gamma_i(\theta)$. Adopting the model CFNE$_{aa}(c)$ the $CTR_i(\theta)$ is still computed as $CTR_i(\theta) = q_i \Lambda_{\theta(a_i)} \Gamma_i(\theta)$, but $\Gamma_i(\theta)$ is now defined as $\Gamma_i(\theta) = \prod_{l=\pi(i;\theta)-c}^{\pi(i;\theta)-1} \gamma_{\alpha(l;\theta),\alpha(l+1;\theta)}$. In the second model $\Gamma_i(\theta)$ becomes $\Gamma_i(\theta) = \prod_{m=\pi(i;\theta)-c}^{\pi(i;\theta)-1} \gamma_{m,\alpha(m;\theta)}$. 

31
Chapter 3. Models and Summary of the Results

The last dimension we introduce concerns the special ad $a_\perp$. Also in this case we analyse in depth only the models $\text{CFNE}_{aa}(c)$ and $\text{CFNE}_{sa}(c)$. Therefore, we will distinguish models on the basis of the definition of $\gamma_{m,\perp}$ for $\text{CFNE}_{sa}(c)$ and $\gamma_{i,\perp}$ and $\gamma_{\perp,i}$ for $\text{CFNE}_{aa}(c)$. We identify the following models:

- $\text{CFNE}_{sa}(c)$–nr ($\text{CFNE}_{aa}(c)$–nr): $\gamma_{m,\perp} = 0$ ($\gamma_{i,\perp} = 0$ and $\gamma_{\perp,i} = 0 \forall i \in \mathcal{N}$). This model captures the situation in which leaving a slot empty between two allocated slots does not provide any advantage. This assumption is done in [22, 35];

- $\text{CFNE}_{sa}(c)$–r ($\text{CFNE}_{aa}(c)$–r): $\gamma_{m,\perp}$ ($\gamma_{i,\perp} = \gamma_{\perp,i} = 1 \forall i \in \mathcal{N} \cup \{\perp\}$). This model captures the situation in which the slots can be distributed in the page in different positions (aka slates) and, in order to raise the user’s attention, we can allocate, e.g., pictures that nullify the externality between the ad allocated before the picture and the ad allocated after the picture as demonstrated in wholepage optimization problems [39];

- $\text{CFNE}_{sa}(c)$–tr ($\text{CFNE}_{aa}(c)$–tr): $a_\perp$ is defined such that, once allocated at slot $s_j$, all the externalities due to the ads previously allocated in $\theta$ are nullified and thus ad $a_i$ is allocated at $s_{j+1}$ has $\Gamma_i(\theta) = 1$.

where nr stands for no reset, r for reset and tr for total reset.

3.1.3 The last new model: Mobile Advertising

The last model we have not analysed yet is the bottom right one, i.e., the Mobile Advertising (MA) model. The arrow connecting the $\text{CFNE}–q_{i,m}$ model to the MA one has a difference w.r.t. the others, it is a dotted line. This is due to the fact that the MA model has a strict relation with the $\text{CFNE}–q_{i,m}$ one, but it is not really a specialization of the latter. We designed and studied this model with a specific application in mind: mobile geo–location advertising [52]. In this environment mobile ads, i.e., ads sent to mobile devices, are targeted based on a user’s location (e.g., streets or squares). Mobile geo–location advertising has been identified as a key growth factor for the mobile market and a crucial ingredient for its success will be the development of effective economic mechanisms.

The idea is that mobile ads, such as coupons and ads in mobile apps, are shown sequentially over time while the user moves in an environment, e.g., a city or shopping centre. Furthermore, users are affected by the same ad in different ways depending on the location in which they receive the ads
3.1. Models

(e.g., if the shop is far from the location in which the ad is received, users are more likely to discard the ad), and the path followed so far can reveal information about the user’s intention (i.e., the user’s next visits). Thus, ad allocations are shown dynamically taking the user behaviour into account, unlike in SSAs, where the entire allocation is shown simultaneously. Notice that none of the online advertising models can be directly applied to mobile geo–location mainly because they do not take into account the future behaviour of the user.

We thus propose the first model for the mobile geo–location advertising problem and we will pair the allocation function with a pay–per–visit scheme, where an advertiser, differently from the previous models, does not pay when his ad is clicked, but pays only if a user actually visits the shop after having received the ad (based on geo–location or by redeeming a coupon). The two main aspects that we capture with the model are that the visit probability depends on the position where the user receives the ad and that the user suffers of the ad fatigue phenomenon [1], i.e., receiving an ad discounts the visit probability associated with the next ads. This creates sponsored–search like externalities, except that the visit probability depends only on the number of ads shown prior, and not on which ads are shown.

We now analyse the problem and the model in a more formal way beginning with the user mobility model. We represent a physical area, e.g., a city, as a graph $G = (T, E)$, e.g., Fig. 3.2(a), where $T$ is the set of the indices of vertices $t_l$, $l \in T$, and $E$ is the set of edges. Vertices are sub-areas, e.g., streets or squares, in which an ad can be sent to a user. A user will move over the graph following a path, denoted by $\psi \in \Psi$, and defined as a sequence of adjacent vertices. $\Psi$ is the set of all the possible paths.

![Figure 3.2: Running example: graph (a), paths with starting vertex $t_1$ (b), tree of paths (c).](image-url)
We denote the first or starting vertex of a path \( \psi \) by \( t_s(\psi) \), and partition paths by this starting position. To this end, we introduce \( \Psi_{t_l} \subseteq \Psi \) which denotes the set of paths \( \psi \) with \( t_s(\psi) = t_l \). For instance, Fig. 3.2(b) depicts 4 paths with starting vertex \( t_1 \). In addition, we associate each path \( \psi \) with a probability \( \omega_\psi \)—estimated, e.g., by means of machine learning tools [24]—that indicates how likely the given path will be followed by users, given the starting vertex of the user. Thus, \( \sum_{\psi \in \Psi_{t_l}} \omega_\psi = 1 \). Since the number of possible paths in \( \Psi_{t_l} \) can be arbitrarily large, we restrict \( \Psi_{t_l} \) to a finite (given) number of paths containing only the paths with the highest probability. We normalize the probabilities \( \omega_\psi \) accordingly.

Given a user’s actual starting vertex, we can build the tree of the paths the user could follow. Fig. 3.2(c) depicts an example of such a tree with starting vertex \( t_1 \). We denote by \( K = \{1, \ldots, K\} \) the indices of the nodes belonging to a tree, where each node \( s_m \) is associated with a single graph vertex \( t_l \), \( l \in T \), whereas each vertex \( t_l \) can be associated with multiple tree nodes \( s_m \), \( m \in K \). The nodes \( s_m \) are strictly related to the concept of slots in SSA. We define \( \sigma_m \) as the probability with which node \( s_m \) is visited by a user given a starting vertex. In particular, \( \sigma_m \) is equal to the sum of the \( \omega_\psi \) of the paths \( \psi \) sharing \( s_m \). Consider, for instance, the bold node \( s_3 \) in Fig. 3.2(c): since \( s_3 \) is contained in both the paths \( \psi_1 \) and \( \psi_2 \), \( \sigma_3 = \omega_{\psi_1} + \omega_{\psi_2} \).

We still denote with \( N = \{1, \ldots, N\} \) the set of indices of the ads \( a_i \), \( i \in N \) and as \( a_\perp \) the special ad corresponding to sending no ads. \( \theta \) still represents an outcome, i.e., an advertising plan (i.e., an allocation of ads to nodes in which the ads are sent to the users). \( \Theta \) is the set of all the advertising plans. We still have the functions \( \pi(i; \theta) \) and \( \alpha(m; \theta) \) with the unique difference that now \( s_m \) are the nodes and not the slots. As with the SSA, we constrain \( \theta \) to not allocate the same ad (except \( a_\perp \)) on different nodes belonging to the same path (the basic idea is that receiving the same ad multiple times does not affect the visit probability of the corresponding shop, but it just increases ad fatigue), while the same ad can be allocated on different paths. \( v_i \in V_i \) represents now the value obtained by advertiser \( i \) when the user visits his shop, instead of clicks on his ad. In this new context, the probability with which a user visits the shop \( a_i \) is given by \( \text{VTR}_i(\theta) \), and we can define the SW just substituting the word CTR with VTR: \( \text{SW}(\theta, \mathbf{v}) = \sum_{i \in N} \text{VTR}_i(\theta) v_i \).

It remains to define how the \( \text{VTR}_i \) is computed. We said that the VTR depends on the position where the user receives the ad and on the fact the user suffers of the ad fatigue phenomenon. The first aspect is modelled inheriting the quality defined in CFNE–\( q_{i,m} \): the parameter \( q_{i,m}, i \in N \) and
3.2 Summary of Results

$m \in \mathcal{K}$, represents the quality of ad $a_i$ when the user receives it in position $s_m$, i.e., the probability the user visits the shop corresponding to $a_i$ when he receives the ad in node $s_m$. The second aspect is represented through a parameter not depending on which ads are shown, but depending on the number of already shown ads. We thus define a function $\mu : \mathcal{K} \times \Theta \rightarrow \mathbb{N}$ returning the number of non–empty ads allocated to nodes that precede $s_m$ from the root node. Finally, we call $VTR_i(\theta) = \sum_{m \in \mathcal{K} : \alpha(m;\theta) = i} \sigma_m \Lambda_{\mu(m;\theta)} q_i, m$, where:

- $\sigma_m$ is the probability that node $s_m$ is visited;
- with abuse of notation, $\Lambda_{\mu(m;\theta)}$ is the aggregated continuation probability, capturing the ad fatigue phenomenon where the user’s attention decreases as more ads are received; we assume the user attention decreases as $\Lambda_{\mu(m;\theta)} = \prod_{l=1}^{\mu(m;\theta)} \lambda_l$ where $\lambda_l \in [0, 1] \forall l \in \{0, \ldots, K-1\}$, $\lambda_l \geq \lambda_{l+1}$, and $\Lambda_0 = 1$.

3.2 Summary of Results

In this section, we summarize the results provided in this thesis and we point out where each result is described.

Initially, we summarize our computational complexity and mechanism design results and subsequently the online learning results.

3.2.1 Computational Complexity and Mechanism Design

The dissertation on this topic is divided in three Chapters: 4, 5 and 6. In Fig. 3.1 we highlight in bold font our original contributions.

Chapter 4 is devoted to the study of the CFNE$_{sa}(c)$ and CFNE$_{aa}(c)$ models when $c$ is a constant and $c < K$. In Sec. 4.1 we formulate the CFNE$_{sa}(1)$ problem as an Integer Linear Program (ILP) and we prove that this ILP can be solved in polynomial time. The extension of this result to the case of $1 < c < K$ is straightforward. This result is also inherited by the specialization of the CFNE$_{sa}(c)$ model, i.e., the Cascade Model, the separable CTR and the only ad–dependent ones, when the user memory window is strictly less than $K$. Literature provided results on the separable CTR and only ad–dependent models, but only considering the case of $c = K$. Differently from CFNE$_{sa}(c)$, the CFNE$_{aa}(c)$–{nr,r} is APX–hard, see Sec. 4.2. We prove it through an approximation preserving reduction from the TSP(1, 2) problem. Given this result, we can also conclude that the more general CFNE(c)–{nr,r} model is APX–hard. For the
Chapter 3. Models and Summary of the Results

CFNE\textsubscript{aa}(c)–nr, in Sec. 4.3.1, we designed an algorithm based on Colour Coding, but given that the approximation guaranteed is not a constant, we cannot conclude the problem is APX–complete, while if a parameter is given (\(\gamma_{\text{min}}\)), this subset of instances is APX–complete, indeed, there exists an algorithm returning a constant approximation in the parameter \(\gamma_{\text{min}}\). For the other two classes of CFNE\textsubscript{aa}(c), CFNE\textsubscript{aa}(c)–{r,tr}, in Sec. 4.3.3, we showed that an approximation algorithm providing a \(\frac{1}{2}\)–approximation solution can be easily designed. Thus, CFNE\textsubscript{aa}(c)–r belongs to the APX–complete class and CFNE\textsubscript{aa}(c)–tr to APX. All the algorithms described can be implemented in a DSIC mechanism.

Chapter 5 is mainly devoted to the study of the computational complexity of the CFNE\textsubscript{aa}(K) models. We leave open the problem of assessing the computational complexity of CFNE\textsubscript{sa}(K) and Cascade Model. This will be a topic of our future studies. In Sec. 5.3 we give an approximation preserving reduction from the direct Longest Path problem to the CFNE\textsubscript{aa}(K)–nr and, while in Sec. 5.4 we design an involved approximation algorithm using several different ideas and sources of approximation; interestingly, its approximation guarantee matches the best known approximation guarantee for the direct Longest Path, thus we conclude that CFNE\textsubscript{aa}(K)–nr is poly–APX–complete. However, this algorithm is not monotone and thus it cannot be used in any truthful mechanism. Therefore, in Sec. 5.5 we design a monotone algorithm that can be used in truthful mechanisms, but that provides a weaker approximation guarantee. Moreover, given the hardness of the problem, in Sec. 5.2.1 we identify classes of tractable instances for which we provide an exact polynomial-time algorithm. For the models, CFNE\textsubscript{aa}(K)–{r,tr} we show that the problem becomes easier and turns out to be APX–complete. We first prove, in Sec. 5.6 the problem to be APX–hard, via an approximation preserving reduction from (a subclass of) ATSP (i.e., asymmetric version of TSP) and then, in Sec. 5.7, we observe the simple greedy algorithm defined in Chapter 4 returns a \(\frac{1}{2}\)–approximate solutions even in this case. We finally note how to use this algorithm to design a truthful mechanism.

Finally, in Chapter 6, we study the MA model mainly from an algorithmic and mechanism design point of view. We conjecture the problem belongs to the class NP–hard, but we have not proved it yet. Specifically, in this chapter we discuss two different problems: the case the tree is simply a single–path, in Sec. 6.1, and the case it is actually a tree, thus, the multi–path environment, in Sec. 6.2.
3.2. Summary of Results

3.2.2 SSA and MAB

The literature review provided in Sec. 2.3.2 left a lot of open questions on the Cascade Model. In this thesis, we provide answers to some of them.

We summarize in Tab. 3.1 the results known in literature and, in bold font, the original results provided in this dissertation.

In Chapter 7 we face the separable CTR model. Specifically, we first review the bound over the auctioneer’s regret provided in [24], where they assume that only the parameter $q_i$ is unknown to the auctioneer. In Sec. 7.1 we consider the regret over the SW and we prove that adopting the same algorithm of [24] we obtain a bound of $\tilde{\Theta}(T^{2/3})$. Then, in Sec. 7.2 we move to the opposite case where only $\Lambda_m, m \in K$, are unknown. Here, we focus on mechanisms that are DSIC in expectation, w.r.t. click realizations in Sec. 7.2.1 and w.r.t. the realizations of the random component of the mechanism in Sec. 7.2.2. In the first case we observe that we can obtain a mechanism with a regret (both on the revenue of the auctioneer and on the SW) with a bound of $0$, but the mechanism is WBB only in expectation. In the second scenario both the regrets are bounded by a constant $O(1)$ and the mechanism is IR and WBB \textit{a posteriori}. In Sec. 7.2.3 we then discuss the possibility of having no–regret DSIC mechanisms, but we provide a negative result. Obviously, this negative result extends to all the generalizations of the separable CTR model.

Finally, we close Chapter 7 studying the case where both $\Lambda_m$ and $q_i$ are unknown by the auctioneer. In Sec. 7.3 given the negative result of Sec. 7.2.3 we focus only on DSIC in expectation mechanisms, obtaining bounds of $\tilde{O}(T^{2/3})$ for both the kinds of regret.

In Chapter 8 we study the Cascade Model. We first review the result provided in [24] regarding the regret over the revenue of the auctioneer when only the parameters $q_i$ are unknown. In [24] the authors observed the existence of a gap between the dependence over $N$ and $K$ of the derived bound and the results in the numerical simulation. In this thesis we modify their proof filling the gap. Notice that the dependence on $T$ does not change. In Sec. 8.1 we provide a result over the SW regret, where the bound is still $\tilde{O}(T^{2/3})$.

Considering other situations of lack of information we obtain only negative results, i.e., a no–regret DSIC mechanism does not exist when we consider the only ad–dependent model with only parameters $\gamma_i$ unknown. In both the cases, the results can be extended to the more general models that can be identified going through the arrows, of the hierarchy depicted in Fig. 3.1, in their opposite direction.
### Table 3.1: Known results on regret bounds for SSA. We remark with bold font the results provided in this dissertation.

<table>
<thead>
<tr>
<th>slots</th>
<th>CTR model</th>
<th>unknown parameters</th>
<th>solution concept</th>
<th>regret over welfare ($R_{SW}^T$)</th>
<th>regret over revenue ($R_T^T$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>–</td>
<td>$q_i$</td>
<td>DSIC</td>
<td>$\widetilde{\Theta}(T^{2/3})$</td>
<td>$\widetilde{\Theta}(T^{2/3})$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DSIC in exp. (w.r.t. mech.)</td>
<td>$\widetilde{\Theta}(T^{1/2})$</td>
<td>$\widetilde{O}(T^{2/3})$</td>
</tr>
<tr>
<td>$&gt;1$</td>
<td>(unconstrained) $CTR_{i,m}$</td>
<td>$CTR_{i,m}$</td>
<td>DISC</td>
<td>$\Theta(T)$</td>
<td>unknown</td>
</tr>
<tr>
<td></td>
<td>CFNE$_{sa}(K)$</td>
<td>$q_i$</td>
<td>DISC</td>
<td>$\Theta(T^{2/3})$</td>
<td>$\Theta(T^{2/3})$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\gamma_{m,i}$</td>
<td>DISC</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
<tr>
<td></td>
<td>separable CTR model / Cascade Model</td>
<td>$\lambda_m$</td>
<td>DSIC</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IC in exp. (w.r.t. clicks)</td>
<td>$0$</td>
<td>$0$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DSIC in exp. (w.r.t. mech.)</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q_i, \lambda_m$</td>
<td>DSIC</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
<tr>
<td>only ad–dependent model</td>
<td></td>
<td></td>
<td>DSIC in exp.</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\gamma_i$</td>
<td>DSIC</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q_i, \gamma_i$</td>
<td>DSIC</td>
<td>$\Theta(T)$</td>
<td>$\Theta(T)$</td>
</tr>
</tbody>
</table>

Chapter 3. Models and Summary of the Results
In this chapter we study the models CFNE\textsubscript{sa}(c) and CFNE\textsubscript{aa}(c) when $c$ is a constant and is strictly lower than $K$.

Specifically, in Sec. \ref{sec:4.1}, we initially show that the problem of finding the optimal allocation for CFNE\textsubscript{sa}(c) can be solved in polynomial time. Then, in Sec \ref{sec:4.2} we focus on the CFNE\textsubscript{aa}(c)--\{nr, r\} problems, proving that even when $c = 1$ the problem is APX–hard. In particular, in the case of CFNE\textsubscript{aa}(c)--nr we design an approximation algorithm, in Sec. \ref{sec:4.3.1} guaranteeing a ratio of $\log(N) / 2\min\{N,K\}$, and, in Sec. \ref{sec:4.3.2} we show that the problem is APX–complete for a specific subset of instances.

We conclude the chapter, proving, in Sec. \ref{sec:4.3.3} that CFNE\textsubscript{aa}(c)--r is APX–complete and CFNE\textsubscript{aa}(c)--tr is APX, designing a $\frac{1}{2}$–approximation algorithm.

### 4.1 CFNE\textsubscript{sa}(c) is in $P$ for constant $c$

We next show that CFNE\textsubscript{sa}(c) is in $P$, whenever $c = O(1)$. Our discussion focuses on CFNE\textsubscript{sa}(1)--nr to simplify the notation. The extensions to $c > 1$ and reset do not need the introduction of new ideas, but only a more
cumbersome notation. CFNE\textsubscript{sa}(1)--nr can be formulated as follows:

\[
\begin{align*}
\max & \sum_{m=2}^{K} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, j \neq i} \gamma_{m-1,j} q_i v_i x_{j,m,i} + \sum_{i \in \mathcal{N}} x_{1,i} q_i v_i \\
\text{s.t.} & \sum_{m=2}^{K} \sum_{j \in \mathcal{N}, j \neq i} x_{j,m,i} + x_{1,i} \leq 1 \quad \forall i \in \mathcal{N} \tag{4.1} \\
& x_{1,i} = \sum_{j \in \mathcal{N}, j \neq i} x_{i,2,j} \quad \forall i \in \mathcal{N} \tag{4.2} \\
& \sum_{j \in \mathcal{N}, j \neq i} x_{j,m,i} = \sum_{j \in \mathcal{N}, j \neq i} x_{i,m+1,j} \quad \forall i \in \mathcal{N}, 2 \leq m < K \tag{4.3} \\
& \sum_{i \in \mathcal{N}} x_{1,i} = 1 \tag{4.4} \\
& \sum_{j \in \mathcal{N}} \sum_{i \in \mathcal{N}, i \neq j} x_{j,m,i} = 1 \quad \forall m \in \mathcal{K} \setminus \{1\} \tag{4.5} \\
\end{align*}
\]

where \(x_{j,m,i} = 1\) iff \(a_i\) is allocated in slot \(s_m\), \(m > 1\), and \(a_j\) is allocated in slot \(s_{m-1}\); \(x_{1,i} = 1\) iff \(a_i\) is allocated in \(s_1\). Constraints (4.1) impose that each ad \(a_j\) can be allocated in, at most, one slot, while, similarly, Constraints (4.4, 4.5) hold when in each slot exactly one ad is displayed. Finally, Constraints (4.2) impose that if \(a_i\) is allocated in slot \(s_1\) \((x_{1,i} = 1)\), then there exists a unique ad \(a_{j}, j \neq i\), allocated in \(s_2\), s.t. the variable \(x_{i,2,j}\) is set to 1, where \(x_{i,2,j} = 1\) iff ads \(a_i\) and \(a_j\) are allocated in the first and second slot, respectively. Constraints (4.3) are the generalization of Constraints (4.2) to the generic slots \(s_m\) and \(s_{m+1}\) with \(m > 1\).

The next proposition proves that we can solve the above ILP in polynomial-time, despite its similarities with the well–known \(NP\)-hard 3D–assignment problem.

\textbf{Proposition 2.} \textit{The continuous relaxation of the ILP above always admits integral optimal solutions.}

\textbf{Proof.} We show that, if there is an optimal fractional solution \(x\), then there are at least two feasible integral solutions with the same value of social welfare.

Specifically, we prove that \(x\) is equivalent to a probability distribution over integral allocations \(\theta = (a_1, \ldots, a_K)\). The probability \(\mathbb{P}(\theta)\) given to \(\theta\) is:
4.1. CFNE$_{sa}(c)$ is in $P$ for constant $c$

\[ \mathbb{P}(\theta) = \prod_{i=1}^{K} \mathbb{P} \left( \pi(i; \theta) = i \bigg| \bigwedge_{j<i} \pi(j; \theta) = j \right) = x_{1,1} \prod_{l=2}^{K} \frac{x_{l-1,l,l}}{\sum_{m \geq l} x_{l-1,l,l,m}}. \]

In order to show that $\mathbb{P}(\theta)$ is actually a probability distribution over allocations, we show that $\sum_{\theta \in \Theta} \mathbb{P}(\theta) = 1$.

The proof is recursive. Let $\Theta'$ be the set of allocations $\theta$ with the same $K - 1$ first ads. The allocations in $\Theta'$ differ only for the ad allocated to $s_K$. To fix the notation, for $\theta \in \Theta'$ let $\alpha(l; \theta) = l$, for $l < K$. We have:

\[
\sum_{\theta \in \Theta'} \mathbb{P}(\theta) = x_{1,1} \prod_{2 \leq l \leq K-1} \left( \frac{x_{l-1,l,l}}{\sum_{m \geq l} x_{l-1,l,l,m}} \right) \sum_{h \geq K} \frac{x_{K-1,K,h}}{\sum_{m \geq K} x_{K-1,K,m}}
\]

\[
= x_{1,1} \prod_{2 \leq l \leq K-1} \left( \frac{x_{l-1,l,l}}{\sum_{m \geq l} x_{l-1,l,l,m}} \right) \sum_{h \geq K} x_{K-1, K, h} \sum_{m \geq K} x_{K-1, K, m}
\]

\[
= x_{1,1} \prod_{2 \leq l \leq K-1} \left( \frac{x_{l-1,l,l}}{\sum_{m \geq l} x_{l-1,l,l,m}} \right).
\]

We can now apply the same reasoning in order to find the probability of the set of allocations that share the same $K - 2$ ads. Iteratively applying this procedure we will arrive to compute the probability of the set of allocations that share at least the first ad. This probability will be $x_{1,i}$ if the first allocated ad is $a_i$. Thus, summing the probabilities of these sets we can conclude that $\sum_{\theta \in \Theta} \mathbb{P}(\theta) = \sum_{i \in N} x_{1,i} = 1$.

Since a fractional optimal $x$ is equivalent to a probability distribution over a number of feasible integral allocations, we have that the value returned by $x$ is $\sum_{\theta \in \Theta} SW(\theta, v) \mathbb{P}(\theta)$. Therefore, if $x$ is not integer, it must be that all the allocations $\theta$ with $\mathbb{P}(\theta) > 0$ have the same $SW(\theta, v)$, otherwise $x$ would be not optimal, being possible to obtain more by selecting the allocation $\theta'$ with the largest $SW$. Thus, if there is a non–integer optimal solution $x$, then there is at least an integer optimal solution.

Hence, given that the solution can be seen as an affine combination of allocations, we can conclude that, if the relaxed problem is solved and the optimal solution is fractional, there is always an integer solution that provides the same $SW$ and that can be reconstructed in polynomial time.

\[ \square \]
To solve the problem when \( c > 1 \), we just need to modify the ILP and allow each variable \( x \) to depend on \( c + 2 \) indices to take into account the at most \( c \) indices of all the ads that precede an ad. We introduce some preliminary definition and then the formulation, but we will not go into the details of it.

For any fixed \( c \) we need to compute \( \sum_{\ell=2}^{c+1} P(N, \ell) \) coefficients \( \gamma_{i_1, \ldots, m, i_{c+1}} \), where \( P(N, \ell) = \frac{N!}{(N-\ell)!} \) is the number of permutations of \( c \) elements extracted from a set of \( N \), and \( \ell \in \{2, \ldots, c+1\} \).

\[
\gamma_{i_1, \ldots, \ell, i_{\ell}} = \prod_{k=1}^{\ell-1} \gamma_{k, i_k}, \quad \forall \{i_1, \ldots, i_\ell\} \subseteq \mathcal{N}, \\
\ell \in \{2, \ldots, c\}
\]

\[
\gamma_{i_1, \ldots, m, i_{c+1}} = \prod_{k=1}^{c} \gamma_{m-c+k-1, i_k}, \quad \forall \{i_1, \ldots, i_{c+1}\} \subseteq \mathcal{N}, \\
m \in \{c+1, \ldots, K\}
\]

We define the following variables:

- \( x_{1,i_1} \) for all \( i_1 \in \mathcal{N} \). If \( x_{1,i_1} = 1 \) then \( \pi(i_1; \theta) = 1 \).

- \( x_{i_1,2,i_2} \) for all \( i_1, i_2 \in \mathcal{N} \) such that \( i_1 \neq i_2 \). If \( x_{i_1,2,i_2} = 1 \) then \( \theta(a_{i_1}) = s_1 \) and \( \pi(i_2; \theta) = 2 \).

- \( \ldots \)

- \( x_{i_1,\ldots,c,i_c} \) for all distinct \( i_1, \ldots, i_c \in \mathcal{N} \). If \( x_{i_1,\ldots,i_c} = 1 \) then \( \pi(i_1; \theta) = 1, \pi(i_2; \theta) = 2, \ldots, \pi(i_c; \theta) = c \).

- \( x_{i_1,\ldots,i_c,m,i_{c+1}} \) for all distinct \( i_1, \ldots, i_{c+1} \in \mathcal{N} \) and \( m \in \{c+1, \ldots, K\} \).
  - If \( x_{i_1,\ldots,i_c,m,i_{c+1}} = 1 \) then \( \theta(a_{i_1}) = s_{m-c}, \pi(i_2; \theta) = m-c+1, \ldots, \pi(i_{c+1}; \theta) = m \).

The problem can be solved via the following ILP, where for the sake of notation, we will assume that all \( \{i_1, \ldots, i_{c+1}\} \) are pairwise distinct.
4.1. CFNE_{sa}(c) is in P for constant c

\[
\max \left\{ \sum_{i_1 \in \mathcal{N}} q_{i_1} v_{i_1} x_{1,i_1} + \sum_{i_2 \in \mathcal{N}} \gamma_{i_1,2,i_2} x_{1,2,i_2} + \cdots + \sum_{i_K \in \mathcal{N}} \gamma_{i_1,...,c,i_c} q_{i_c} v_{i_c} x_{1,...,c,i_c} + \sum_{m=c+1}^{K} \sum_{i_1 \in \mathcal{N}, i_{c+1} \in \mathcal{N}} \gamma_{i_1,...,m,i_{c+1}} v_{i_{c+1}} x_{1,...,i_{c+1}} \right\}
\]

\[
x_{1,i_j} + \sum_{i_1 \in \mathcal{N}} x_{1,i_1,i_j} + \cdots + \sum_{i_c \in \mathcal{N}} x_{1,...,c,i_j} + \sum_{m=c+1}^{K} x_{1,...,i_c,m,i_j} \leq 1 \quad \forall i_j \in \mathcal{N}
\]

\[
x_{i_1,...,\ell,i_\ell} = \sum_{i_{\ell+1} \in \mathcal{N}} x_{i_1,...,i_\ell+1,i_{\ell+1}} \quad \forall \ell \in \{1,\ldots,c\}, \quad \forall i_1,\ldots,i_\ell \in \mathcal{N}
\]

\[
\sum_{i_1 \in \mathcal{N}} x_{i_1,...,i_c,m,i_{c+1}} = \sum_{i_j \in \mathcal{N}} x_{i_2,...,i_{c+1},m+1,i_j} \quad \forall m \in \{c+1,\ldots,K-1\}, \quad \forall i_2,\ldots,i_{c+1} \in \mathcal{N}
\]

\[
\sum_{i_1 \in \mathcal{N}, \ldots, i_\ell \in \mathcal{N}} x_{i_1,...,\ell,i_\ell} = 1 \quad \forall \ell \in \{1,\ldots,c\}
\]

\[
\sum_{i_1 \in \mathcal{N}, \ldots, i_c \in \mathcal{N}, i_{c+1} \in \mathcal{N}} x_{i_1,...,c,i_{c+1}} = 1 \quad \forall m \in \{c+1,\ldots,K\}
\]

\[
x_{i_1,...,\ell,i_\ell} \in \{0,1\} \quad \forall \ell \in \{1,\ldots,c\}, \quad \forall i_1,\ldots,i_\ell \in \mathcal{N}
\]

\[
x_{i_1,...,c,i_{c+1}} \in \{0,1\} \quad \forall m \in \{c+1,\ldots,K\}, \quad \forall i_1,\ldots,i_{c+1} \in \mathcal{N}
\]

We can also extend the first ILP formulation in order to represent the different reset models. This just requires the introduction of \(K\) additional variables for \(a_{\perp}\) to be visualized in each slot (together with some constraints to fix each variable for \(a_{\perp}\) to a slot).

Since it is well known, by LP theory, that the ellipsoid algorithm can be forced (in polynomial–time) to output an integral optimal solution (if any), Proposition 2 yields the following.

**Theorem 6.** For \(c = O(1)\), there is a polynomial-time optimal algorithm for CFNE_{sa}(c).
Mechanism design also becomes an easy problem for CFNE\(_{sa}(c)\), \(c = O(1)\), since the optimal algorithm can be used to obtain a truthful VCG mechanism.

### 4.2 CFNE\(_{aa}(1)\)–\{nr, r\} is APX–hard

In this section we prove that CFNE\(_{aa}(c)\) is APX–hard even in the easiest case when \(c = 1\). We conduct the proof by a reduction from the \(TSP(1, 2)\) problem, known to be APX–hard \cite{47}.

An instance of \(TSP(1, 2)\) consists of a complete weighted graph \(H = (\mathcal{V}, \mathcal{V}^2, w)\) where \(\mathcal{V} = \{1, \ldots, V\}\) is the set of \(V\) vertices of \(H\) and \(w : \mathcal{V}^2 \rightarrow \{1, 2\}\) is the weight function defined on the edges of \(H\), which can take only values in \{1, 2\}. A solution to the \(TSP(1, 2)\) problem is a Hamiltonian tour of minimum cumulative weight.

**Definition 6. (Reduction from \(TSP(1, 2)\) to CFNE\(_{aa}(1)\)–\{nr, r\}).** Given an instance \(H\) of \(TSP(1, 2)\), for each vertex \(i \in \{1, \ldots, V - 1\}\), we construct an ad \(a_i\), whereas for vertex \(V\) we construct two ads: \(a_V\) and \(a_{V+1}\). Thus, we obtain a set \(\mathcal{N} = \mathcal{V} \cup \{V + 1\}\) comprising \(N = V + 1\) ads. For each ad \(a_i\) we set \(q_i = 1\) and \(v_i = 1\). For each \(m \in \mathcal{K}\), we set \(\Lambda_m = 1\). Parameters \(\gamma_{i,j}\) are defined as in (4.6). The instance of CFNE\(_{aa}(1)\)–\{nr, r\} has \(K = V + 1\) slots.

\[
\gamma_{i,j} = \begin{cases} 
1 & \text{if } i \leq V, j < V, i \neq j \text{ and } w_{i,j} = 1; \\
\beta & \text{if } i \leq V, j < V, i \neq j \text{ and } w_{i,j} = 2; \\
1 & \text{if } i < V \text{ and } j = V + 1 \text{ and } w_{i,V} = 1; \\
\beta & \text{if } i < V \text{ and } j = V + 1 \text{ and } w_{i,V} = 2; \\
0 & \text{(CFNE\(_{aa}(1)\)–nr) if } j \in \mathcal{N} \text{ and } a_i = a_{\perp}; \\
0 & \text{(CFNE\(_{aa}(1)\)–nr) if } i \in \mathcal{N} \text{ and } a_j = a_{\perp}; \\
1 & \text{(CFNE\(_{aa}(1)\)–r) if } j \in \mathcal{N} \text{ and } a_i = a_{\perp}; \\
1 & \text{(CFNE\(_{aa}(1)\)–r) if } i \in \mathcal{N} \text{ and } a_j = a_{\perp}; \\
0 & \text{otherwise.}
\end{cases} \tag{4.6}
\]

To complete the reduction, we need to show that we can map an \(\alpha\)–approximate solution of CFNE\(_{aa}(1)\)–\{nr, r\} back to \(TSP(1, 2)\). To accomplish this, we will restrict ourselves to allocations \(\theta\) for the CFNE\(_{aa}(1)\)–\{nr, r\} problem that are well–formed, meaning that they are: (i) \(a_{\perp}\)–free (i.e., \(\theta\) does not contain any fictitious ad) and (ii) have ad \(a_V\) in the first slot and
a_{V+1} in the last slot. Given a well–formed $\alpha$–approximate allocation $\theta_\alpha = (a_V, a_1, \ldots, a_K, a_{V+1})$ we can easily obtain a $\beta$–approximate tour $\tau_\beta = (t_v, t_1, \ldots, t_K)$ by simply traversing the nodes in the order in which the corresponding ads are listed in $\theta_\alpha$.

Before proving our main result of this section, we next prove that restricting to well–formed allocations comes at no extra approximation cost, as given a non–well–formed allocation we can always obtain in polynomial time a well–formed allocation with a non–smaller social welfare value.

**Proposition 3.** Given a non–well–formed allocation $\theta_{NW}$ for a CFNE$_{aa}(1)$–{nr, r} instance obtained by the reduction of Definition 6, it is possible to construct a well formed allocation $\theta_W$ s.t. $SW(\theta_W, v) \geq SW(\theta_{NW}, v)$, provided that $\beta > 3/4$.

**Proof.** We exhibit a 2–step procedure that given in input a non–well–formed allocation computes in polynomial time a well–formed allocation while not decreasing the social welfare. In Step 1, the procedure removes all ads $a_\perp$, thus obtaining an $a_\perp$–free allocation $\theta_F$, whereas in Step 2, starting from $\theta_F$ it computes an allocation s.t. ads $a_V$ and $a_{V+1}$ are allocated in the first and last position, respectively.

**Step 1: removing $a_\perp$.** Suppose that $\theta_{NW}$ allocates $a_\perp$ (or otherwise we can skip to Step 2). We show that we obtain a better allocation without $a_\perp$. We conduct the prove for model CFNE$_{aa}(1)$–r, and highlight that it holds a fortiori also for CFNE$_{aa}(1)$–nr. We enumerate the possible situations, analyzing in depth just one of them (the others can be proved similarly).

1. Consider allocations $\theta_{NW} = (a_{i_1}, \ldots, a_{i_j}, a_{V+1}, a_\perp, a_V, a_{i_{j+1}}, \ldots, a_{i_{V-3}}, a_{i_{V-2}})$ and $\theta'$ defined as $\theta_{NW}$ but substituting $a_\perp$ with the unallocated ad $a_z$, $z < V$. We have $SW(\theta_{NW}, v) = \sum_{h=1}^7 \phi_h$ where $\phi_1 = 1$ ($\phi_1$ is the contribution of $a_{i_1}$ only), $\phi_2 = \sum_{l=2}^j \gamma_{i_{l-1}, i_l}$, $\phi_3 = \gamma_{ij, V+1}$, $\phi_4 = 0$ (the contribution of $a_\perp$), $\phi_5 = \gamma_{\perp, V} = 1$, $\phi_6 = \gamma_{V, i_{j+1}}$ and $\phi_7 = \sum_{l=j+2}^{V-2} \gamma_{i_{l-1}, i_l}$. Similarly, $SW(\theta', v) = \sum_{h=1}^7 \phi'_h$ where $\phi'_1 = 1$ (the contribution of $a_V$), $\phi'_2 = \gamma_{V, i_1}$, $\phi'_3 = \gamma_{i_{j-1}, i_j}$, $\phi'_4 = \gamma_{i_{j-2}, i_{j-1}}$, $\phi'_5 = \gamma_{z, i_{j+1}}$, $\phi'_6 = \sum_{l=j+2}^{V-2} \gamma_{i_{l-1}, i_l} = \phi_5$, $\phi'_7 = \gamma_{V-2, V+1}$. Since $\phi_2, \phi'_2, \phi_5, \phi'_5, \phi_7, \phi'_7 \geq \beta$, when $\beta > \frac{3}{4}$, we have $SW(\theta', v) = \sum_{h=1}^7 \phi'_h \geq 4\beta + \phi_2 + \phi_5 + 1 > 3 + \phi_2 + \phi_5 + 1 \geq \sum_{h=1}^7 \phi_h = SW(\theta_{NW}, v)$.

2. Consider allocation $\theta_{NW} = (\ldots, a_i, a_\perp, a_\perp, a_j, \ldots)$ where $i, j \in N \cup \{\perp\}$. It can be easily observed that by substituting the first $a_\perp$ with a non–allocated ad $a_z$ where $z \in N$, thus obtaining $\theta' = (\ldots, a_i, a_z, a_\perp, a_j, \ldots)$, we have $SW(\theta', v) \geq SW(\theta_{NW}, v)$ for $\beta \geq 0$. 

45
3. Consider allocation $\theta_{NW} = (\ldots, a_i, a_\perp, a_j, \ldots)$ where $i, j < V$. If $\theta' = (\ldots, a_i, a_z, a_j, \ldots)$ with $z < V$ then $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (a_V, \ldots, a_i, a_j, \ldots)$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (\ldots, a_i, a_j, \ldots, a_{V+1})$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$.

4. Consider allocation $\theta_{NW} = (\ldots, a_V, a_\perp, a_j, \ldots)$, with $j, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (\ldots, a_V, a_z, a_j, \ldots)$, with $i, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$. If $\theta' = (a_V, \ldots, a_i, \ldots, a_{V+1})$, where $i < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

5. Consider allocation $\theta_{NW} = (\ldots, a_{V+1}, a_\perp, a_j, \ldots)$, with $j, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$. If $\theta' = (\ldots, a_z, a_j, \ldots, a_{V+1})$, with $j, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

6. Consider allocation $\theta_{NW} = (\ldots, a_{V+1}, a_\perp, a_j, \ldots)$, with $i, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$. If $\theta' = (\ldots, a_i, a_z, \ldots, a_{V+1})$, with $i, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

7. Consider allocation $\theta_{NW} = (\ldots, a_i, a_\perp, a_{V+1}, \ldots)$, with $i, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (\ldots, a_i, a_z, \ldots, a_{V+1})$, with $i < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

8. Consider allocation $\theta_{NW} = (\ldots, a_V, a_\perp, a_{V+1}, \ldots)$, with $i, z < V$, $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$.

9. Consider allocations $\theta_{NW} = (a_i, a_\perp, a_j, \ldots)$ or $\theta_{NW} = (\ldots, a_i, a_\perp, a_j)$. When $\beta > \frac{3}{4}$, it is always possible to find an allocation $\theta'$ where $a_\perp$ has been removed and $SW(\theta', v) > SW(\theta_{NW}, v)$.

Step 1 is iterated until all $a_\perp$ ads are removed and an $a_\perp$–free allocation $\theta_F$ is obtained.

**Step 2:** moving $a_V$ and $a_{V+1}$. We are given in input an $a_\perp$–free allocation $\theta_F$. Let us suppose that ad $a_V$ and $a_{V+1}$ are not allocated in the first and last slots, respectively, as otherwise $\theta_F$ would already be well–formed. We enumerate all the possible situations and we provide details only for one of them:

1. For notational convenience, we rename the ads s.t. $\theta_F = (a_1, a_2, \ldots, a_i, a_{V+1}, a_{i+1}, \ldots, a_j, a_V, a_{j+1}, \ldots, a_{V-1})$. We show that in this case
moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. $SW(\theta_F, v)$ can be written as $SW(\theta_F, v) = \sum_{h=1}^{8} \phi_h$ where $\phi_1 = 1$ ($\phi_1$ is the contribution of $a_1$ only), $\phi_2 = \sum_{i=2}^{\gamma - 1, \ell} \phi_3 = \gamma_i, V+1, \phi_4 = \gamma_{V+1, i+1} = 0$, $\phi_5 = \sum_{l=i+2}^{V-1} \gamma_l - 1, \ell$, $\phi_6 = \gamma_j, V = 0$, $\phi_7 = \gamma_{V, j+1}$ and $\phi_8 = \sum_{l=j+2}^{V-1} \gamma_l - 1, \ell$. Let $\theta_W$ be the well-formed allocation obtained from $\theta_F$ by moving $a_V$ to the first slot and $a_{V+1}$ to the last slot, i.e., $\theta_W = (a_V, a_1, \ldots, a_i, a_{i+1}, \ldots, a_j, a_{j+1}, \ldots, a_{V-1}, a_{V+1})$. We can write $SW(\theta_W, v) = \sum_{h=1}^{8} \phi_h'$ where $\phi_1' = \gamma_{V, 1}$, $\phi_2' = \sum_{i=2}^{1} \gamma_{1-1, l} = \phi_2$, $\phi_3' = \gamma_{V-1, V+1}$, $\phi_4' = \gamma_i, i+1$, $\phi_5' = \sum_{l=i+2}^{V-1} \gamma_l - 1, l = \phi_5$, $\phi_6' = 1$ (which accounts for the contribution of ad $a_V$), $\phi_7' = \gamma_{j, j+1}$ and $\phi_8' = \sum_{l=j+2}^{V-1} \gamma_l - 1, l = \phi_8$. Notice that $\phi_1', \phi_3', \phi_4', \phi_7' \geq \beta$. Even in the case in which $SW(\theta_F, v)$ has the highest possible value, (i.e., when $\phi_1, \phi_3, \phi_7 = 1$), we have $SW(\theta_W, v) > SW(\theta_F, v)$ if $\beta > \frac{1}{2}$. Indeed: $\sum_{i=1}^{8} \phi_i' > \sum_{i=1}^{8} \phi_i$ implies $\beta + \beta + \beta + 1 + \beta > 1 + 1 + 1$, i.e., $\beta > \frac{1}{2}$.

2. For $\theta_F = (\ldots, a_V, \ldots, a_{V+1}, \ldots)$, $\theta_F = (a_V, \ldots, a_{V+1}, \ldots)$, $\theta_F = (\ldots, a_V, a_{V+1})$ or $\theta_F = (a_{V+1}, \ldots, a_V)$, $SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{1}{2}$;

3. For $\theta_F = (a_{V+1}, \ldots, a_V, \ldots)$, $\theta_F = (\ldots, a_{V+1}, \ldots, a_V)$, $\theta_F = (\ldots, a_V, a_{V+1}, \ldots)$, $SW(\theta_W, v) > SW(\theta_1, v)$ when $\beta > \frac{2}{3}$;

4. For $\theta_F = (\ldots, a_V, \ldots, a_{V+1})$, $\theta_F = (\ldots, a_{V+1}, a_V)$, $\theta_F = (a_V, a_{V+1}, \ldots)$, $\theta_F = (a_{V+1}, a_V, \ldots)$, $SW(\theta_W, v) > SW(\theta_F, v)$ if $\beta > 0$.

All the previous cases hold given that $\beta > \frac{3}{4}$. Thus moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. \qed

We are now ready to prove our APX-hardness result.

**Theorem 7.** $CFNE_{aa}(1)–\{nr, r\}$ is APX-hard.

**Proof.** With a slight abuse of notation, we denote by $SW(\tau)$ the value of the tour $\tau$ in the $TSP(1, 2)$ problem, while $SW(\theta, v)$ is the social welfare of $\theta$ in $CFNE_{aa}(1)$. Since by Proposition \[5\] we can now assume that $\theta$ is well-formed, it is easy to check that an optimal solution of the $TSP(1, 2)$ problem (denoted by $\tau^*$) corresponds to an optimal solution of the $CFNE_{aa}(1)$ problem (denoted by $\theta^*$) where the ads are allocated into the slots in the same order as the tour $\tau^*$ starting from $a_V$; $a_{V+1}$ is the last ad and no $a_\perp$ is present. For the sake of presentation, suppose the vertices of the graph and the ads are renamed s.t. the optimal solution $\theta^*$ of
Chapter 4. CFNE_{sa}(c) and CFNE_{aa}(c) with $c < K$

CFNE_{aa}(1) is $\theta^* = (a_V, a_1, a_2, \ldots, a_{V-1}, a_{V+1})$. Let $\alpha$ be the number of edges of weight 2 in $\tau^*$. Then, there are $\alpha$ couples $(a_i, a_j)$ of consecutive ads with $\gamma_{i,j} = \beta$. Thus, $SW(\tau^*) = (N - \alpha) \cdot 1 + \alpha \cdot 2 = N + \alpha$ and $SW(\theta^*, v) = (N + 1 - \alpha) \cdot 1 + \alpha \cdot \beta$.

Let us assume we have a well–formed solution $\theta'$ s.t. $SW(\theta', v) \geq (1 - \epsilon)SW(\theta^*, v)$.

Given that $\theta'$ is an $(1 - \epsilon)$–approximation of $\theta^*$, $\alpha'$ must be s.t. the following condition holds: $SW(\theta', v) = N + 1 - \alpha' + \alpha' \beta \geq (1 - \epsilon)(N + 1 - \alpha + \alpha \beta) = (1 - \epsilon)SW(\theta^*, v)$, from which we obtain the following upper bound on $\alpha'$:

$$\alpha' \leq (1 - \epsilon)\alpha + \frac{\epsilon(N + 1)}{1 - \beta} \quad (4.7)$$

From $\theta' = (a_V, a_1, \ldots, a_{V-1}, a_{V+1})$ we can produce a solution for TSP(1, 2) with the following tour $\tau' = (a_V, a_1, \ldots, a_{V-1})$; $\tau'$ will contain $\alpha'$ edges with weight 2 and then $SW(\tau') = N + \alpha'$. Thus, we can now show that $\tau'$ is a $(1 + \epsilon/(1 - \beta))$–approximation of $\tau^*$:

$$SW(\tau') = N + \alpha' \leq N + (1 - \epsilon)\alpha + \frac{\epsilon(N + 1)}{1 - \beta} \leq N + \alpha + \frac{\epsilon(N + \alpha)}{1 - \beta} \leq \left(1 + \frac{\epsilon}{1 - \beta}\right)(N + \alpha) = \left(1 + \frac{\epsilon}{1 - \beta}\right)SW(\tau^*)$$

where the first inequality follows from (4.8). Given that TSP(1, 2) is known to be APX–hard, $\epsilon$ cannot go arbitrarily close to zero and CFNE_{aa}(1)–r is then APX–hard.

4.3 Approximation algorithms for CFNE_{aa}(c)

4.3.1 Approximating CFNE_{aa}(c)–nr

The Color Coding method can be applied to CFNE_{aa}(c)–nr to design an exponential–time algorithm finding the optimal solution and a simple modification of such algorithm returns a $\frac{\log(N)}{2\min\{N,K\}}$ approximation in polynomial time.
The basic idea behind the Color Coding method is to define a set of colors \( C \), in our case \(|C| = K\), to randomly (with uniform probability) assign a color to an ad, to find the best allocation under the constraint that each ad in the allocation must have a different color, and to repeat the procedure a number of times returning the best found allocation. We denote by \( S \subseteq C \) a subset of colors and by \( \text{col}(a_i) \) a function returning the color assigned to \( a_i \). Given a coloring \( \text{col} \), the best allocation under the constraint that each ad in the allocation must have a different color is found by dynamic programming. For \(|S| > c\), \( W(S, \langle a_{h_0}, \ldots, a_{h_c} \rangle) \) contains the value of the best allocation with colors in \( S \) in which the last \( c + 1 \) ads are \( a_{h_0}, \ldots, a_{h_c} \) from top to bottom. (The definition naturally extends for \(|S| \leq c\).) Starting from \( W(\emptyset, \langle \rangle) = 0 \), we can compute \( W \) recursively. E.g., for \(|S| > c\), \( W(S \cup \{ \text{col}(a_{h_c}) \}, \langle a_{h_0}, \ldots, a_{h_c} \rangle) = \Lambda_{|S|+1}h_{h_c} \prod_{i=0}^{c-1} \gamma_{h_i,h_{i+1}} + \max_a W(S, \langle a, a_{h_0}, \ldots, a_{h_{c-1}} \rangle) \) if \( \text{col}(a_{h_c}) \notin S \) and \(-\infty\) otherwise. Given a random coloring, the probability that the ads composing the best allocation are colorful is \( \frac{K^1}{K^2} \). Thus, repeating the procedure \( r e^K \) times, where \( r \geq 1 \), the probability of finding the best allocation is \( 1 - e^{-r} \). The complexity is \( O((2c)^K N c^{+2}) \). The algorithm can be derandomized by means of a hash function with an additional cost of \( O(\log^2(N)) \) [4].

By applying the above algorithm to the first \( K' \) slots, where \( K' = \min\{K, \lceil \log(N) \rceil \} \), we obtain an algorithm with complexity \( O(K^{3.5} N c^{+2} \log^2(N)) \). We observe that if \( c \) is not a constant, the complexity is exponential. It is not too hard to note that such an algorithm is \( \frac{\log(N)}{2 \min\{N,K\}} \)-approximate.

**Proposition 4.** Given \( \theta^* \), the optimal allocation over \( K \) slots, and \( \theta^*_{K'} \), the optimal allocation over the first \( K' \leq \min\{N,K\} \) slots, we have \( SW(\theta^*_{K'}) \geq \frac{1}{2 \min\{N,K\}} SW(\theta^*) \).

**Proof.** Partition \( K'' = \min\{N,K\} \) slots in groups of \( K' \) consecutive slots. There could be remaining slots that will constitute the last group with less then \( K' \) slots. The number of groups in which the \( K \) slots are divided is \( NG = \lceil \frac{K''}{K'} \rceil \). Call \( G_i = \{(i-1)K'+1, \ldots, \min(iK',K)\} \), for \( i \in \mathcal{NG} = \{1, \ldots, NG\} \), the \( i \)-th group of indices of \( K' \) slots.

We let \( SW(\theta, v|G_i) = \sum_{m \in G_i} \Lambda_m \Gamma_{\alpha(m;\theta)}(\theta) q_{\alpha(m;\theta)} v_{\alpha(m;\theta)} \), for any \( \theta \in \Theta \). Since \( SW(\theta^*, v) = \sum_{i=1}^{NG} SW(\theta^*, v|G_i) \), there must exist a group \( G_i \) s.t. \( SW(\theta^*, v|G_i) \geq \frac{1}{NG} SW(\theta^*, v) \). Observing that \( \lceil \frac{K''}{K'} \rceil \leq \frac{K''}{K'} + 1 \) and \( K' \leq K'' \) we get \( SW(\theta^*, v|G_i) \geq \frac{K'}{2K'} SW(\theta^*, v) \). The proof concludes by noting that, by optimality, \( SW(\theta^*_{K'}, v|G_i) \geq SW(\theta^*, v|G_i) \). \( \square \)
Moreover, this algorithm is MIR and as such can be used to design a truthful mechanism.

### 4.3.2 CFNE\(_{aa}^+(c)\)–nr is APX–complete

In this section we introduce a subset of instances of CFNE\(_{aa}^+(c)\)–nr, denoted by CFNE\(_{aa}^+(c)\)–nr. The instances of CFNE\(_{aa}^+(c)\)–nr are such that their contextual graph is complete and \(0 < \gamma_{\text{min}} = \min_{i,j \in \mathcal{N}, i \neq j} \gamma_{i,j}\). We prove that in this subset the problem becomes APX–complete, indeed we first show that the problem is still APX–hard and then that any \(\alpha\)-approximate algorithm for Weighted 3-Set Packing (W3SP) can be turned into an \((\alpha \gamma_{\text{min}}^c)\)–approximation algorithm for CFNE\(_{aa}^+(c)\)–nr.

#### 4.3.2.1 CFNE\(_{aa}^+(1)\)–nr hardness

In this section we revisit the APX–hardness proof of CFNE\(_{aa}^+(c)\)–{nr, r}, in order to show that, with some adjustments, even CFNE\(_{aa}^+(c)\)–nr is APX–hard. The proof is still conducted by a reduction from the TSP\((1,2)\) problem.

We recall that an instance of TSP\((1,2)\) consists of a complete weighted graph \(H = (\mathcal{V}, \mathcal{V}^2, w)\) where \(\mathcal{V} = \{1, \ldots, V\}\) is the set of \(V\) vertices of \(H\) and \(w : \mathcal{V}^2 \to \{1, 2\}\) is the weight function defined on the edges of \(H\), which can take only values in \{1, 2\}. A solution to the TSP\((1,2)\) problem is a Hamiltonian tour of minimum cumulative weight.

**Definition 7. (Reduction from TSP\((1,2)\) to CFNE\(_{aa}^+(1)\)–nr).** Given an instance \(H\) of TSP\((1,2)\), for each vertex \(i \in \{1, \ldots, V - 1\}\), we construct an ad \(a_i\), whereas for vertex \(V\) we construct two ads: \(a_V\) and \(a_{V+1}\). Thus, we obtain a set \(\mathcal{N} = \mathcal{V} \cup \{V + 1\}\) comprising \(N = V + 1\) ads. For each ad \(a_i\), we set \(q_i = 1\) and \(v_i = 1\). For each \(m \in \mathcal{K}\), we set \(\Lambda_m = 1\). Parameters \(\gamma_{i,j}\) are defined as in (4.6). The instance of CFNE\(_{aa}^+(1)\)–nr has \(K = V + 1\) slots.

\[
\gamma_{i,j} = \begin{cases} 
1 & \text{if } i \leq V, j < V, i \neq j \text{ and } p_{i,j} = 1 \\
\beta & \text{if } i \leq V, j < V, i \neq j \text{ and } p_{i,j} = 2 \\
1 & \text{if } i < V \text{ and } j = V + 1 \text{ and } p_{i,V} = 1 \\
\beta & \text{if } i < V \text{ and } j = V + 1 \text{ and } p_{i,V} = 2 \\
0 & \text{if } j \in \mathcal{N} \text{ and } a_i = a_{\bot} \\
0 & \text{if } i \in \mathcal{N} \text{ and } a_j = a_{\bot} \\
\varsigma & \text{otherwise}
\end{cases}
\]
To complete the reduction, we need to show that we can map an α–approximate solution of $\text{CFNE}_{aa}^+(1)$–nr back to $TSP(1,2)$. To accomplish this, we will restrict ourselves to allocations $\theta$ for the $\text{CFNE}_{aa}^+(1)$–nr problem that are well–formed, meaning that they are: (i) $a_\perp$–free (i.e., $\theta$ does not contain any fictitious ad) and (ii) have ad $a_V$ in the first slot and $a_{V+1}$ in the last slot. Given a well–formed $\alpha$–approximate allocation $\theta_\alpha = (a_V, a_1, \ldots, a_K, a_{V+1})$ we can easily obtain a $\beta$–approximate tour $\tau_\beta = (t_v, t_1, \ldots, t_K)$ by simply traversing the nodes in the order in which the corresponding ads are listed in $\theta_\alpha$.

Before proving our main result of this section, we next prove that restricting to well–formed allocations comes at no extra approximation cost, as given a non–well–formed allocation we can always obtain in polynomial time a well–formed allocation with a non-smaller social welfare value.

**Proposition 5.** Given a non–well–formed allocation $\theta_{NW}$ for a $\text{CFNE}_{aa}^+(1)$–nr instance obtained by the reduction of Definition 7, it is possible to construct a well formed allocation $\theta_W$ s.t. $SW(\theta_W, v) \geq SW(\theta_{NW}, v)$, provided that $\beta > \max\{\frac{2+\varepsilon}{3}, \frac{3}{4}\}$.

**Proof.** We exhibit a 2–step procedure that given in input a non–well–formed allocation computes in polynomial time a well–formed allocation while not decreasing the social welfare. In Step 1, the procedure removes all ads $a_\perp$, thus obtaining an $a_\perp$–free allocation $\theta_F$, whereas in Step 2, starting from $\theta_F$ it computes an allocation s.t. ads $a_V$ and $a_{V+1}$ are allocated in the first and last position, respectively.

**Step 1: removing $a_\perp$.** Suppose that $\theta_{NW}$ allocates $a_\perp$ (or otherwise we can skip to Step 2). We show that we obtain a better allocation without $a_\perp$. We enumerate the possible situations, analyzing in depth just one of them (the others can be proved similarly).

1. Consider allocations $\theta_{NW} = (a_{i_1}, \ldots, a_{i_j}, a_{V+1}, a_\perp, a_V, a_{i_{j+1}}, \ldots, a_{i_{V-2}}, a_{i_{V-1}})$ and $\theta' = (a_V, a_{i_1}, \ldots, a_{i_j}, a_z, a_{i_{j+1}}, \ldots, a_{i_{V-2}}, a_{V+1}), z < V$. 

$SW(\theta_{NW}, v)$ can be written as $SW(\theta_{NW}, v) = \sum_{h=1}^{\gamma} \phi_h$ where $\phi_1 = 1$ ($\phi_1$ is the contribution of $a_{i_1}$ only), $\phi_2 = \sum_{l=2}^{j} \gamma_{i_{l-1}, i_l}$, $\phi_3 = \gamma_{i_{j}, V+1}$, $\phi_4 = 0$ (the contribution of $a_\perp$), $\phi_5 = \gamma_{i_{j}, V+1}$, $\phi_6 = \gamma_{V, i_{j+1}}$, and $\phi_7 = \sum_{l=j+2}^{V-2} \gamma_{i_{l-1}, i_l}$. We can write $SW(\theta', v) = \sum_{h=1}^{\gamma} \phi'_h$ where $\phi'_1 = 1$ (which accounts for the contribution of ad $a_V$), $\phi'_2 = \gamma_{V, i_{j+1}}, \phi'_3 = \sum_{l=2}^{j} \gamma_{i_{l-1}, i_l} = \phi_2$, $\phi'_4 = \gamma_{i_{j}, z}, \phi'_5 = \gamma_{z, i_{j+1}}, \phi'_6 = \sum_{l=j+2}^{V-2} \gamma_{i_{l-1}, i_l} = \phi_5, \phi'_7 = \gamma_{i_{V-2}, V+1}$. Notice that $\phi'_2, \phi'_4, \phi'_5, \phi'_7 \geq \beta$. Thus, in the worst case we have that $SW(\theta', v) = \sum_{h=1}^{\gamma} \phi'_h > \sum_{h=1}^{\gamma} \phi_h = SW(\theta_{NW}, v)$ holds when $\beta > \frac{3}{4}$.
2. Consider allocation $\theta_{NW} = (\ldots, a_i, a_{\perp}, a_j, \ldots)$ where $i, j$ can be any, including $\perp$. It can be easily observed that by substituting the first $a_{\perp}$ with a non-allocated ad $a_z$ where $z \in \mathcal{N}$, obtaining $\theta' = (\ldots, a_i, a_z, a_{\perp}, a_j, \ldots)$, we have $SW(\theta', v) \geq SW(\theta_{NW}, v)$. Indeed, $SW(\theta', v) - SW(\theta_{NW}, v) \geq 0$. Notice that $SW(\theta', v) - SW(\theta_{NW}, v) = 0$ either if $i = V + 1$ or if $i = V$ and $z = V + 1$, while it is strictly positive in all the other cases. Thus, if in $\theta$ a sequence of consecutive $a_{\perp}$ appears, we can substitute all of them except the last one with ads $a_z$ with $z \in \mathcal{N}$.

3. Consider allocation $\theta_{NW} = (\ldots, a_i, a_{\perp}, a_j, \ldots)$ where $i, j < V$. If $\theta' = (\ldots, a_i, a_z, a_j, \ldots)$ with $z < V$ then $SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (a_z, \ldots, a_i, a_j, \ldots), z = V, SW(\theta') > SW(\theta_{NW})$ when $\beta > \frac{1}{2}$. If $\theta' = (\ldots, a_i, a_j, \ldots, a_z), z = V + 1, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$.

4. Consider allocation $\theta_{NW} = (\ldots, a_V, a_{\perp}, a_j, \ldots)$. If $\theta' = (\ldots, a_V, a_z, a_j, \ldots), j, z < V, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$. If $\theta' = (\ldots, a_V, a_j, \ldots, a_z), j < V, and z = V + 1, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$.

5. Consider allocation $\theta_{NW} = (\ldots, a_i, a_{\perp}, a_V, \ldots)$. If $\theta' = (a_V, \ldots, a_i, a_z, \ldots), i, z < V, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$. If $\theta' = (a_V, \ldots, a_i, a_j, \ldots, a_z), i < V, and z = V + 1, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

6. Consider allocation $\theta_{NW} = (\ldots, a_{V+1}, a_{\perp}, a_j, \ldots)$. If $\theta' = (\ldots, a_z, a_{V+1}, a_j, \ldots, a_{V+1}), j, z < V, SW(\theta') > SW(\theta_{NW})$ when $\beta > \frac{2}{3}$. If $\theta' = (a_z, \ldots, a_j, \ldots, a_{V+1}), j < V, and z = V, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{2}{3}$.

7. Consider allocation $\theta_{NW} = (\ldots, a_{i}, a_{\perp}, a_{V+1}, \ldots)$. If $\theta' = (\ldots, a_i, a_z, a_{V+1}, \ldots, a_{V+1}), i, z < V, SW(\theta') > SW(\theta_{NW})$ when $\beta > \frac{1}{2}$. If $\theta' = (a_v, \ldots, a_i, \ldots, a_z), i < V, and z = V + 1, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1 + \epsilon}{3}$.

8. Consider allocation $\theta_{NW} = (\ldots, a_{V}, a_{\perp}, a_{V+1}, \ldots)$. If $\theta' = (\ldots, a_{V}, a_z, a_{V+1}, \ldots), z < V, SW(\theta', v) > SW(\theta_{NW}, v)$ when $\beta > \frac{1}{2}$.

9. Consider allocation $\theta_{NW} = (a_i, a_{\perp}, a_j, \ldots)$. When $\beta > \frac{3}{4}$, it is always possible to find an allocation $\theta'$ where $a_{\perp}$ has been removed and $SW(\theta', v) > SW(\theta_{NW}, v)$.
10. Consider allocation \( \theta_{NW} = (\ldots, a_i, a_\bot, a_j) \). When \( \beta > \frac{3}{4} \), it is always possible to find an allocation \( \theta' \) where \( a_\bot \) has been removed and \( SW(\theta', v) > SW(\theta_{NW}, v) \).

Step 1 is iterated until all \( a_\bot \) ads are removed and an \( a_\bot \)-free allocation \( \theta_F \) is obtained.

**Step 2: moving \( a_V \) and \( a_{V+1} \).** We are given in input an \( a_\bot \)-free allocation \( \theta_F \). Let us suppose that ad \( a_V \) and \( a_{V+1} \) are not allocated in the first and last slots, respectively, as otherwise \( \theta_F \) would already be well–formed. We enumerate all the possible situations and we provide details only for one of them:

1. Consider allocation \( \theta_F \) where there is no \( a_\bot \), \( a_V \) is not displayed in the first slot and \( a_{V+1} \) is not displayed in the last slot. We rename the ads such that \( \theta_F = (a_1, a_2, \ldots, a_i, a_{V+1}, a_{i+1}, \ldots, a_j, a_V, a_{j+1}, \ldots, a_{V-1}) \).

We show that in this case moving \( a_V \) into the first slot and \( a_{V+1} \) into the last slot strictly increases the social welfare. \( SW(\theta_F) \) can be written as \( SW(\theta_F) = \sum_{h=1}^{8} \phi_h \) where \( \phi_1 = 1 \) (\( \phi_1 \) is the contribution of \( a_1 \) only), \( \phi_2 = \sum_{l=2}^{i+1} \gamma_{l-1,l} \), \( \phi_3 = \gamma_{i,V+1} \), \( \phi_4 = \gamma_{V+1,i+1} = \varsigma \), \( \phi_5 = \sum_{l=i+2}^{j} \gamma_{l-1,l} \), \( \phi_6 = \gamma_{j,V} = \varsigma \), \( \phi_7 = \gamma_{V,j+1} \) and \( \phi_8 = \sum_{l=j+2}^{V-1} \gamma_{l-1,l} \).

Call \( \theta_W \) the allocation obtained from \( \theta_F \) by moving \( a_V \) to the first slot and \( a_{V+1} \) to the last slot, i.e., \( \theta_W = (a_V, a_1, \ldots, a_i, a_{V+1}, \ldots, a_j, a_{j+1}, \ldots, a_{V-1}, a_{V+1}) \).

We can write \( SW(\theta_W, v) = \sum_{h=1}^{8} \phi'_h \) where \( \phi'_1 = \gamma_{V+1} \), \( \phi'_2 = \sum_{l=2}^{i+1} \gamma_{l-1,l} = \phi_2 \), \( \phi'_3 = \gamma_{V+1,i+1} = \phi_3 \), \( \phi'_4 = \gamma_{i,i+1} \), \( \phi'_5 = \sum_{l=i+2}^{j} \gamma_{l-1,l} = \phi_5 \), \( \phi'_6 = \gamma_{j,V} = \phi_6 = 1 \) (which accounts for the contribution of ad \( a_V \)), \( \phi'_7 = \gamma_{j,j+1} \) and \( \phi'_8 = \sum_{l=j+2}^{V-1} \gamma_{l-1,l} = \phi_8 \). Notice that \( \phi'_1, \phi'_3, \phi'_4, \phi'_7, \phi'_8 \geq \beta \).

Even in the case in which \( SW(\theta_F, v) \) is as large as it is possible (i.e., when \( \phi_1, \phi_3, \phi_7 = 1 \)), we have \( SW(\theta_W, v) > SW(\theta_F, v) \). Indeed:

\[
\sum_{i=1}^{8} \phi'_i > \sum_{i=1}^{8} \phi_i \\
\phi'_1 + \phi'_3 + \phi'_4 + \phi'_6 + \phi'_7 > \phi_1 + \phi_3 + \phi_4 + \phi_6 + \phi_7 \\
\beta + \beta + 1 + \beta > 1 + 1 + \varsigma + \varsigma + 1 \\
4\beta > 2(1 + \varsigma) \\
\beta > \frac{1 + \varsigma}{2}
\]

that holds given that \( \beta > \frac{1+\varsigma}{2} \) and for \( \varsigma < 1 \), given that \( \forall \varsigma < 1 \beta < 1 \).
Thus moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. This concludes the proof.

2. Consider allocation $\theta_F$ where there is no $a_\bot$, $a_V$ is not displayed in the first slot and $a_{V+1}$ is not displayed in the last slot. We rename the ads such that $\theta_F = (a_1, a_2, \ldots, a_i, a_V, a_{V+1}, a_{j+1}, \ldots, a_{V-1})$. It is easy to notice that $\theta_2 = (a_1, a_2, \ldots, a_i, a_{V+1}, a_V, a_{i+1}, \ldots, a_{V-1})$ provides a larger social welfare.

We, now, show that in this case moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. All the other cases holds the same *a fortiori*. $SW(\theta_2, v)$ can be written as $SW(\theta_2) = \sum_{h=1}^{6} \phi_h$ where $\phi_1 = 1$ ($\phi_1$ is the contribution of $a_1$ only), $\phi_2 = \sum_{l=2}^{i-2} \gamma_{l-1,l}$, $\phi_3 = \gamma_{i,V+1}$, $\phi_4 = \gamma_{V+1,V} = \varsigma$, $\phi_5 = \gamma_{V,i+1}$, $\phi_6 = \sum_{l=i+2}^{V-1} \gamma_{l-1,l}$. Call $\theta_W$ the allocation obtained from $\theta_2$ by moving $a_V$ to the first slot and $a_{V+1}$ to the last slot, i.e., $\theta_3 = (a_V, a_1, \ldots, a_i, a_{i+1}, \ldots, a_{V-1}, a_{V+1})$. We can write $SW(\theta_W, v) = \sum_{h=1}^{6} \phi'_h$ where $\phi'_1 = \gamma_{V,1}$, $\phi'_2 = \sum_{l=2}^{i} \gamma_{l-1,l} = \phi_2$, $\phi'_3 = \gamma_{V-1,V+1}$, $\phi'_4 = \gamma_{i,i+1}$, $\phi'_5 = 1$ (the contribution of $a_V$ in the first slot), $\phi'_6 = \sum_{l=i+2}^{V-1} \gamma_{l-1,l} = \phi_6$. Notice that $\phi'_1, \phi'_3, \phi'_5 \geq \beta$.

Even in the case in which $SW(\theta_2, v)$ is as large as it is possible (i.e., when $\phi_1, \phi_3, \phi_5 = 1$), we have $SW(\theta_W, v) > SW(\theta_2, v)$. Indeed:

\[
\sum_{i=1}^{6} \phi'_i > \sum_{i=1}^{6} \phi_i
\]

\[
\phi'_1 + \phi'_3 + \phi'_4 + \phi'_5 > \phi_1 + \phi_3 + \phi_4 + \phi_5
\]

\[
\beta + \beta + 1 > 1 + 1 + \varsigma + 1
\]

\[
3\beta > 2 + \varsigma
\]

\[
\beta > \frac{2 + \varsigma}{3}
\]

that holds given that $\beta > \frac{2 + \varsigma}{3}$ and for $\varsigma < 1$, given that $\forall \varsigma < 1 \beta < 1$.

Thus moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. This concludes the proof.

3. Consider an allocation $\theta_F = (\ldots, a_V, \ldots, a_{V+1}, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1})$, $SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{1 + \varsigma}{2}$;

4. Consider an allocation $\theta_F = (a_{V+1}, \ldots, a_V, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1})$, $SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{2 + \varsigma}{3}$;
5. Consider an allocation $\theta_F = (a_V, \ldots, a_{V+1}, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{1+\varsigma}{2}$;

6. Consider an allocation $\theta_F = (\ldots, a_V, \ldots, a_{V+1})$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$;

7. Consider an allocation $\theta_F = (\ldots, a_{V+1}, \ldots, a_V)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{2+\varsigma}{3}$;

8. Consider an allocation $\theta_F = (a_{V+1}, \ldots, a_V)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{1+\varsigma}{2}$;

9. Consider an allocation $\theta_F = (a_{V+1}, \ldots, a_V)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$;

10. Consider an allocation $\theta_F = (a_V, a_{V+1}, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$;

11. Consider an allocation $\theta_F = (a_V, a_{V+1}, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$;

12. Consider an allocation $\theta_F = (a_{V+1}, a_V, \ldots)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$;

13. Consider an allocation $\theta_F = (\ldots, a_V, a_{V+1})$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \frac{1+\varsigma}{2}$;

14. Consider an allocation $\theta_F = (\ldots, a_{V+1}, a_V)$ and $\theta_W = (a_V, \ldots, a_{V+1}), SW(\theta_W, v) > SW(\theta_F, v)$ when $\beta > \varsigma$.

All the previous cases hold given that $\beta > \max \left\{ \frac{2+\varsigma}{3}, \frac{3}{4} \right\}$. Thus moving $a_V$ into the first slot and $a_{V+1}$ into the last slot strictly increases the social welfare. \qed

We are now ready to prove our APX–hardness result.

**Theorem 8.** \( \text{CFNE}_{aa}^+ (1) – \{ r, tr \} \) is APX–hard.

**Proof.** With a slight abuse of notation, we denote by $SW(\tau)$ the value of the tour $\tau$ in the TSP(1, 2) problem, while $SW(\theta, v)$ is the social welfare of $\theta$ in CFNE$_{aa}^+ (1)$–nr. Since by Proposition 5. we can now assume that $\theta$ is well–formed, it is easy to check that an optimal solution of the TSP(1, 2) problem (denoted by $\tau^*$) corresponds to an optimal solution of the CFNE$_{aa}^+ (1)$–nr (denoted by $\theta^*$) where the ads are allocated into the slots in the same order as the tour $\tau^*$ starting from $a_V$; $a_{V+1}$ is the last
ad and no \( a_\perp \) is present. For the sake of presentation, suppose the vertices of the graph and the ads are renamed s.t. the optimal solution \( \theta^* \) of CFNE\(_{aa}^+\)(1)–nr is \( \theta^* = (a_V, a_1, a_2, \ldots, a_{V-1}, a_{V+1}) \). Let \( \alpha \) be the number of edges of weight 2 in \( \tau^* \). Then, there are \( \alpha \) couples \((a_i, a_j)\) of consecutive ads with \( \gamma_{i,j} = \beta \). Thus, \( SW(\tau^*) = (N - \alpha) \cdot 1 + \alpha \cdot 2 = N + \alpha \) and \( SW(\theta^*, v) = (N + 1 - \alpha) \cdot 1 + \alpha \cdot \beta \).

Let us assume we have a well–formed solution \( \theta' \) s.t. \( \frac{SW(\theta', v)}{SW(\theta^*, v)} \geq 1 - \epsilon \). Suppose this solution contains \( \alpha' \) couples \((a_i, a_j)\) of consecutive ads with \( \gamma_{i,j} = \beta \).

Given that \( \theta' \) is an \((1 - \epsilon)\)–approximation of \( \theta^* \), \( \alpha' \) must be s.t. the following condition holds: \( SW(\theta', v) = N + 1 - \alpha' + \alpha' \beta \geq (1 - \epsilon)(N + 1 - \alpha + \alpha \beta) = (1 - \epsilon)SW(\theta^*, v) \), from which we obtain the following upper bound on \( \alpha' \):

\[
\alpha' \leq (1 - \epsilon)\alpha + \frac{\epsilon(N + 1)}{1 - \beta} \tag{4.8}
\]

From \( \theta' = (a_V, a_1, \ldots, a_{V-1}, a_{V+1}) \) we can produce a solution for TSP(1, 2) with the following tour \( \tau' = (a_V, a_1, \ldots, a_{V-1}) \); \( \tau' \) will contain \( \alpha' \) edges with weight 2 and then \( SW(\tau') = N + \alpha' \). Thus, we can now show that \( \tau' \) is a \((1 + \epsilon/(1 - \beta))\)–approximation of \( \tau^* \):

\[
SW(\tau') = N + \alpha' \leq N + (1 - \epsilon)\alpha + \frac{\epsilon(N + 1)}{1 - \beta} \\
\leq N + \alpha + \frac{\epsilon(N + \alpha)}{1 - \beta} \leq \left(1 + \frac{\epsilon}{1 - \beta}\right)(N + \alpha) \\
= \left(1 + \frac{\epsilon}{1 - \beta}\right)SW(\tau^*)
\]

where the first inequality follows from (4.8). Given that TSP(1, 2) is known to be APX–hard, \( \epsilon \) cannot go arbitrarily close to zero and CFNE\(_{aa}^+\)(1) is then APX–hard.

4.3.2.2 CFNE\(_{aa}^+(c)\)–nr completeness

The Weight 3 Set Packing problem (W3SP) consists of finding a sub-collection of pairwise-disjoint subsets of maximal weight, where each subset has a cardinality at most 3 and has an associated to a weight. Literature provides several constant-ratio approximate algorithms to solve this problem, e.g., the algorithm in [14] provides a 1/2-approximation in time quadratic in the
4.3. Approximation algorithms for $\text{CFNE}_{\alpha\gamma}(c)$

number of sets. We now present an approximation preserving reduction from $\text{CFNE}_{\alpha\gamma}(c)$–nr to W3SP, similar in spirit to that defined, for positive only externalities, i.e., an ad influences an other augmenting its CTR, in [22].

**Theorem 9.** Given an $\alpha$–approximate algorithm for W3SP, we can obtain an $(\alpha\gamma_{\min})$–approximation algorithm for $\text{CFNE}_{\alpha\gamma}(c)$–nr.

**Proof.** Given an instance of $\text{CFNE}_{\alpha\gamma}(c)$–nr, we obtain an instance of W3SP by means of the following reduction. To simplify the presentation, we suppose that $K$ is even (the proof can be easily extended for an odd $K$).

We divide the $K$ into $K/2$ blocks of two slots each. We construct a collection of $K/2\cdot\binom{N}{2}$ sets, each set having the form $\{a_i,a_j,p\}$, where $p \in \{1,3,5,\ldots,K-1\}$ and $i,j \in \mathcal{N}$. The weight of a set is defined as the maximum social welfare that ads $a_i$ and $a_j$ can provide when assigned to slots $s_p$ and $s_{p+1}$ without taking into considerations the externalities of $a_i$ and $a_j$ on the ads allocated to slots $s_m$, $m \neq p, p+1$. Specifically, $W(a_i,a_j,p) = \max\{\Lambda_pq_i v_i + \Lambda_{p+1}q_j v_j, \Lambda_p q_j v_j + \Lambda_{p+1}q_i v_i\}$. Note that there is an immediate mapping between solutions of W3SP and $\text{CFNE}_{\alpha\gamma}(c)$–nr. For a solution $\theta^*_{\gamma}$ of W3SP, let $W(\theta^*_{\gamma})$ denote its total weight. Now, let $\theta^*_S$ and $\theta^*$ denote, respectively, an optimal allocation for W3SP and an optimal allocation for $\text{CFNE}_{\alpha\gamma}(c)$–nr. Furthermore, let $\theta^*_S$ be an $\alpha$–approximate solution for W3SP, and $\theta^*$ be the corresponding solution to $\text{CFNE}_{\alpha\gamma}(c)$–nr. Since in W3SP, outer-block externalities are not taken into consideration, we have: $W(\theta^*_S) \geq SW(\theta^*,v)$ and $SW(\theta^*,v) \geq \gamma_{\min}^\alpha W(\theta^*_S)$. From these inequalities we obtain: $SW(\theta^*,v) \geq \gamma_{\min}^c W(\theta^*_S) \geq \alpha \gamma_{\min}^c SW(\theta^*,v)$.

We can finally conclude:

**Corollary 1.** If $\gamma_{\min}$ is bounded from below by a constant (i.e., $\gamma_{\min} \in \Omega(1)$), then $\text{CFNE}_{\alpha\gamma}(c)$–nr is approximable within a constant factor.

And the corollary above then proves that for $\gamma_{\min} \in \Omega(1)$, $\text{CFNE}_{\alpha\gamma}(c)$–nr is APX-complete.

**Non monotonicity** It can be easily shown that the above algorithm is not monotone by the following counterexample. Consider an instance $I$ of $\text{CFNE}_{\alpha\gamma}(1)$–nr with $N = K = 4$ wherein $\Lambda_3 \gamma_{z,4} < \Lambda_4 \gamma_{3,4}$, for $z \in \{1,2\}$, $v_1, v_2 \gg v_3, v_4$ and $\gamma_{1,2} = \gamma_{2,1} = 1$ so that $W(a_1,a_2,1)$ is much bigger than any other $W(a_i,a_j,1)$. Therefore, any reasonable approximation of the W3SP instance constructed upon $I$ must return sets $\{a_1,a_2,1\}$ and
\( \{a_3, a_4, 3\} \). Additionally consider \( v_4 < \frac{\Lambda_4 \gamma_{4,3}}{\Lambda_3 - \Lambda_3 \Lambda_4 \gamma_{3,4}} \) so that \( W(a_3, a_4, 3) = \Lambda_3 q_3 v_3 + \Lambda_4 \gamma_{3,4} q_4 v_4 \). So the solution \( \theta \) returned by the algorithm run on \( I \) places \( a \) in \( s_4 \), resulting in \( CTR_4(\theta) = q_4 \Lambda_4 \gamma_{3,4} \). Take now the instance \( I' \) defined as \( I \) except that \( v_1, v_2 \gg v'_4 > \frac{\Lambda_4 \gamma_{4,3}}{\Lambda_3 - \Lambda_3 \Lambda_4 \gamma_{3,4}} > v_4 \). As before, the approximation algorithm for W3SP will return sets \( \{a_1, a_2, 1\} \) and \( \{a_3, a_4, 3\} \) but this time \( W'(a_3, a_4, 3) = \Lambda_3 q_4 v_4 + \Lambda_4 \gamma_{4,3} q_3 v_3 \). Therefore, the solution \( \theta' \) returned by the algorithm run on \( I' \) places ad \( a_4 \) in slot \( s_3 \), i.e., \( CTR_4(\theta') = q_4 \Lambda_3 \gamma_{z,4} \), where \( z \in \{1, 2\} \) is the ad placed in slot \( s_2 \) in the allocation \( \theta' \). The algorithm is therefore not monotone and cannot be used to design a truthful mechanism.

### 4.3.3 CFNE\(_{aa}(c)–r\) is APX–complete

We present a \( \frac{1}{2} \)-approximate polynomial–time greedy algorithm for CFNE\(_{aa}(c)–r\) without any assumption on \( c \), thus showing that this problem is APX–complete.

We sort the ads \( a_i, i \in \mathcal{N} \), in non-increasing order of \( q_i v_i \). W.l.o.g., label the ads such that \( q_1 v_1 \geq q_2 v_2 \geq \ldots \geq q_N v_N \). We allocate ads in this order in odd slots. In all the other slots we allocate \( a_\perp \).

**Proposition 6.** The greedy algorithm described above is \( \frac{1}{2} \)-approximate.

**Proof.** We denote with \( \theta_{1/2} \) the allocation obtained by the algorithm. We want to prove that \( SW(\theta_{1/2}, \mathbf{v}) \geq SW(\theta^*, \mathbf{v})/2 \). W.l.o.g., rename the ads so that \( q_1 v_1 \geq q_2 v_2 \geq \ldots \geq q_N v_N \). Let \( K' = \lceil K/2 \rceil \). We have \( SW(\theta_{1/2}, \mathbf{v}) = \sum_{m \leq K'} \Lambda_{2m-1} q_m v_m \). (This is because \( \gamma_{i,\perp} = 1 \) and \( \gamma_{\perp,i} = 1 \).) On the other hand, \( SW(\theta^*, \mathbf{v}) \leq \sum_{m \leq K} \Lambda_m q_m v_m \). Since \( \Lambda_i q_i v_i \geq \Lambda_{i+1} q_{i+1} v_{i+1} \), we have \( \Lambda_i q_i v_i \geq 1/2 \sum_{m=i+1} \Lambda_m q_m v_m \). We can conclude \( SW(\theta_{1/2}, \mathbf{v}) = \sum_{m \leq K} \Lambda_{2m-1} q_m v_m \geq 1/2 \sum_{m \leq K} \Lambda_{2m-1} q_{2m-1} v_{2m-1} \geq 1/2 \sum_{m \leq K} \Lambda_m q_m v_m \geq SW(\theta^*, \mathbf{v})/2 \).

This algorithm is MIR, the range being the set of allocations in which (non-fictitious) ads can be displayed only in the odd slots. Therefore, we can design a truthful \( \frac{1}{2} \)-approximate mechanism for CFNE\(_{aa}(c)–r\).

It can be easily noticed that this approximation algorithm provides the same results with the CFNE\(_{aa}(c)–tr\) model, but for this model the problem of determining its complexity class is still open and will be object of our future research activity. We can only conclude that CFNE\(_{aa}(c)–tr\) is an APX problem.
In the previous chapter we focused on the $\text{CFNE}_{aa}(c)$ and $\text{CFNE}_{sa}(c)$ models when $c < K$, here we focus only on the $\text{CFNE}_{aa}(c)$ in the particular case where $c = K$.

In Sec. 5.1 we show that all the $\text{CFNE}_{aa}(K)$ models are NP–hard, and, in Sec. 5.2, we propose an algorithm for finding the optimal allocation in all the three models $\text{CFNE}_{aa}(K)$–{nr, r, tr}. Given the hardness of the problem, in Sec. 5.2.1 we isolate instances of $\text{CFNE}_{aa}(K)$–nr that can be solved in polynomial time.

We complete the characterization of the $\text{CFNE}_{aa}(K)$–nr problem showing that it belongs to the poly–APX–complete complexity class: in Sec. 5.3 we prove the hardness and in Sec. 5.4 the completeness, showing the existence of an approximation algorithm providing a ratio of $O\left(\frac{\log(N)}{\min\{N,K\}}\right)$. This algorithm is not monotone, thus in Sec. 5.5, at the cost of a worse ratio, we introduce an approximation algorithm that can be used to design a DSIC mechanism.

Finally, we prove the APX–hardness of $\text{CFNE}_{aa}(K)$–{r, tr} in Sec. 5.6 and its completeness, in Sec. 5.7 designing an algorithm that achieves a constant ratio of $\frac{1}{2}$. 
5.1 CFNE\textsubscript{aa}(K)–\{nr, r, tr\} is NP–hard

In this section we show that CFNE\textsubscript{aa}(K)–\{nr, r, tr\} is NP–hard.

**Theorem 10.** Given an instance of CFNE\textsubscript{aa}(K)–\{nr, r, tr\} and a constant $\chi$, the problem of deciding whether there exists $\theta \in \Theta$ s.t. $SW(\theta) \geq \chi$ is NP-hard.

**Proof.** We reduce from the Hamiltonian Path problem. We are given a direct graph $G' = (T, A)$ and need to determine if there exists a path that visits each vertex exactly once. Let $I = |T|$. We map every vertex $t_i$ of $G'$ to an ad $a_i$ and define $q_i = v_i = 1$. We have $I$ slots, with $\Lambda_m = 1 \forall m \in I = \{1, \ldots, I\}$. We now define the contextual graph $G = (I, E)$ in the following way: $(i, j) \in E \iff (t_i, t_j) \in A$. All the edges in $E$ weigh 1 and $\chi = I$. No matter the model of reset, it can be easily noticed that the highest value of social welfare reachable by an allocation is $I$. This value is obtained in a solution $\theta$ when all the $I$ allocated ads are different from $a_\bot$ and $\forall m < I, \gamma_{\alpha(m;\theta),\alpha(m+1;\theta)} = 1$. By definition this corresponds to a Hamiltonian path in $G'$. Thus, the optimum for CFNE\textsubscript{aa}(K)–\{nr, r, tr\} is $I$ iff there is an Hamiltonian Path in $G'$.

This result allows an interesting parallel with the classic Cascade Model, for which NP-hardness is only conjectured. Moreover, although NP-hard, it could be the case that significant instances of CFNE\textsubscript{aa}(K)–\{nr, r, tr\} can be solved exactly in reasonable time by means of an exponential–time algorithm.

5.2 Exact algorithm

Since there is no efficient algorithm for solving CFNE\textsubscript{aa}–\{nr, r, tr\} (unless NP=P), in this section we seek to find a viable algorithmic approach having the best possible asymptotic complexity, bearing in mind that polynomial time complexity is unattainable.

We start our analysis by observing that the optimization version of the Longest Path problem we use in our APX–hardness proof can be solved in $O((2e)^{|T|}|T|^3 \log^2(|T|))$, where $|T|$ is the number of the vertices, by using the Color Coding technique (CC) \[4\] that we have already faced in Sec. 4.3.1. However, when applied to CFNE\textsubscript{aa}–\{nr, r, tr\}, CC has a complexity of $O((2e)^K K^2 N^{K+1}(\log N)^2)$ and is outperformed by the $O(N^K)$ time complexity of any branch–and–bound algorithm.

Intuitively, CC generates some colorings and, for each coloring, CC assigns a color to the vertices (ads in CFNE\textsubscript{aa}(K)–\{nr, r, tr\}) and finds, in
5.2. Exact algorithm

a dynamic programming fashion, the best solution in which only one vertex (ad) per color is allocated. The key feature of CC is that, for each pair \((S, v)\), where \(S\) is a subset of colors and \(v\) is a vertex (ad), it is possible to store only one (the best in terms of the length of the path) partial solution in which only vertices with colors in \(S\) are allocated and the last vertex is \(v\). In \(\text{CFNE}_{aa}(K)-\{\text{nr, r, tr}\}\) this is not possible, since we could have an exponential number of Pareto efficient solutions for each \((S, a)\) (we report details when we describe our approximation algorithm for \(\text{CFNE}_{aa}(K)-\text{nr}\), Sec. 5.4).

For these reasons, we propose a branch-and-bound algorithm (see Algorithm 2), which employs a suitably defined bounding function in order to safely prune suboptimal portions of the search tree. Algorithm 2 works as follows. It takes as input three parameters: (i) the current allocation \(\theta\), (ii) the number of slots currently allocated and (iii) the best value of \(SW\) found so far. It recursively constructs a feasible allocation adding an ad at each recursive call until all slots are occupied. At first, it is invoked with parameters \((\emptyset, 1, 0)\). At Step 1 of each recursive invocation, it temporarily stores the current (partial) allocation \(\theta\) as the best one \(\theta^*\). If the base case of the recursion is reached (i.e., \(l > K\)), then all the slots have been filled and \(\theta^*\) is returned. Otherwise, at Step 3 the algorithm chooses an ad and, at Step 4, allocates it to slot \(s_l\) (the first empty slot in \(\theta\)). At Step 5, it computes, by means of the bounding function \(bf(\theta)\) (described later), an upper bound to the \(SW\) obtainable by completing the current allocation \(\theta\). If the so-computed upper bound does not exceed the \(SW\) of the best allocation found so far \((\text{best}\ SW)\), the current branch of the search tree is pruned. Otherwise the algorithm continues allocating the other empty slots recursively (Step 6).

We now define function \(bf(\theta)\). Consider a partial allocation \(\theta\) that allocates \(l < K\) slots, where \(\mathcal{L} \subseteq \mathcal{N}\) is the set of indices of the already allocated ads. For the sake of notational conciseness, let: (i) \(\Psi = \max_{i,j \in \mathcal{N}} \{\gamma_{i,j}\}\); (ii) \(M_{(i)}(\mathcal{L})\) denote the \((i)\)-th highest \(q_i \cdot v_i\) value for \(i \notin \mathcal{L}\). We adopt a different bounding function depending on reset model. We identify with \(\text{CFNE-tr}\) the CFNE problem with total reset. The value of \(bf(\theta)\) for each reset model is given in the following:

\[
bf(\theta) = \begin{cases} 
\sum_{m=l+1}^K \Lambda_m \Gamma_{\alpha(l;\theta)}(\theta) \Psi^{m-l} M_{(m-l)}(\mathcal{L}) & (\text{nr}) \\
\sum_{m=l+1}^K \Lambda_m M_{(m-l)}(\mathcal{L}) & (\text{tr}) \\
\sum_{m=l+1}^K \Lambda_m \Gamma_{\alpha(l;\theta)}(\theta) M_{(m-l)}(\mathcal{L}) & (\text{r}) 
\end{cases}
\]  

Finally, it is easy to check that: (i) \(bf(\theta)\) can be computed in polyno-
mial time with complexity $O(N \log N)$ and (ii) it actually returns an upper bound of the SW value obtainable by completing the partial allocation $\theta$.

\begin{algorithm}
\caption{\texttt{fEAlloc($\theta, l, \text{bestSW}$)}
\begin{algorithmic}[1]
\State $\theta^* = \theta$
\If{$l \leq K$} 
\Forall{$i \neq \alpha(m; \theta) \ \forall m < l$} 
\State $\alpha(l; \theta) = i$
\EndFor
\If{$\sum_{j \in N} CTR_j(\theta) \cdot v_j + \Gamma_{\alpha(l; \theta)}(\theta) \cdot bf(\theta) > \text{bestSW}$} 
\State $(\theta', \text{bestSW}') = \texttt{fEAlloc}(\theta, l + 1, \text{bestSW})$
\If{$\text{bestSW}' > \text{bestSW}$} 
\State $\text{bestSW} = \text{bestSW}'$
\EndIf
\EndIf
\EndIf
\Return $(\theta^*, \text{bestSW})$
\end{algorithmic}
\end{algorithm}

5.2.1 Efficient Algorithm for CFNE$_{aa}(K)$–nr Easy Instances

We now identify a significant class of instances of CFNE$_{aa}(K)$–nr for which we can design a polynomial-time optimal algorithm. These instances are characterized by the fact that the underlying contextual graph is a DAG, thus modelling nearly monopolistic markets in which the ads can be organized hierarchically. The idea of Algorithm 3 is that since DAGs can be sorted topologically in polynomial time, then we can rename the ads as $a_1, \ldots, a_N$ so to guarantee that for any pair of ads $a_i, a_j$, if $i < j$ then $(j, i) \notin E$. We can then prove that we can focus w.l.o.g. on \textit{ordered} allocations $\theta$, i.e., for any pair of allocated ads $a_i, a_j$, with $i < j$, $\theta(a_i) \leq \theta(a_j)$. Consider an unordered $\theta$ and let $a_i$ be the first ad (from the top) for which there exists $a_j, i < j$, such that $\pi(i; \theta) > \pi(j; \theta)$. Since $\gamma_{j,i} = 0$ then all the ads $a_k$ s.t. $\pi(k; \theta) \geq \pi(i; \theta)$ have $CTR_k(\theta) = 0$ and, therefore, we can prune $\theta$ of (i.e., substitute with $a_{\perp}$) $a_i$ and all the subsequent ads without any loss in the social welfare. But then in the class of ordered allocations, the optimum has an optimal substructure and we can use dynamic programming. Let $D[i, m]$ be the value of the optimal ordered allocation that uses only slots $s_m, \ldots, s_K$ and allocates ad $a_i$ in $s_m$. It is not hard to see that $D[i, m] = \Lambda_m g_i v_i + \max_{j > i} \gamma_{i,j} D[j, m + 1]$ and that the optimum is $\max_{i \in N} D[i, 1]$. In the pseudo-code of the algorithm, we simply construct the table $D$ after the topological sort of the contextual graph (with renaming

62
5.3. CFNE$_{aa}(K)$–nr is poly–APX hard

of the ads) is done. Specifically, in Steps 2–3, the algorithm computes the values of the last row of $D$, then in Steps 5–6 of the last column. Finally, in Steps 8–10, following the rule $D[i, m] = \max_{j>i} \Lambda_m q_i v_i + \gamma_{i,j} \cdot D[j, m + 1]$, the algorithm computes the values of all the other elements, starting from the row corresponding to ad $a_{N-1}$ till the first row is reached. The algorithm terminates identifying the optimal allocation as the one with maximum values among the ones stored in the first column.

Since the topological sort takes time $O(N^2)$, the whole algorithm runs in time $O(KN^2)$.

Algorithm 3

1: TOPOLOGICALSORT($G$)
2: for all $m \leq K$ do
3: $D[N, m] = \Lambda_m q_N v_N$
4: end for
5: for all $i \leq N$ do
6: $D[i, K] = \Lambda_K q_i v_i$
7: end for
8: for $i = N - 1$ to 1 do
9: for $m = K - 1$ to 1 do
10: $D[i, m] = \Lambda_m q_i v_i + \max_{j>i} \gamma_{i,j} D[j, m + 1]$
11: end for
12: end for
13: return $(\max_{i\in N} D[i, 1])$

Since social welfare maximization is a utilitarian problem, and given that the algorithm above is optimal we can use the celebrated VCG mechanism to obtain a polynomial-time optimal truthful mechanism.

5.3 CFNE$_{aa}(K)$–nr is poly–APX hard

In Sec. 5.1 we showed that CFNE$_{aa}(K)$–nr is NP–hard, but in this section we provide a stronger result.

Theorem 11. CFNE$_{aa}(K)$–nr is poly–APX–hard.

Proof. We reduce from the Longest Path problem. An instance of the Longest Path problem consists of a direct graph $G' = (T, A)$ where $T$ is the set of vertices of the graph and $A \neq \emptyset$ is the set of unweighed edges. The problem demands to compute a longest simple path, i.e., a maximum length path that visits each vertex of the graph at most once. This problem is poly–APX–complete [15] and the best known asymptotic approximation is $\frac{\log |T|}{|T|}$.
From an instance $G' = (T, A)$ of Longest Path we obtain an instance of CFNE$_{aa}(K)$–nr as follows. For each vertex $t_i \in T$ we add an ad $a_i$, with $q_i = v_i = 1$ and for each directed arc $(t_i, t_j) \in A$ we add an arc $(i, j)$ in $E$. Furthermore, we set $\gamma_{i,j} = 1$ if $(i, j) \in E$ and $\gamma_{i,j} = 0$ otherwise. Finally, we set $N = K = |T|$ and $\Lambda_m = 1, \forall m \in K$.

Given an ordered sequence of vertices $\rho = (t_1, t_2, \ldots, t_N)$, we denote as $\text{len}(\rho)$ the length of the path that starts in $t_1$ and visits the nodes in $\rho$ till the first node $t_j$ s.t. $(t_j, t_{j+1}) \not\in A$ is reached. Let us denote as $\rho^*$ the sequence that describes the longest path in $G'$ and as $\theta^*$ the allocation that maximizes the social welfare in the instance of CFNE$_{aa}(K)$–nr defined upon $G'$. It is easy to check that $\text{len}(\rho^*) = \text{SW}(\theta^*, v) - 1$. Indeed, $\theta^*$ allocates sequentially from the first slot the ads that correspond to the vertices composing the longest path. Conversely, we can transform an allocation $\theta$ into a sequence of vertices $\rho$ just by substituting the ads with their corresponding vertices until the first $a_\bot$ in $\theta$ is found. Thus, we have that for $\theta$ and the corresponding $\rho$ it holds $\text{len}(\rho) = \text{SW}(\theta, v) - 1$.

Let us consider a generic $\alpha$-approximate allocation $\theta_\alpha$ for CFNE$_{aa}(K)$–nr, i.e., $\text{SW}(\theta_\alpha, v) \geq \alpha \text{SW}(\theta^*, v)$. Note that since $A$ is non-empty, there is a solution $\theta_2$ to CFNE$_{aa}(K)$–nr of social welfare at least 2. In the following, we let $\theta_\beta$ denote the solution in $\{\theta_\alpha, \theta_2\}$ with maximum social welfare. Note that since $\theta_\alpha$ is an $\alpha$-approximate solution so is $\theta_\beta$. By letting $\rho_\beta$ denote the path constructed from $\theta_\beta$ as described above, we prove that the reduction preserves the approximation (up to a constant factor):

$$\text{len}(\rho_\beta) = \text{SW}(\theta_\beta, v) - 1 \geq \frac{1}{2} \text{SW}(\theta_\beta, v) \geq \frac{\alpha}{2} \text{SW}(\theta^*, v) = \frac{\alpha}{2} (\text{len}(\rho^*) + 1) \geq \frac{\alpha}{2} \text{len}(\rho^*).$$

### 5.4 CFNE$_{aa}(K)$–nr is poly–APX complete

We now show that the problem is in poly–APX, indeed there exists an algorithm with an approximation ratio that is asymptotically the same of the Longest Path. Our approximation algorithm combines the color coding algorithm [4] together with three approximation steps.

The CC algorithm we present here is different with respect to the one described in Sec. 4.3.1. More precisely, we use a different data structure in which we store the partial allocations we generate during the dynamic programming procedure. Both data structures are equivalent, in the sense that we can store the same allocations, but, while the data structure used here allows us to design a polynomial-time approximation algorithm, the
data structure used in Sec. 4.3.1 does not.

Going into the details of the algorithm, there is set $C$ that contains $K$ different colors. To find the best colorful allocation, given a random coloring we do the following. For $S \subseteq C$, we define $(S, a_i)$ as the set of partial allocations with the properties of having the same number $|S|$ of allocated ads (each colored with a different color of $S$) in the first $|S|$ slots and having ad $a_i$ in slot $s_{|S|}$. We start from $S = \emptyset$ where no ad is allocated. Then, allocating one of the ads in the first position, we add one color to $S$ until $S = C$. Iteratively, the algorithm extends the allocations in $(S, a_i)$ appending a new ad, say $a_j$, with a color not in $S$ in slot $s_{|S|+1}$ obtaining $(S \cup \text{col}(a_j), a_j)$ where we recall $\text{col}(a_j)$ returns the color of $a_j$. Each partial allocation in $(S, a_i)$ is characterized by the values of $SW$ and $\Gamma_i$. We can safely discard all the Pareto dominated partial allocations: given two allocations $\theta_1$ and $\theta_2$ in $(S, a_i)$, we say that $\theta_2$ is Pareto dominated by $\theta_1$ iff $SW(\theta_1) \geq SW(\theta_2)$ and $\Gamma_i(\theta_1) \geq \Gamma_i(\theta_2)$. However, there is no guarantee that the number of allocations in $(S, a_i)$ is polynomially bounded and, in principle, all the generated $O(N^K)$ partial allocations may be Pareto efficient. Notice that, differently, when applied on the Longest Path, we can safely store only one solution per table entry $(S, v)$, $v$ denoting a vertex of the graph, given that each solution is characterized only by a single attribute (the path length) and we can discard all the solutions except the longest. Thus we store at most $O(2^K)$ paths.

The complexity per coloring is $O(2^K N^{K+1} K^2)$. CC generates $e^K$ random colorings, but it can be derandomized with a cost of $\log^2(N)$ and a total complexity $O((2e)^K K^2 N^{K+1}(\log N)^2)$.

**Approximation 1.** We apply CC over a reduced number $K'$ of slots, where $K' = \min(\lceil \log(N) \rceil, K)$, implying the following approximation ratio.

**Proposition 7.** Given $\theta^*$, the optimal allocation over $K$ slots, and $\theta^*_{K'}$, the optimal allocation over the first $K' \leq \min\{N, K\}$ slots, we have $SW(\theta^*_{K'}, v) \geq \frac{1}{2 \min\{N, K\}} SW(\theta^*, v)$. 

**Proof.** Partition $K'' = \min\{N, K\}$ slots in groups of $K'$ consecutive slots. There could be remaining slots that will constitute the last group with less then $K'$ slots. The number of groups in which the $K$ slots are divided is $NG = \lceil \frac{K''}{K'} \rceil$. Call $G_i = \{(i-1)K' + 1, \ldots, \min(iK', K)\}$, for $i \in NG = \{1, \ldots, NG\}$, the $i$-th group of indices of $K'$ slots.

We let $SW(\theta, v|G_i) = \sum_{m \in G_i} \Lambda_m \Gamma_{\alpha(m; \theta)}(\theta) q_{\alpha(m; \theta)} v_{\alpha(m; \theta)}$, for any $\theta \in \Theta$. Since $SW(\theta^*, v) = \sum_{i=1}^{NG} SW(\theta^*, v|G_i)$, there must exist a group $G_i$ s.t. $SW(\theta^*, v|G_i) \geq \frac{1}{NG} SW(\theta^*, v)$. Observing that $\lceil \frac{K''}{K'} \rceil \leq \frac{K''}{K'} + 1$ and
Chapter 5. CFNE_{aa}(K)

\( K' \leq K'' \) we get \( SW(\theta^*, v|G_i) \geq \frac{K'}{2K''}SW(\theta^*, v) \). The proof concludes by noting that, by optimality, \( SW(\theta^*_{K''}, v) \geq SW(\theta^*, v|G_i) \).

Approximation 2. In CC, we discard allocations \( \theta \) in which \( \Gamma_i(\theta) \) of the last allocated ad \( a_i, i \in N \), is less than a given \( \delta \in [0, 1] \), implying the following approximation ratio.

**Proposition 8.** Given \( \theta^*_{K''} \), the optimal allocation over \( K' \) slots, and \( \theta^\delta_{K''} \), the optimal allocation among the allocations \( \theta \in \Theta \) where the last allocated ad \( a_i, i \leq N \), satisfies \( \Gamma_i(\theta) \geq \delta \), we have \( \Delta(\theta^\delta_{K''}, v) \geq (1 - \delta) SW(\theta^*_{K''}, v) \).

**Proof.** Consider the allocation \( \theta^*_{K''} \) and assume that the last ad satisfying \( \Gamma_i(\theta^*_{K''}) \geq \delta \) is the one in slot \( s_i \). Recalling the notation \( SW(\theta, v|S) \) for \( S \subseteq K \), provided in the proof of Proposition 7, by optimality of \( \theta^*_{K''} \), we have \( SW(\theta^*_{K''}, v) \geq \frac{1}{\Gamma^*_{K''}(l+1)}SW(\theta^*_{K''}, v|[l+1, \ldots, K]) \). Indeed, on the r.h.s. we have a lower bound on the social welfare that the ads allocated by \( \theta^*_{K''} \) in slots \( s_{l+1}, \ldots, s_{K''} \) would have if shifted to the first slot. If this were bigger than \( SW(\theta^*_{K''}, v) \) then \( \theta^*_{K''} \) would not be optimal. But then since \( \Gamma^*_{K''}(l+1) < \delta \), we have \( \delta SW(\theta^*_{K''}, v) \geq SW(\theta^*_{K''}, v|[l+1, \ldots, K]) \).

Finally we have that \( \theta^\delta_{K''} \), the allocation that removes from \( \theta^*_{K''} \) the ads allocated from \( s_{l+1} \) to \( s_{K''} \), has \( SW(\theta^\delta_{K''}, v) = SW(\theta^*_{K''}, v) - SW(\theta^*_{K''}, v|[l+1, \ldots, K]) \geq SW(\theta^*_{K''}, v) - \delta SW(\theta^*_{K''}, v) = (1 - \delta)SW(\theta^*_{K''}, v) \).

Approximation 3. In CC, we use rounded values for \( \gamma_{i,j} \). More precisely, we use \( [\frac{1}{\tau} \log \frac{1}{\gamma_{i,j}}] \) in place of \( \log \frac{1}{\gamma_{i,j}} \), where the normalization constant \( \tau \) is defined below. The constraint due to Proposition 8 is now a capacity constraint of the form \( \sum_{m \in K : m < \frac{1}{\tau} \log \frac{1}{\gamma_{i,j}}} \frac{1}{m^{\alpha(m, \theta), \alpha(m+1, \theta)}} \leq \frac{1}{\tau} \log \frac{1}{\delta} \). Notice that, with rounded values, the capacity can assume a finite number of values (i.e., \( [\frac{1}{\tau} \log \frac{1}{\delta}] \)) and therefore we can now bound the number of allocations to be stored in \( (S, a_i) \). More precisely, for each value of capacity, we can discard all the allocations except one maximizing the social welfare measured with rounded values. This step has the following consequences on the approximation guarantee.

**Proposition 9.** Given \( \theta^\delta_{K''} \), defined as in Proposition 8 and \( \theta^{\delta\epsilon}_{K''} \), the optimal allocation when the rounding procedure is applied, we have that, choosing \( \tau = \frac{1}{K''} \log \frac{1}{1-\epsilon} \), \( SW(\theta^{\delta\epsilon}_{K''}, v) \geq (1 - \epsilon) SW(\theta^{\delta}_{K''}, v) \).

**Proof.** Let \( \xi^x_{m, m+1} \) be a shorthand for \( \log \frac{1}{\gamma_{\alpha(m, \theta^x_{K''}), \alpha(m+1, \theta^x_{K''})}} \) and \( x(i) \) be a shorthand for \( \pi(i; \theta^x_{K''}) \), for \( x \in \{\delta, \delta\epsilon\} \). By definition:

\[
SW(\theta^{\delta\epsilon}_{K''}, v) = \sum_{i \in N} \Lambda_{\delta(i)} \Gamma_i(\theta^{\delta\epsilon}_{K''}) q_i v_i
\]
5.4. CFNE\textsubscript{\textit{aa}}(K)–nr is poly–APX complete

\[
= \sum_{i \in N} \Lambda_{\delta(i)} \prod_{m < \delta(i)} 2^{-\frac{\epsilon m}{m+1} q_i v_i}.
\]

Since \(\xi_{\delta m,m+1} \leq \tau(|\frac{1}{\tau}\xi_{\delta m,m+1}| + 1)\), we then have

\[
SW (\theta_{K'}^{\delta\epsilon}, v) \geq \sum_{i \in N} \Lambda_{\delta(i)} \prod_{m < \delta(i)} 2^{-\tau(|\frac{1}{\tau}\xi_{\delta m,m+1}| + 1)} q_i v_i
\]

\[
\geq \sum_{i \in N} \Lambda_{\delta(i)} \prod_{m < \delta(i)} 2^{-\tau(|\frac{1}{\tau}\xi_{\delta m,m+1}| + 1)} q_i v_i,
\]

where the latter inequality follows from optimality of \(\theta_{K'}^{\delta\epsilon}\). Given that \(|y| \leq y\) we can conclude that \(SW (\theta_{K'}^{\delta\epsilon}, v)\) is bounded from below by:

\[
\sum_{i \in N} \Lambda_{\delta(i)} \left( \prod_{m < \delta(i)} 2^{\log \gamma_{\alpha(m;\theta_{K'}^{\delta\epsilon})} - \tau} \right) q_i v_i
\]

\[
\geq 2^{-K'\tau} \cdot \sum_{i} \Lambda_{\delta(i)} \Gamma_i (\theta_{K'}^{\delta\epsilon}) q_i v_i
\]

\[
= (1 - \epsilon) \cdot \sum_{i} \Lambda_{\delta(i)} \Gamma_i (\theta_{K'}^{\delta\epsilon}) q_i v_i = (1 - \epsilon) SW (\theta_{K'}^{\delta\epsilon}, v).
\]

This concludes the proof. \(\square\)

The approximation ratio of the algorithm is thus \((1 - \delta)(1 - \epsilon)\frac{\log(N)}{2 \min\{N,K\}}\), asymptotically the same as the best known approximation ratio of the Longest Path once \(N = K\). The complexity instead can be derived as follows. The maximum number of allocations that can be stored in each \((S, a_i)\) is \(O\left(\frac{\log(1)}{\tau}\right)\) with \(\tau = \frac{\log(1)}{K'\epsilon} \) thanks to dominations. Thus, given that \(\log(\frac{1}{1-\epsilon}) \rightarrow 0\) as \(\epsilon \rightarrow 0\), the number of elements is \(O(K'\frac{1}{\epsilon})\). Thus, the complexity when \(K' = \log(N)\) is \(O((2e)^{\log(N)} \frac{1}{\epsilon} \log(\frac{1}{\delta}) N^2 \log^4(N)) = O(\frac{1}{\epsilon^2} N^3 \log^4(N)).\)

Notice that all the three above approximations are necessary in order to obtain a polynomial–time algorithm. Approximation 2 and Approximation 3 allow us to bound the number of the allocations stored per pair \((S, a_i)\) and would lead, if applied without Approximation 1, to a complexity \(O((2e)^K K^2 N^2 \log^2(N) \frac{1}{\epsilon^2})\). Notice also that, without Approximation 2, the possible values for the capacity are not upper bounded. Approximation 1 allows us to remove the exponential dependence on \(K\) and to obtain polynomial complexity.
Non–monotonicity Unluckily, the approximation algorithm we have just introduced cannot be adopted in a DSIC mechanism.

**Proposition 10.** The approximation algorithm is non–monotone.

**Proof.** Let us initially consider the case where Approximation 1 is not used, therefore all the \( K \) slots can be allocated. We will discuss below how to extend the proof to the case where Approximation 1 is used.

Consider the following instance of CFNE\(_{aa}(K)\)–nr:

- \( K = 3 \) slots;
- \( N = 4 \) ads, where \( q_1v_1 = 2^{2\tau} \frac{\Lambda_2 - \Lambda_3 2^{-6\tau}}{\Lambda_2 - \Lambda_3} + 3, q_2v_2 = x, q_3v_3 = q_4v_4 = 1 \), where \( \tau \) is the generic rounding factor of Approximation 3;
- the contextual graph is s.t. \( \gamma_{i,j} = 0 \) \( \forall i,j \in N \) except: \( \gamma_{1,2} = 2^{(-4+\phi)\tau}, \gamma_{1,3} = 2^{-\tau}, \gamma_{2,4} = 2^{-\tau}, \gamma_{3,2} = 2^{-\tau}. \phi \) is a small number;
- the rounded capacity \( \left\lfloor \log \frac{1}{\gamma_{1,2}} \right\rfloor = +\infty \forall i,j \in N \) except: \( \left\lfloor \log \frac{1}{\gamma_{1,3}} \right\rfloor = 3, \left\lfloor \log \frac{1}{\gamma_{2,4}} \right\rfloor = 1, \left\lfloor \log \frac{1}{\gamma_{3,2}} \right\rfloor = 1. \)
- the \( K \) colours are \( \{o_1, o_2, o_3\} \).

The product \( q_1v_1 \) has been chosen s.t., when \( x \) is in the neighbourhood of \( 2^{2\tau} \frac{\Lambda_2 - \Lambda_3 2^{-4\tau}}{\Lambda_2 - \Lambda_3} \), \( a_1 \) is always allocated in the first slot. Thus, we can focus only on the colouring that assigns colour \( o_1 \) to \( a_1, o_2 \) to \( a_2 \) and \( o_3 \) to \( a_3 \) and \( a_4 \). Indeed, with this colouring the two longest path of the contextual graph are colourful, i.e., the unique two colourful allocations are \( \theta_1 = (a_1, a_3, a_2) \) in the set \( \{\{o_1, o_2, o_3\}, a_2\} \) and \( \theta_2 = (a_1, a_2, a_4) \) in the set \( \{\{o_1, o_2, o_3\}, a_4\} \).

Notice that, with this colouring, all the allocations where there is a pair of ads \((a_i, a_j)\) with \( \gamma_{i,j} = 0 \) are infeasible, not satisfying the capacity bound. We will now prove that the approximation algorithm is not monotone with respect to \( a_2 \).

Let us denote by \( \tilde{SW} \) the social welfare computed on the basis of the rounded values. It is easy to check that the following hold: \( \tilde{SW}(\theta_1) = 2^{2\tau} \frac{\Lambda_2 - \Lambda_3 2^{-6\tau}}{\Lambda_2 - \Lambda_3} + 3 + \Lambda_2 2^{-4\tau} x + \Lambda_3 2^{-6\tau} \) and \( \tilde{SW}(\theta_2) = 2^{2\tau} \frac{\Lambda_2 - \Lambda_3 2^{-6\tau}}{\Lambda_2 - \Lambda_3} + 3 + \Lambda_2 2^{-\tau} + \Lambda_3 2^{-4\tau} x \). Notice that the rounded \( CTR_2 \) in \( \theta_2 \) is always greater than the one in \( \theta_1 \), given \( \Lambda_2 \geq \Lambda_3 \), while \( CTR_2(\theta_1) = \Lambda_3 2^{-2\tau} > \Lambda_2 2^{(-4+\phi)\tau} = CTR_2(\theta_2) \) when \( \frac{\Lambda_2}{\Lambda_3} < 2^{2\tau-\phi\tau} \).

We have that \( \tilde{SW}(\theta_1, v) > \tilde{SW}(\theta_2, v) \) when \( x > 2^{2\tau} \frac{\Lambda_2 - \Lambda_3 2^{-4\tau}}{\Lambda_2 - \Lambda_3} \). Thus \( a_2 \) gets a lower CTR by increasing her bid, which proves that the algorithm is not monotone.
The example can be extended also to the case where Approximation 1 is applied introducing ads with \( q v = 0 \) and \( \gamma_{i,j} = 0 \), s.t. \( \log N = K \).

5.5 A Monotone Algorithm for CFNE\(_{aa}(K)\)–nr

It is not unusual that there is a gap between the best approximation and the best monotone approximation. A simple \( 1/K \)-approximate truthful mechanism can be obtained through an application of the single-item second price auction. Specifically, we allocate the ad with the maximum \( q_i v_i \), guaranteeing a \( 1/K \)-approximation of the optimum, and charge the winner the second highest product.

Anyway, even if we do not improve the worst-case bound, we can provide better allocations still in polynomial time. Here we define a family of monotone algorithms that return an approximation of \( O(1/K) \), loosing a factor of \( \log(N) \) w.r.t. the best, non-monotone, known approximation. Our algorithms are based on the algorithm to find the optimal solution for CFNE\(_{aa}(K)\)–nr when \( G \) is a DAG, see Sec. 5.2.1. The basic idea is to fix an order \( \succ \) among the ads and rename the ads s.t. \( j > i \) iff \( a_i \succ a_j \). Then, the algorithm removes all the edges in \( E \) except the edges \((i,j)\) where \( i < j \). In this way we have a new graph \( G' = (\mathcal{N}', \mathcal{E}') \) that is a DAG. Hence, we can apply Algorithm 3 from Step 2. In order to preserve the monotonicity of the algorithm, \( \succ \) must not depend on the reports \( \hat{v}_i \) of the agents.

Here some \( \succ \) that can be adopted:

- \( \succ_q \) \( a_i \succ a_j \) iff \( q_i \geq q_j \);
- \( \succ_{\gamma_{\max}} \) \( a_i \succ a_j \) iff \( \max_{z \in \mathcal{N} \setminus \{i\}} \gamma_{iz} \geq \max_{z \in \mathcal{N} \setminus \{j\}} \gamma_{jz} \);
- \( \succ_{\gamma} \) \( a_i \succ a_j \) iff \( \sum_{z \in \mathcal{N} \setminus \{i\}} \gamma_{iz} \geq \max_{z \in \mathcal{N} \setminus \{j\}} \gamma_{jz} \);
- \( \succ_{\frac{q}{1-\gamma_{\max}}} \) \( a_i \succ a_j \) iff \( \frac{q_i}{1-\max_{z \in \mathcal{N} \setminus \{i\}} \gamma_{iz}} \geq \frac{q}{1-\max_{z \in \mathcal{N} \setminus \{j\}} \gamma_{jz}} \);
- \( \succ_r \) a random order among the ads is chosen.

It is easy to see that in the worst case, this family of algorithms provides the same solution of the algorithm allocating the ad with the maximum \( q_i v_i \).

Choosing an order that does not depend on the declaration of the advertiser, the algorithm selects a subset of feasible allocations \( \Theta' \subset \Theta \) and returns \( \theta^* = \arg \max_{\theta' \in \Theta'} SW(\theta, \hat{v}) \). Thus the algorithm is maximal in the range and a truthful mechanism is obtained imposing VCG-like payments [45].
5.6 \( \text{CFNE}_{aa}(K) \)–\{r, tr\} is APX–hard

In this section we will prove the APX–hardness of \( \text{CFNE}_{aa}(K) \)–\{r, tr\}, but first we have to highlight the relation holding between the two models: \( \text{CFNE}_{aa}(K) \)–r and \( \text{CFNE}_{aa}(K) \)–tr.

Let us define the following class of instances of \( \text{CFNE}_{aa}(K) \)–\{r, tr\}.

**Definition 8.** \( \mathcal{B}–\text{CFNE}_{aa}(K) \)–\{r, tr\} is the sub-problem of \( \text{CFNE}_{aa}(K) \)–\{r, tr\} where allowed instances satisfy (i) \( \Lambda_m = 1, \forall m \in K \); (ii) \( q_i = v_i = 1, \forall i \in N \) and (iii) \( \gamma_{i,j} = \{0,1\} \) for all \( i, j \in N \).

Since we use instances of \( \mathcal{B}–\text{CFNE}_{aa}(K) \)–\{r, tr\} in the APX–hardness with the same reduction from ATSP, the same argument proves the APX–hardness of both \( \text{CFNE}_{aa}(K) \)–r and \( \text{CFNE}_{aa}(K) \)–tr, thanks to similarities between the two models, as shown in the following.

For notational convenience, we denote as \( SW^{tr}(\theta, v) \) and \( SW^{r}(\theta, v) \) the value of the allocations \( \theta \) under the total reset and reset model, respectively.

**Proposition 11.** Let \( \theta = (a_1, \ldots, a_K) \) be an allocation such that \( \gamma_{i,i+1} = 0 \) and \( a_{i+1} \neq a_{\bot} \) for some \( i \in \{1, \ldots, K-1\} \). Let \( \theta' \) be the allocation obtained from \( \theta \) by substituting \( a_{i+1} \) with \( a_{\bot} \). Then \( SW^{tr}(\theta', v) \geq SW^{tr}(\theta, v) \) and \( SW^{r}(\theta', v) \geq SW^{r}(\theta, v) \).

**Proof.** Consider \( \text{CFNE}_{aa}(K) \)–tr. We have that \( \Gamma_i(\theta)q_iv_i = 1, \Gamma_{i+1}(\theta)q_{i+1}v_{i+1} = 0 \) and \( \Gamma_z(\theta) = 0 \) \( \forall z \) s.t. \( i + 1 < z \leq K \). Let us consider the allocation \( \theta' \) obtained from \( \theta \) by substituting \( a_{i+1} \) with \( a_{\bot} \), for all \( i \) such that \( \gamma_{i,i+1} = 0 \) and \( a_{i+1} \neq a_{\bot} \). It is easy to check that \( \Gamma_z(\theta') \geq \Gamma_z(\theta) \) \( \forall z \) s.t. \( i + 1 < z \leq K \). Thus, \( SW^{tr}(\theta', v) \geq SW^{tr}(\theta, v) \).

Consider \( \text{CFNE}_{aa}(K) \)–r. Given an allocation \( \theta = (a_1, \ldots, a_K) \) and (\( a_i, a_j \)) the first pair where \( \gamma_{i,j} = 0 \), we have that \( \Gamma_i(\theta)q_iv_i = 1, \Gamma_j(\theta)q_jv_j = 0 \) and \( \Gamma_z(\theta) = 0 \) \( \forall z > j \), \( \forall z \) s.t. \( i + 1 < z \leq K \). Substituting \( a_i \) with \( a_{\bot} \) we obtain \( \Gamma_j(\theta')q_jv_j = 1 \) and \( \Gamma_z(\theta') \geq \Gamma_z(\theta) \) \( \forall z \) s.t. \( i + 1 < z \leq K \). Thus, \( SW^{r}(\theta', v) \geq SW^{r}(\theta, v) \). \( \square \)

We highlight that Proposition [11] allows us to restrict our attention to allocations where no pair of adjacent non-fictitious ads \( a_i, a_{i+1} \) exists such that \( \gamma_{i,i+1} = 0 \).

**Proposition 12.** For the \( \mathcal{B}–\text{CFNE}_{aa}(K) \)–\{r, tr\} problem, \( SW^{tr}(\theta, v) = SW^{r}(\theta, v) \) holds for all allocations \( \theta \) that do not contain a pair of consecutive ads \( (a_i, a_j) \) s.t. \( \gamma_{i,j} = 0 \).

The proof easily follows from the definitions of \( SW^{tr} \) and \( SW^{r} \).
5.6. CFNE\(_{aa}(K)\)–\(\{r, tr\}\) is APX–hard

**Proposition 13.** CFNE\(_{aa}(K)\)–\(\{r, tr\}\) is APX–hard. More specifically, CFNE\(_{aa}(K)\)–\(\{r, tr\}\) cannot be approximated within a factor of \(\frac{1}{1+\alpha}\), for \(\alpha < \frac{1}{412}\), unless \(P = NP\).

**Proof.** As proved above in Propositions 11 and 12 the two models \(B\)–CFNE\(_{aa}(K)\)–\(tr\) and \(B\)–CFNE\(_{aa}(K)\)–\(r\) are equivalent for the class of instances used in the following proof, therefore these arguments apply to both as well.

We reduce from the Asymmetric TSP with weights in \(\{1, 2\}\), hereinafter denoted as \(ATSP(1, 2)\), to \(B\)–CFNE\(_{aa}(K)\)–\(\{r, tr\}\). The \(ATSP(1, 2)\) problem demands finding a minimum cost Hamiltonian tour in a complete directed weighted graph \(G' = (T, A)\) where \(T\) is the set of nodes of \(G'\), \(A\) is the set of edges and the weight function \(w_{i,j} \in \{1, 2\}\) for all edges \((i, j) \in A\). \(ATSP(1, 2)\) cannot be approximated in polynomial time within a factor of \(\frac{1}{1+\beta}\), with \(\beta < 1/206\) [34]. In the following, we will denote as \(\tau\) a solution of an \(ATSP(1, 2)\) instance, and as \(cost(\tau)\) its cost. The optimal tour will be denoted as \(\tau^*\).

Given an instance of \(ATSP(1, 2)\) on graph \(G' = (T, A)\) we construct an instance of \(B\)–CFNE\(_{aa}(K)\)–\(\{r, tr\}\) as follows: (i) for each vertex \(t_i \in T\) we generate an ad \(a_i\) with \(q_i = v_i = 1\), then we have \(N = |T|\); (ii) the contextual graph is \(G = (N', E)\), where \((i, j) \in E\) iff \(w_{i,j} = 1\); (iii) for all \((i, j) \in E, \gamma_{i,j} = 1\); and finally (iv) the number of slots is equal to the cost of the optimal tour \(\tau^*\) in \(ATSP(1, 2)\), i.e., \(K = cost(\tau^*)\). We will show at the end of the proof how we can deal with the fact that we do not know \(cost(\tau^*)\). Observe that with \(K = cost(\tau^*)\), we have \(SW(\theta^*, v) = N, \theta^*\) denoting the optimal solution of the CFNE\(_{aa}(K)\)–\(\{r, tr\}\) instance constructed. The definition of the reduction is completed by observing that an allocation \(\theta\) for the \(B\)–CFNE\(_{aa}(K)\)–\(\{r, tr\}\) that allocates all the \(N\) ads can be easily mapped back to a tour \(\tau\) for the \(ATSP(1, 2)\) by simply substituting the ad with the corresponding vertex of the graph \(G'\).

Let us suppose for the sake of contradiction that there exists a \(\frac{1}{1+\alpha}\)–approximate algorithm for \(B\)–CFNE\(_{aa}(K)\)–\(\{r, tr\}\), with \(\alpha < \frac{\beta}{2} < \frac{1}{412}\). Let \(\theta_\alpha\) be the \(\frac{1}{1+\alpha}\)–approximate solution returned by such an algorithm, i.e., \(SW(\theta_\alpha, v) \geq \frac{1}{1+\alpha}SW(\theta^*, v) = \frac{N}{1+\alpha}\). It is easy to check that \(\theta_\alpha\) consists of \(\frac{N}{1+\alpha}\) ads, each providing a contribution of 1 to the social welfare, while there are \(SW(\theta^*, v) - \frac{N}{1+\alpha}\) ads that w.l.o.g. we can consider empty. Moreover, being \(\alpha < 1\) \(\frac{N}{1+\alpha} \geq cost(\tau^*) - \frac{N}{1+\alpha}\) holds. For the sake of conciseness, hereinafter we omit the ceiling notation. Let \(\tau_\beta\) be the tour obtained from \(\theta_\alpha\). We state that in \(\tau_\beta\) there are, at least, \(\frac{2N}{1+\alpha} - cost(\tau^*) - 1\)
Chapter 5. CFNE\textsubscript{aa}(K)

edges of weight 1. Divide the ads allocated in \( \theta_\alpha \) in two sets: the \( \frac{N}{1+\alpha} \) allocated ads \( a_i \ i \in \mathcal{N} \) and \( a_\bot \). Allocate in alternation one of the \( \frac{N}{1+\alpha} \) ads \( a_i \), with \( i \in \mathcal{N} \), and one of the \( \text{cost}(\tau^*) \) \(-\frac{N}{1+\alpha} \) ads \( a_\bot \). When the slot index \( 2(\text{cost}(\tau^*) - \frac{N}{1+\alpha}) \) is reached, the available \( a_\bot \) are finished, thus, in the following \( \text{cost}(\tau^*) - 2(\text{cost}(\tau^*) - \frac{N}{1+\alpha}) = \frac{2N}{1+\alpha} - \text{cost}(\tau^*) \) slots, only non-fictitious ads \( a_i \ i \in \mathcal{N} \), are consecutively allocated (no slots are left empty). This means that in \( \theta_\alpha \), where the ads are disposed in a different way, we still have the guarantee that there are \( \frac{2N}{1+\alpha} - \text{cost}(\tau^*) \) \(-1 \) pairs of consecutive ads \((a_i,a_j)\) s.t. \( \gamma_{i,j} = 1 \). Thus, in the tour \( \tau_\beta \) there are, at least, \( \frac{2N}{1+\alpha} - \text{cost}(\tau^*) - 1 \) edges of weight 1. Therefore, given that a tour is composed of \( N \) edges, in \( \tau_\beta \) there can be at most \( N - \frac{2N}{1+\alpha} + \text{cost}(\tau^*) + 1 \) edges of weight 2. The length of \( \tau_\beta \) is upper-bounded by \( \text{cost}(\tau_\beta) \leq \frac{2N}{1+\alpha} - \text{cost}(\tau^*) - 1 + 2(N - \frac{2N}{1+\alpha} + \text{cost}(\tau^*) + 1) = \text{cost}(\tau^*) + \frac{2Na}{1+\alpha} + 1 \). Now we can state:

\[
\text{cost}(\tau_\beta) \leq \text{cost}(\tau^*) + \frac{2\alpha N}{1+\alpha} + 1 \leq \text{cost}(\tau^*) + 2\alpha N \\
\leq \text{cost}(\tau^*) + 2\alpha \text{cost}(\tau^*) = (1 + 2\alpha) \text{cost}(\tau^*) \\
< (1 + \beta) \text{cost}(\tau^*) ,
\]

where: (i) the second inequality holds for \( N \geq \frac{1+\alpha}{2\alpha^2} \); (ii) the third inequality holds since \( N \leq \text{cost}(\tau^*) \) and (iii) the last inequality holds since, by assumption, \( \alpha < \frac{\beta}{2} \). Thus, for the instances where \( N \geq \frac{1+\alpha}{2\alpha^2} \) if there were an algorithm that \( \frac{1}{1+\alpha} \)-approximates \( B\text{-CFNE}_{aa}(K)\)\{-r, tr\} with \( \alpha < \frac{1}{412} \), there would be a \( \frac{1}{1+\beta} \) approximation of \( ATSP(1,2) \) with \( \beta < \frac{1}{206} \). We obtained an absurd.

We finally show that we can deal with the non existence of the oracle returning \( \text{cost}(\tau^*) \). Notice that, for all the instances of \( ATSP(1,2) \) with \( N \) vertices, \( N \leq \text{cost}(\tau^*) \leq 2N \). Thus, given \( \text{cost}(\tau^*) \) is unknown we run the polynomial \( \frac{1}{1+\alpha} \)-approximation algorithm of \( \text{CFNE}_{aa}(K)\)\{-r, tr\} for all the values \( K = m \) with \( m \in \{N, \ldots, 2N\} \). For each execution of the algorithm we obtain a tour \( \tau_\beta^m \). We then choose \( \tau_\beta = \arg \min_{\tau_\beta^m : m \in \{N, \ldots, 2N\}} \text{cost}(\tau_\beta^m) \), thus it is guaranteed that \( \text{cost}(\tau_\beta) \leq \text{cost}(\tau_\beta^{\text{cost}(\tau^*)}) \).

5.7 CFNE\textsubscript{aa}(K)\{-r, tr\} is APX–complete

We can easily show that \( \text{CFNE}_{aa}(K)\)\{-r, tr\} is in APX. The simple greedy algorithm we introduce in Sec. 4.3.3 provides an approximation of \( \frac{1}{2} \) event when \( c = K \). For more details, see Sec. 4.3.3. The algorithm provides a
constant approximation w.r.t. to the optimal solution, thus, this proves that $\text{CFNE}_{aa}(K) - \{r, tr\}$ is APX–complete.

Moreover, the algorithm is monotone and MIR. Hence, it can be augmented with the (polynomial–time computable) VCG payments in order to obtain truthful mechanisms. We therefore have the following.

**Theorem 12.** There exists a $\frac{1}{2}$–approximate truthful polynomial-time mechanism for $\text{CFNE}_{aa}(K) - \{r, tr\}$. 
In this chapter we provide a specific model of externality that is suitable for Mobile Advertising scenarios in which a user is moving with a mobile device into a physical area. The physical area is divided in subareas and the user can receive ads on his mobile device only in these subareas. Our goal is to optimise the way the ads are sent to the user on the basis of the path the user follows. Differently from the two previous chapters, we focus mainly on the problem of designing an algorithm for finding the optimal solution, and identifying special cases for which computing the solution is a polynomial time task, leaving as future work a more accurate theoretical study of the computational hardness of the problem.

Specifically, we study two different settings: the single–path case, Fig. 6.1 (a), where every user visits the same sub areas in the same order and the multi–path case, Fig. 6.1(b), where different users can follow different paths. In the latter case, the set of paths starting with a specific subarea can be seen as a tree. For more details see Sec. 3.1.3, where we introduced the model.

For the single–path case, we first introduce an algorithm computing the optimal allocation, Sec. 6.1.1. Since the algorithm can require exponential time, we then identify instances of the problem where the optimal allocation
can be found in polynomial time, in Sec. 6.1.2. Finally, in Sec. 6.1.3 we
study a polynomial time approximation algorithm.

We follow the same steps also for the multi–path case: in Sec. 6.2.1 we
propose an exact algorithm, in Sec. 6.2.2 we identify easy instances and in
Sec. 6.2.3 an approximation algorithm.

Finally, we experimentally evaluate our algorithms in Sec. 6.3.

6.1 Single–Path Case

In this section, we study the basic case where the tree of all the possible
paths is restricted to a single path. We first present a mechanism with an
efficient allocation function \( f_E \), and then go on to present a mechanism
with an approximate allocation function \( f_A \). The main challenge of an ef-
ficient mechanism is finding the optimal advertising plan, i.e., the alloca-
tion of ads to nodes that maximises social welfare, subject to the constraint
that each ad \( a_i, i \in N \), can appear at most once on the (single) path (ex-
cept ad \( a_\perp \)). To this end, we show that this allocation problem is a varia-
tion of a known linear assignment problem (AP) as in [17]. In particular,
when the aggregated continuation probability, \( \Lambda_l \), is a constant (i.e., when
\( \lambda_l = 1 \forall l \in \{0, \ldots, K - 1\} \), which means that there are no externalities) the
single–path problem corresponds to the classical 2–index AP (2AP), where
the aim is to allocate a set of tasks to a set of agents while minimising/max-
imising the sum of costs/profits \( w \), subject to each agent having exactly one
task. In our problem, agents correspond to nodes \( K \), and tasks to ads \( N \).
Furthermore, since there is a single path, we have that \( \sigma_m = 1, \forall m \in K \).
Then, for \( \lambda_l = 1 \forall l \in K \), the value for allocating ad \( a_i \) to slot \( s_m \) is given

\[ f_E(a_i, s_m) = \begin{cases} 1 & \text{if } a_i \text{ is allocated to slot } s_m, \\ 0 & \text{otherwise}. \end{cases} \]

\[ f_A(a_i, s_m) = \begin{cases} \sigma_m & \text{if } a_i \text{ is allocated to slot } s_m, \\ 0 & \text{otherwise}. \end{cases} \]

\[ w(a_i, s_m) = \begin{cases} \lambda_i & \text{if } a_i \text{ is allocated to slot } s_m, \\ 0 & \text{otherwise}. \end{cases} \]

\[ \text{maximize } \sum_{i} \sum_s w(a_i, s) \text{ subject to } \sum_s f(a_i, s) = 1. \]

\[ \text{maximize } \sum_{i} \sum_s w(a_i, s) \text{ subject to } \sum_s f(a_i, s) \leq 1. \]
6.1. Single–Path Case

\[
\begin{array}{c|cc}
   & \hat{v}_1 \cdot q_{i,1} & \hat{v}_a \cdot q_{i,2} \\
\hline
   a_1 & 1 & 2 \\
a_2 & 2 & 4
\end{array}
\quad \begin{array}{c|cc}
   & \hat{v}_t \cdot q_{i,1} & \hat{v}_t' \cdot q_{i,2} \\
\hline
   a_1 & 3 & 6 \\
a_2 & 2 & 4
\end{array}
\]

Table 6.1: Two scenarios used as examples.

by the expected reward \( E[v_i|\alpha(m; \theta) = i] = w_{i,m} = \hat{v}_i q_{i,m} \) (to maximise).

Now, it is well known that the 2AP can be solved in polynomial time by means of the Hungarian algorithm \([36, 40]\)—with complexity \( O(\max\{K^3, N^3\}) \)—or linear programming (LP) as in \([17]\), where the continuous relaxation results in a basic integer solution. However, when \( \lambda_l < 1 \), the nature of our problem becomes fundamentally different. In particular, the 2AP optimal solution always requires that all the agents are assigned with a task. This also holds in our setting when \( \lambda_1 = 1 \forall l \in \{0, \ldots, K - 1\} \). However, when \( \lambda_l < 1 \), it can be optimal to leave some nodes unallocated, as shown in the following example:

**Example 1.** Consider the case represented in Table 6.1(a) where \( K = 2 \), \( N = 2 \). The optimal solution of the 2–index AP allocates \( a_1 \) in \( s_1 \), and \( a_2 \) in \( s_2 \) (in bold in the table). Instead, with \( \lambda_1 < 0.75 \), the optimal solution of our problem allocates only \( a_2 \) in \( s_2 \).

When \( \lambda_l < 1 \), our allocation problem can be formulated as a variation of the 3–index assignment problem (3AP) as:

\[
\max_{\theta \in \Theta} \sum_{i \in \mathcal{N}} \sum_{m \in \mathcal{K}} \sum_{l \in \mathcal{L}} \Lambda_l \cdot \hat{v}_i \cdot q_{i,m} \cdot x_{i,m,l}
\]

\[
\sum_{m \in \mathcal{K}} \sum_{l \in \mathcal{L}} x_{i,m,l} \leq 1 \quad \forall i \in \mathcal{N} \quad (6.1)
\]

\[
\sum_{i \in \mathcal{N}} \sum_{l \in \mathcal{L}} x_{i,m,l} \leq 1 \quad \forall m \in \mathcal{K} \quad (6.2)
\]

\[
\sum_{i \in \mathcal{N}} \sum_{m \in \mathcal{K}} x_{i,m,l} \leq 1 \quad \forall l \in \mathcal{L} \quad (6.3)
\]

\[
\sum_{i \in \mathcal{N}} x_{i,m,l} - \sum_{i \in \mathcal{N}} \sum_{m' \in \mathcal{K}, m' < m} x_{i,m',l-1} \leq 0 \quad \forall m \in \mathcal{K}, l \in \mathcal{L} \setminus \{0\} \quad (6.4)
\]

\[
x_{i,m,l} \in \{0, 1\} \quad \forall i \in \mathcal{N}, m \in \mathcal{K}, l \in \mathcal{L} \quad (6.5)
\]
where \( x_{i,m,l} = 1 \) if \( i = \alpha(m; \theta) \) (i.e., ad \( a_i \) is allocated to node \( s_m \)) and \( l = \mu(m; \theta) \) (i.e., \( a_i \) is the \( l + 1 \)-th allocated non-empty ad along the path); \( x_{i,m,l} = 0 \) otherwise. \( \mathcal{L} = \{0, \ldots, K - 1\} \) contains all the possible values of \( l \), indeed an ad can be preceded from 0 to \( K - 1 \) ads. Constraints (6.1) ensure that each ad \( a_i \neq a_{\perp} \) is allocated at most once; Constraints (6.2) ensure that each node is allocated to an ad \( a_i \neq a_{\perp} \) at most once; Constraints (6.3) ensure that there cannot be two ads with the same number of preceding ads (except for the empty ad); Constraints (6.4) ensure that, whenever \( x_{i,m,l} = 1 \) (i.e., if some ad \( a_i \neq a_{\perp} \) is allocated to a node \( s_m \) with \( l \) preceding ads), then \( l \) ads must be actually allocated in the path preceding \( s_m \).

Compared to the 3AP formulation, our problem has Constraints (6.4) as additional constraints. Moreover, our objective function is a special case of the 3AP objective function (the original 3AP function is given by \( \max \sum_{i \in \mathcal{N}} \sum_{m \in \mathcal{K}} \sum_{l \in \mathcal{L}} w_{i,m,l} \cdot x_{i,m,l} \)). The maximization version of 3AP is easily shown to be \( \mathcal{NP} \)-hard by reduction from the 3-dimentional matching problem (3DMP), however, there is no straightforward reduction from \( \mathcal{NP} \)-hard problems to ours.

Furthermore, we can show that the continuous relaxation of our allocation problem admits, differently from 2-index AP, non-integer optimal solutions and thus the above integer mathematical programming formulation cannot be solved (in polynomial time) by LP tools, as shown by the following example.

**Example 2.** We have \( N = 3, K = 3, \) and \( \lambda_l = 0.2 \forall l \). Parameters \( q_{i,m} \) and \( \hat{v}_i \) are: \( q_{1,3} = q_{2,1} = q_{3,2} = 1 \), while all the others \( q_{i,m} = 0 \), and \( \hat{v}_1 = 100 \), \( \hat{v}_2 = 79 \), \( \hat{v}_3 = 70 \). The optimal solution of the continuous relaxation of our allocation problem is: \( x_{1,3,0} = x_{1,3,1} = x_{2,1,0} = x_{3,2,1} = 0.5 \), with a social welfare of 106.5. Instead, the optimal integer solution is: \( x_{1,3,0} = 1 \), with a social welfare of 100.

### 6.1.1 Exact Algorithm

We start by considering the unrestricted setting. For this setting, any branch- and-bound algorithm enumerating all the allocations, e.g., using standard integer programming or [12] for 3AP, has a complexity of \( O(N^K) \) in the
6.1. Single–Path Case

worst case. We show that it is possible to have an algorithm for $f_E$ with a better complexity.

Our algorithm, named OptimalSinglePath, works as follows. First, we split the problem into subproblems. In detail, let $B \subseteq \mathcal{K}$ denote a set of the indices of nodes such that non–empty ads ($a_i \neq a_\perp$) are allocated into the nodes $s_m$, $m \in B$, and empty ads ($a_\perp$) into all the other nodes $s_m$, $m \notin B$. Note that there are exactly $2^K$ such combinations (assuming for the sake of simplicity that $N \geq K$). Now, for a given $B$, the number of nodes with non–empty ads preceding any $s_m$, $m \in B$, is fixed and does not depend on the specific feasible allocation $\theta$, thus we can denote it by $\mu(m; B)$ instead of $\mu(m; \theta)$. Then, the problem of finding the optimal allocation for a given $B$ can be formulated as an AP where $w_{i,m} = \hat{v}_i q_{i,m} \Lambda_{\mu(m;B)}$. This problem can be solved by using an AP–solving oracle with a complexity of $O(N^3)$. Our algorithm then calls the AP–solving oracle for each $B \subseteq \mathcal{K}$. Finally, the algorithm returns the best found allocation. The complexity is $O(2^K N^3)$.

Obviously, this algorithm, finding the allocation maximising the $SW$, can be adopted in VCG mechanism with Clarke pivoting in order to obtain a DSIC, AE, IR, and WBB mechanism.

6.1.2 Efficient Algorithms in restricted domains

Even if the problem of finding the optimal solution for a single–path MA seems to be hard, we have identified two restricted domains in which $f_E$ is easy.

Node–independent qualities Assume that, for every ad $a_i$, $i \in \mathcal{N}$, the following holds: $q_{i,m} = q_{i,m'} = q_i$ for all $m, m' \in \mathcal{K}$. In words, the visit probability does not depend on the specific node where the ad is shown, but only on the ad itself and the number of preceding ads shown. In this case, the mobile geo–location advertising reduces to the separable CTR model that is known to be easy [35]. Notice that, in this special case, the optimal advertising plan prescribes that all the slots (nodes) are filled with an ad.

Single–node maximal ads We say that an ad is maximal for a given node $s_m$, denoting it by $a_{m}^{\max}$, whenever the ad is the best one (in terms of expected value) for node $s_m$. Formally: $a_{m}^{\max} = \arg \max_{i \in \mathcal{N}} \{q_{i,m} \hat{v}_i\}$. Assume that each ad is maximal in at most a single node of the path. Formally: $a_{m}^{\max} \neq a_{m'}^{\max}$ for all $m, m' \in \mathcal{K}$ with $m \neq m'$. This is reasonable when there are many ads and the quality strongly depends on the distance between the shop and
the current position of the user, e.g., the user decides to visit the shop only if it is right next to him. In this case, if the algorithm allocates an ad to a given node, then it will allocate the maximal ad (this is not the case if an ad is maximal in multiple nodes).

The algorithm we proposed (Algorithm 4) is based on dynamic programming and works as follows. First, suppose that nodes of set $K$ are numbered in increasing order from the root $s_1$ to the leaf $s_K$. Each subproblem is characterized by a pair $(l, m)$ with $l \in \{0, \ldots, K - 1\}$ and $m \in \{1, \ldots, K\}$ and aims at finding the optimal allocation of the subpath of nodes from $s_m$ to $s_K$ when the number of ads allocated in the subpath of nodes from $s_1$ to $s_{m-1}$ is $l$. The rationale of the algorithm is to start from the leaf of the path and to move backward given that, in the case each node has a different maximal, the optimal allocation of a subproblem $(l, m)$ does not depend on the optimal allocation of a subproblem $(l', m')$ strictly including $(l, m)$, i.e., $l' \leq l$ and $m' < m$.

The algorithm uses two $K \times K$ matrices: $\Phi$ and $\Pi$. Each element $\Phi[l, m]$ is the optimal allocation of subproblem $(l, m)$ and it is represented as a set of pairs $(i, k)$ each indicating that ad $a_i$ is allocated in $s_k$; while each element $\Pi[l, m]$ is the expected value of the optimal allocation of subproblem $(l, m)$.

In Steps 1–3 the algorithm fills all the elements of the last column of $\Pi$, i.e., $\Pi[l, K] \forall l \in \{0, \ldots, K - 1\}$, with the value $\Lambda_l \cdot q_{a_{K,m}^{\text{max}}} \cdot \hat{v}_{a_{K,m}^{\text{max}}}$, i.e., the contribute that ad $a_{K,m}^{\text{max}}$ provides to the social welfare when $a_{K,m}^{\text{max}}$ is allocated in node $s_K$ and it is the $l + 1$–th allocated ad. Notice that any optimal allocation will have an ad allocated in the last node. Then, in Steps 5–15 the algorithm selects each node $s_m$ from $s_{K-1}$ to $s_1$, and finds the optimal advertising plan for the subpath from $s_m$ to $s_K$. Consider a generic element $\Pi[l, m]$, the algorithm decides whether it is better to allocate $a_m^{\text{max}}$ in $s_m$ as the $l + 1$–th ad (thus $\Pi[l, m] = \Lambda_l \cdot q_{a_m^{\text{max}}, m} \cdot \hat{v}_{a_m^{\text{max}}} + \Pi[l + 1, m + 1]$) or to leave node $s_m$ empty (thus $\Pi[l, m] = \Pi[l, m + 1]$). At the end of the execution, $\Phi[1, 1]$ returns the optimal allocation of ads into the nodes of the path. The algorithm just requires to fill the two matrices $\Phi$ and $\Pi$, thus its computational complexity is $O(K^2)$.

Notice that Algorithm 4 can also be extended to find the optimal allocation even when some ads are maximal in more than one node. The idea of the algorithm is to enumerate all the possible ways to remove all the conflicts and then, for each combination, to compute the best allocations using Algorithm 4. Specifically, first, choose an ad $a_i$ that is maximal in more than one node. In order to remove the conflicts, for each node in which $a_i$ is maximal, the algorithm creates a new problem where $a_i$ is maximal
Algorithm 4

1: for all \(l \in \{0, \ldots, K-1\}\) do
2: \(\Pi[l, K] = \Lambda_l \cdot q_{a_{K}^{max},K} \cdot \hat{v}_{a_{K}^{max}}\)
3: \(\Phi[l, K] = \{(a_{K}^{max}, K)\}\)
4: end for
5: \(m = K - 1\)
6: while \(m \geq 1\) do
7: for all \(l \in \{0, \ldots, m-1\}\) do
8: if \(\Pi[l, m+1] \geq \Lambda_l \cdot q_{a_{m}^{max},m} \cdot \hat{v}_{a_{m}^{max}} + \Pi[l+1, m+1]\) then
9: \(\Pi[l, m] = \Pi[l, m+1]\) and \(\Phi[l, m] = \Phi[l, m+1]\)
10: else
11: \(\Pi[l, m] = \Lambda_l \cdot q_{a_{m}^{max},m} \cdot \hat{v}_{a_{m}^{max}} + \Pi[l+1, m+1]\)
12: \(\Phi[l, m] = \Phi[l+1, m+1] \cup \{(a_{m}^{max}, m)\}\)
13: end if
14: end for
15: \(m = m - 1\)
16: end while
17: return \(\Phi[1, 1]\)

only for that node and is removed from the set of ads that can be displayed in the other conflicting nodes. This procedure removes some conflicts, but may generate new ones due ads becoming maximal in the nodes where \(a_{i}\) has been removed. This algorithm iteratively proceeds until each node has a different maximal ad and, thus, we can apply Algorithm 4. Finally, the algorithm returns the advertising plan that maximises the \(SW\) among the plans returned by all the executions of Algorithm 4. In the worst case, the complexity is \(O(K^{\min\{K,N\}}K^2)\) that is worse than the complexity of OptimalSinglePath.

Finally, we can conclude that in both the restricted domains it is possible to obtain a polynomial time DSIC mechanism adopting VCG payments with Clarke pivoting rule.

6.1.3 Approximate Mechanisms

Since the efficient mechanisms in the unrestricted domains discussed above do not scale, in this section we study approximation algorithms.

Existing results show that 3AP does not admit any polynomial–time approximation scheme (PTAS), but it does admit a constant–ratio (the best one is \(\frac{1}{2}\)) approximation algorithms [50]. These approximation algorithms are based on the similarity between 3AP and the weighted \(k\)–set packing (W\(k\)SP) problem and the existence of approximation algorithms with ratio \(O(\frac{1}{k})\) for this latter problem [7]. Specifically, any 3AP can be formulated as
Chapter 6. Mobile Advertising

a W3SP. However, our allocation problem cannot be formulated as \( W_k\)SP due to additional Constraints (6.4) that cannot be formulated as set packing constraints of the form \( \sum \cdot \leq 1 \). Hence, there is not any straightforward reduction allowing us to resort such approximation algorithms. However, it is possible to design an \textit{ad hoc} polynomial–time approximation algorithm with constant approximation ratio. A preliminary proposition, that recalls Approximation 2 in Sec. [5.4] is necessary for the description of the algorithm.

**Proposition 14.** Suppose we limit the total number of ads allocated, such that the continuation probability, \( \Lambda_{l} \), of the last allocated ad is at least \( \delta \), i.e., \( \forall m \in K : \Lambda_{\mu(m;\hat{\theta})} \geq \delta \). Then, the optimal social welfare given the reduced allocation space is at least \( (1 - \delta) \) the optimal social welfare when considering the set of all possible allocations \( \Theta \).

**Proof.** We denote by \( \hat{\theta}^* \) the optimal allocation of the MA problem and by \( K' \subseteq K \) the set of nodes in which non–empty ads are allocated. Divide \( K' \) into two subsets as follow: \( K'^+ \) is the set of nodes \( s_m \) with \( \Lambda_{\mu(m;\hat{\theta}^*)} \geq \delta \) and \( K'^- \) is the set of nodes \( s_m \) with \( \Lambda_{\mu(m;\hat{\theta}^*)} < \delta \). We can write:

\[
SW(\hat{\theta}^*, \hat{\nu}) = \sum_{m \in K'} \Lambda_{\mu(m;\hat{\theta}^*)} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m
\]

\[
= \sum_{m \in K'^+} \Lambda_{\mu(m;\hat{\theta}^*)} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m + \sum_{m \in K'^-} \Lambda_{\mu(m;\hat{\theta}^*)} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m
\]

where \( \sum_{m \in K'^+} \Lambda_{\mu(m;\hat{\theta}^*)} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m \) is the social welfare of the truncated solution that allocates ads only to nodes \( m \in K'^+ \).

Now consider the same allocation in which all the ads that are allocated to nodes \( m \in K'^+ \) are removed: \( \hat{\theta}' \). The resulting social welfare is \( \sum_{m \in K'^-} \Lambda_{\mu(m;\hat{\theta}')} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m \), where \( \Lambda_{\mu(m;\hat{\theta}')} \) is the aggregated continuation probability of node \( s_m \) when only the ads \( a_{\alpha(m';\hat{\theta}^*)} \) with \( m' \in K'^- \) are allocated.

Given the definition of \( K'^+ \) and \( K'^- \), we know that \( \frac{1}{\delta} \Lambda_{\mu(m;\hat{\theta}^*)} \leq \Lambda_{\mu(m;\hat{\theta}^*)} \), \( \forall m \in K'^- \), and that:

\[
\sum_{m \in K'^-} \Lambda_{\mu(m;\hat{\theta}')} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m \geq \frac{1}{\delta} \sum_{m \in K'^-} \Lambda_{\mu(m;\hat{\theta}^*)} \hat{\nu}_\alpha(m;\hat{\theta}^*) q_\alpha(m;\hat{\theta}^*),m
\]

Moreover, by definition of \( OPT \), we know that:
6.1. Single–Path Case

\[ SW(\theta^*, \hat{v}) \geq \sum_{m \in K'} \Lambda_{\mu(m; \theta^*)} \hat{\nu}_\alpha(m; \theta^*) q_\alpha(m; \theta^*),m \]  

(6.7)

Combining Eq. (6.6) with Eq. (6.7) we obtain that \( \delta SW(\theta^*, \hat{v}) \geq \sum_{m \in K'} \Lambda_{\mu(m; \theta^*)} \hat{\nu}_\alpha(m; \theta^*) q_\alpha(m; \theta^*),m \), and thus:

\[ SW(\theta^*, \hat{v}) = \sum_{m \in Nod'} \Lambda_{\mu(m; \theta^*)} \hat{\nu}_\alpha(m; \theta^*) q_\alpha(m; \theta^*),m \geq SW(\theta^*, \hat{v}) - \delta \cdot SW(\theta^*, \hat{v}) \]

finally:

\[ \sum_{m \in M^+} \Lambda_{\mu(m; \theta^*)} \cdot \hat{\nu}_\alpha(m; \theta^*) \cdot q_\alpha(m; \theta^*),m \geq (1 - \delta) \cdot SW(\theta^*, \hat{v}) \]

This completes the proof of the proposition.

We now present our approximate algorithm, \( f_A \), which is a slight modification of the OptimalSinglePath algorithm. The basic idea is that the exponential nature of the algorithm can be eliminated by fixing the maximum number of allocated non–empty ads to a given \( m \). The algorithm generates all the possible combinations \( B \) with \( |B| \leq m \) and then finds the optimal allocation for each combination \( B \) by calling a 2AP–solving oracle.

**Proposition 15.** Algorithm \( f_A \) has a polynomial computational complexity \( O(K^m N^3) \) and is an \( (1 - \prod_{l=1}^{m-1} \lambda_l) \)–approximation algorithm.

**Proof.** Consider the partial permutations of \( K \) elements in at most \( m \) positions. The number of partial permutations is \( K + K(K - 1) + \ldots + K(K - 1)(K - m + 1) = O(K^m) \). Since we are interested in combinations and not in permutations, we can bound the number of combinations by \( O(K^m) \). We can conclude that the computational complexity of the whole algorithm is \( O(K^m N^3) \), since the complexity of the Hungarian algorithm is \( O(N^3) \). This concludes the first part of the proof. Focus now on the approximation bound.

Given that \( f_A \) allocates at most \( m \) ads, all the allocated non–empty ads have an aggregated continuation probability of \( \Lambda_l \geq \prod_{l'=1}^{m-1} \lambda_{l'} \). Thus, applying Proposition 14 with \( \delta = \prod_{l=1}^{m-1} \lambda_{l'} \), we can conclude that \( f_A \) algorithm provides an approximation of \( (1 - \prod_{l=1}^{m-1} \lambda_{l'}) \) w.r.t. the optimal solution. \( \square \)
It is worth noting that \( f_A \) does not guarantee a constant approximation ratio given that \( \lambda_l \) can be arbitrarily close to 1 and, therefore, the bound can be arbitrarily close to 0. However, the approximation ratio does not depend on \( K \) and \( N \) and therefore the algorithm scales to large instances. We remark that when \( \lambda_l \) is close to 1, it would seem “natural” to approximate our allocation problem as a 2AP, by rounding \( \lambda_l \) to 1. This new algorithm is denoted by \( f_{A_2} \). However, we can state the following negative (from a mechanism design point of view) result.

**Proposition 16.** \( f_{A_2} \) is an \( \prod_{l=1}^{[N]-1} \lambda_l \)-approximation algorithm, but is not monotone.

**Proof.** Let \( SW(\theta, \hat{v}) = \sum_{m \in \mathcal{K}} \Lambda_{\mu(m; \theta)} q_{\alpha(m; \theta), m} \cdot \hat{v}_{\alpha(m; \theta)} \) and \( \hat{SW}(\theta, \hat{v}) = \sum_{m \in \mathcal{K}} q_{\alpha(m; \theta), m} \cdot \hat{v}_{\alpha(m; \theta)} \). The allocations that maximise the two social welfares are \( f_E(\hat{v}) = \theta^* = \arg \max_{\theta \in \Theta} SW(\theta, \hat{v}) \) and \( f_{A_2}(\hat{v}) = \tilde{\theta} = \arg \max_{\theta \in \Theta} \hat{SW}(\theta, \hat{v}) \). \( l_{\text{max}} \) is the number of allocated ads in \( \tilde{\theta} \).

\[
SW(\tilde{\theta}, \hat{v}) = \sum_{m \in \mathcal{K}} \Lambda_{\mu(m; \tilde{\theta})} q_{\alpha(m; \tilde{\theta}), m} \hat{v}_{\alpha(m; \tilde{\theta})} \geq \Lambda_{l_{\text{max}}-1} \sum_{m \in \mathcal{K}} q_{\alpha(m; \tilde{\theta}), m} \hat{v}_{\alpha(m; \tilde{\theta})} \\
\geq \Lambda_{l_{\text{max}}-1} \sum_{m \in \mathcal{K}} q_{\alpha(m; \theta^*), m} \hat{v}_{\alpha(m; \theta^*)} \\
\geq \Lambda_{l_{\text{max}}-1} \sum_{m \in \mathcal{K}} \Lambda_{\mu(m; \theta^*)} q_{\alpha(m; \theta^*), m} \hat{v}_{\alpha(m; \theta^*)} \\
= \Lambda_{l_{\text{max}}-1} SW(\theta^*, \hat{v})
\]

In the worst case, \( l_{\text{max}} = K \), thus the advertising plan produced by \( f_{A_2} \) has an approximation ratio of \( \prod_{l=1}^{K-1} \lambda_l \).

The non–monotonicity can be proved by counterexample. Consider the case in Tab. 6.1(a) with \( \lambda_l = 0.2 \) \( \forall l \in \{1, \ldots, K-1\} \), \( q_{1,1} = 0.5 \), and \( q_{1,2} = 1.0 \). \( f_{A_2} \) produces the allocation \( \theta_1 \) in which \( a_1 \) allocated in \( s_1 \) and \( a_2 \) in \( s_2 \); with this allocation \( VTR_1(\theta_1) = q_{1,1} = 0.5 \). Now consider the case in Tab. 6.1(b) where \( v_1 \) increases w.r.t. the previous case. \( f_{A_2} \) produces the allocation \( \theta_2 \) in which \( a_2 \) is allocated in \( s_1 \) and \( a_1 \) in \( s_2 \); with this allocation \( VTR_1(\theta_2) = \lambda_1 q_{1,2} = 0.2 \).

After having observed that \( f_{A_2} \) cannot be adopted in a DSIC mechanism, we prove that the adoption of \( f_A \) as allocation function allows the definition of DSIC mechanism.

**Proposition 17.** \( f_A \) is maximal in range.
Proof. \( f_A \) restricts the set of outcomes \( \Theta \) to a subset \( \Theta' \subseteq \Theta \) where only feasible allocations are those with at most \( \overline{m} \) allocated ads. The restriction does not depend on the reports of the advertisers but only on the information available to the mechanism. Given the reward declared by the advertisers, \( f_A \) selects the allocation \( \theta' = \arg \max_{\theta \in \Theta'} SW(\theta, \hat{v}) \), thus it is maximal in range.

As shown in [45], any allocation function that is maximal in range, if combined with VCG–like payments with Clarke pivoting as leads to a DSIC mechanism. Specifically, the mechanism satisfies also IR and WBB; the proof is easy by definition of VCG–like payments.

6.2 Multi–Path Case

In this section, we extend the results previously discussed to the general case with multiple paths.

6.2.1 Exact Allocation

We focus only on the allocation function, referred to as \( f_{EM} \) (since the VCG payments with Clarke pivot can again be used to obtain a DSIC mechanism). We can formulate the problem of finding the optimal allocation as an integer linear program by extending the single–path formulation as follows:

\[
\max \sum_{i \in N} \sum_{\psi \in \Psi_t} \sum_{l \in \mathcal{L}_\psi} \sum_{m \in \mathcal{K}_\psi} \omega_\psi \cdot \Lambda_l \cdot \hat{v}_i \cdot q_{i,m} \cdot x_{i,m,l,\psi}
\]

\[
\sum_{m \in \mathcal{K}_\psi} \sum_{l \in \mathcal{L}_\psi} x_{i,m,l,\psi} \leq 1 \quad \forall i \in \mathcal{N}, \psi \in \Psi_t \quad (6.8)
\]

\[
\sum_{i \in \mathcal{N}} \sum_{l \in \mathcal{L}_\psi} x_{i,m,l,\psi} \leq 1 \quad \forall \psi \in \Psi_t, m \in \mathcal{K}_\psi \quad (6.9)
\]

\[
\sum_{i \in \mathcal{N}} \sum_{m \in \mathcal{K}_\psi} x_{i,m,l,\psi} \leq 1 \quad \forall l \in \mathcal{L}_\psi, \psi \in \Psi_t \quad (6.10)
\]

\[
\sum_{i \in \mathcal{N}} x_{i,m,l,\psi} - \sum_{i \in \mathcal{N}} \sum_{m' \in \mathcal{K}_\psi: m' < m} x_{i,m',l-1,\psi} \leq 0 \quad \forall \psi \in \Psi_t, m \in \mathcal{K}_\psi, l \in \mathcal{L}_\psi \setminus \{0\} \quad (6.11)
\]
where $\mathcal{K}_\psi$ and $\mathcal{L}_\psi$ depend on the specific path $\psi$. Basically, the variables for the single–path case, $x_{i,m,l}$, are replicated for each path $\psi$, i.e., $x_{i,m,l,\psi}$. Each path $\psi$ must satisfy the same constraints we have in the single–path case and, in addition, Constraints (6.12) force nodes that are shared by multiple paths to be assigned to the same ad. The objective function maximizes the (expected) social welfare. We notice that differently from the single–path case, even when $\lambda = 1$, our problem can no longer be formulated as a 2AP (it is a variation of the 2AP with additional constraints whose continuous relaxation admits non–integer solutions). In this case, we use the classical branch–and–bound algorithm whose complexity is $O(N^K)$.

### 6.2.2 Efficient Algorithms in restricted domains

As in the case of single–path, we can identify a restricted domain where $f_{EM}$ is computationally easy. For the single–path case we have shown that, when the nodes of the path have different maximal ads, the problem becomes easy. In the multi–path environment we can state something stronger: when, for each path $\psi$, all the nodes belonging to $\mathcal{K}_\psi$ have different maximal ads, $f_{EM}$ is computationally easy. Thus, we allow an ad to be maximal in multiple nodes, as long as these nodes belong to different paths.

An optimal algorithm for this restricted domain is given by Algorithm 5, which extends Algorithm 4 to the multi–path case. To this end, we need to define two additional functions: $\text{suc} : \mathcal{K} \rightarrow \mathcal{P}(\mathcal{K})$, which returns, for any node $m \in \mathcal{K}$, $\text{suc}(m)$, the set of children nodes of $s_m$ in the multi–path tree; and $\text{cp} : \mathcal{K} \rightarrow \mathbb{N}$, which returns, for any node $m \in \mathcal{K}$, $\text{cp}(m)$, the number of nodes on the path from the root of the tree, $s_1$, to node $s_m$ (including the root $s_1$ and $s_m$).

Algorithm 5 is based on a recursive procedure and proceeds as follows. The base case is reached when the parameter of the algorithm is a leaf node. Then, the algorithm builds the optimal advertising plan from the subpaths generated starting from the leaf nodes and backtracking until the root node $s_1$ is reached. Each call to the algorithm $f_{MP}(s_m)$ requires the allocation of two vectors $\mathbf{\Phi}_m$ and $\mathbf{\Pi}_m$ with size $\text{cp}(m)$. $\Phi_m[l]$ and $\Pi_m[l]$ are the optimal allocation and its value respectively, in the subtree with root $s_m$ when the first displayed ad in the subtree will be the $l + 1$–th in the tree allocation.
Algorithm 5 $f_{MP}(m)$

1: if $suc(m) = \emptyset$ then
2:   for all $l \in \{0, \ldots, cp(m) - 1\}$ do
3:     $\Pi_m[l] = \sigma_m \Lambda_l q_{a_m^{\max},m} v_{a_m^{\max}}$ and $\Phi_m[l] = \{(a_m^{\max},m)\}$
4:   end for
5: else
6:   $\Pi_m[l] = 0$ and $\Phi_m[l] = \emptyset$
7:   for all $m' \in suc(m)$ do
8:     $\Pi_{m'}, \Phi_{m'} = f_{MP}(m')$
9:   end for
10: if $\sum_{m' \in suc(m)} \Pi_{m'}[l] \geq \sigma_m \Lambda_l q_{a_m^{\max},m} v_{a_m^{\max}} + \sum_{m' \in K} \Pi_{m'}[l + 1]$ then
11:   $\Pi_m[l] = \sum_{m' \in suc(m)} \Pi_{m'}[l]$ and $\Phi_m[l] = \bigcup_{m' \in suc(m)} \Phi_{m'}[l + 1]$
12: else
13:   $\Phi_m[l] = \bigcup_{m' \in suc(m)} \Phi_{m'}[l + 1] \cup \{(a_m^{\max},m)\}$
14: end if
15: end for
16: end if
17: end if
18: return $\Pi_m, \Phi_m$

Algorithm 5 starts its execution by being called with $f_{MP}(1)$, where $s_1$ is the root node of the multi–path tree. Steps 2-3 deal with the base case. If a leaf node $s_m$ has been reached, $\forall l \ \Pi_m[l]$ is filled with the value provided by the maximal ad $a_m^{\max}$ in node $s_m$ when it will be the $l + 1$–th displayed ad on its path, i.e., $\Pi_m[l] = \sigma_m \Lambda_l q_{a_m^{\max},m} v_{a_m^{\max}}$. Then, the algorithm stores in $\Phi_m[l], \forall l$, the fact that $a_m^{\max}$ has been allocated in $s_m$. Indeed, in any optimal allocation an ad is allocated in the last node of each path. Steps 6-15 deal with the recursion step. The algorithm, at Step 8 recursively call $f_{MP}(m')$ for all the children nodes $s_{m'}$ of $s_m$. Finally, the algorithm decides whether it is optimal to leave the node non–filled, Step 12, or to allocate $a_m^{\max}$ in $s_m$. Steps 14-15 These last operations are similar to the one of Algorithm 4, except that, for node $n$, the value provided by the ad allocated has to be weighted by the probability the node is reached during the paths ($\sigma_m$). At the end of the execution of Algorithm 5, $\Phi_1[1]$ stores the optimal advertising plan. The complexity of the algorithm is $O(|\Psi| |L| K)$.

### 6.2.3 Approximate Mechanism

We now consider an approximate mechanism for the multi-path setting, and show that it is possible to provide a maximal–in–range approximation algorithm $f_{AM}$ with approximation ratio that is constant w.r.t. $K$ and $N$, but
decreases with $|\Psi_t|$. Let $SW_\psi(\theta, \hat{v}) = \sum_{m \in K_\psi} \sigma_m \Lambda_{\mu(m; \theta) q(\alpha(m; \theta), m \hat{\alpha}(m; \theta))}$ denote the social welfare for a single path in the tree, and define $\theta^* = \arg \max_{\theta \in \Theta} SW(\theta, \hat{v})$ and $\theta^*_\psi = \arg \max_{\psi \in \Psi_t} SW_\psi(\theta^*, \hat{v})$. Given this, the following holds:

**Proposition 18.** The value $\max_{\psi \in \Psi_t} \{SW_\psi(\theta^*_\psi, \hat{v})\}$ is never worse than $1/|\Psi_t|$ of the optimal allocation for the entire tree.

**Proof.** Given the definition before we know that $SW_\psi(\theta^*_\psi, \hat{v}) \geq SW_\psi(\theta^*, \hat{v})$, $\forall \psi \in \Psi_t$, and that $\sum_{\psi \in \Psi_t} SW_\psi(\theta^*, \hat{v}) \geq SW(\theta^*, \hat{v}) = OPT$. Thus, since $\sum_{\psi \in \Psi_t} SW_\psi(\theta^*_\psi, \hat{v}) \geq OPT$, we can state that $\max_{\psi \in \Psi_t} \{SW_\psi(\theta^*_\psi, \hat{v})\} \cdot |\Psi_t| \geq OPT$ and therefore $1/|\Psi_t| \leq \max_{\psi \in \Psi_t} \{SW_\psi(\theta^*_\psi, \hat{v})\}$. This completes the proof. □

By using this proposition, we can provide a simple approximation algorithm that computes the best allocation $\theta^*_\psi$, $\forall \psi \in \Psi_t$ and selects the allocation of the path with the maximum $SW_\psi(\theta^*_\psi, \hat{v})$, obtaining a bound of $1/|\Psi_t|$. However, this algorithm requires exponential time, which is the same as finding the optimal allocation of the single-path problem. By approximating this latter problem as described in Sec. 6.1.3, we obtain a polynomial-time approximation algorithm with bound $1 - \prod_{l=1}^{\ell} \lambda_l |\Psi_t|$. It is easy to see that the algorithm is maximal in range as in the single-path case, and therefore it is possible to design a DSIC, WBB, IR, VCG–like mechanism with Clarke pivoting.

## 6.3 Experimental Evaluation

Given that this chapter does not provide a fine characterization of the complexity of the model under study, we complete the characterization of such model by providing an experimental evaluation where we compare the run time and the quality of the solutions obtained using the algorithms we described in the previous section.

### 6.3.1 Experimental setting

We represent the experimental environment by a $10 \times 10$ grid map in which each cell corresponds to a vertex of graph $G$. We associate each advertiser $a_i$ with a cell, $s_i$, in which we place the shop of $a_i$. The reward $v_i$ is uniformly drawn from $[0, 100]$. To generate paths, we randomly select a starting vertex $t_{st}$ and, from $t_{st}$, we build the paths moving randomly to the adjacent (horizontally and vertically) cells until the desired length
of the path is reached. The quality \( q_{i,m} \) is uniformly drawn from \([0, 1]\) if \( s_m = s_i \), and it is \( \max\{0, q_{i,i} - d_i \cdot \text{dist}(s_i, t)\} \) if \( s_m = t \neq s_i \), where \( d_i \) is a coefficient uniformly drawn from \([0, 1]\) and \( \text{dist}(s_i, t) \) is the Manhattan distance between \( s_i \) and \( t \) (normalized w.r.t. the maximum Manhattan distance among two cells in the grid map). The basic idea is that the quality linearly decreases as the distance between the current node and \( s_i \) increases, and \( d_i \) gives the decreasing speed. We assume a constant continuation probability \( \lambda_l = \lambda \forall l \in \{1, \ldots, K - 1\} \). We generate 50 instances for each of the following configurations: \( \lambda = 0.5 \) and \( K \in \{10, 20, 30, 40, 50\} \), and \( \lambda = 0.8 \) and \( K \in \{10, 20, 30\} \). In all instances \( N = 30 \). For our mathematical programming formulations we use AMPL as modeling language and CPLEX 11.0.1 to solve them. The experiments were conducted on an Unix computer with 2.33GHz CPU, 16Gb RAM, and kernel 2.6.32-45.

6.3.2 Single–path results

We ran \( f_E \) (specifically, the OptimalSinglePath implementation) and \( f_A \) with \( m \in \{1, 2, 3\} \). The results are depicted in Fig. 6.2. The average run time (left) and the average approximation ratio (AAR) obtained with different \( m \) (right) are plotted as \( K \) varies. The two top plots are with \( \lambda = 0.5 \), while the two bottom plots are with \( \lambda = 0.8 \). We observe that the run time of \( f_E \) strictly depends on \( \lambda \): the larger \( \lambda \), the longer the run time. This is because, in the optimal allocation, the number of allocated ads increases as \( \lambda \) increases (7 with \( \lambda = 0.5 \) and 16 with \( \lambda = 0.8 \)), requiring a larger number of possible allocations to be considered. With \( \lambda = 0.5 \), \( f_E \) can be used in practice to solve instances with a large number of nodes (up to 50) within \( 10^3 \) s, while, with \( \lambda = 0.8 \), \( f_E \) cannot be used for \( K > 30 \) (we found instances that were not solved even after 10 hours). Instead, the run time of \( f_A \) is constant in \( \lambda \), and, differently from the worst–case complexity, run time is sub linear in \( K \). On the other hand, the AAR, as theoretically expected, decreases as \( \lambda \) increases. However, \( f_A \) largely satisfies the theoretical bound, e.g., with \( \lambda = 0.5 \) and \( m = 2 \), the theoretical bound is 0.5, while we experimentally observed an AAR of 0.83.

6.3.3 Multi–path results

We ran \( f_{EM} \) and \( f_{AM} \) with \( m \in \{1, 2, 3\} \). The results are depicted in Fig. 6.3. By \( m^* \) we denote a variation of \( f_{AM} \) in which we adopt \( f_E \) to find the optimal solution \( \theta^*_{\psi} \) on the single path \( \psi \) (used because, as discussed above, the run time of \( f_E \) is tractable for 20 nodes or less). The figures show, for \( \lambda = 0.5 \), the average run time (left) and the AAR obtained with
Figure 6.2: Average run time (left) and approximation ratio (right) as $K$ varies. $\lambda = 0.5$ (top) and $\lambda = 0.8$ (bottom).
6.3. Experimental Evaluation

![Graphs showing run time and approximation ratio](image)

**Figure 6.3:** Average run time (left) and approximation ratio (right) as $|\Psi_t|$ varies with $\lambda = 0.5$ in the multi–path case.

different values of $m$ (right) as $|\Psi_t|$ varies, while the length of each path is uniformly drawn from $\{1, \ldots, 20\}$. With $|\Psi_t| = 15$ and 20, we interrupted the execution of $f_{EM}$ in 2 instances due to the set time limit (1200 s); with $\lambda = 0.8$, the number of interrupted executions is 11 when $|\Psi_t| = 20$. Thus, $f_{EM}$ can be used in practice with instances with no more than 20 paths and a small $\lambda$. We experimentally observed that the AAR is much better than the theoretical bounds. In particular, the theoretical bound decreases as $\frac{1}{|\Psi_t|}$; instead, experimentally, the ratios seem to converge to values $\geq 0.6$ as $|\Psi_t|$ increases. Also, $m^*$ provides the best performance in terms of trade–off between run time and AAR.
CHAPTER 7

Learning with Separable CTR

With this chapter we start the second part of the thesis, devoted to the study of truthful mechanisms in environments where some parameters are unknown to the auctioneer. Parameters that the auctioneer needs to estimate during the repeated executions of the mechanism.

Specifically, in this chapter we focus on the simplest model we introduced in Sec. 2.3 i.e., the separable CTR model. Hence, we consider the case where CTRs depend only on the quality of the ads and on the position of the slots in which the ads are allocated.

We have already seen that when all the parameters are known by the auctioneer, the efficient allocation $\theta^*$ prescribes that the ads are allocated to the slots in decreasing order w.r.t. their expected reported value $q_i \hat{v}_i$ and we call $f^*$ the allocation function returning it. More formally, for any $m \in \mathcal{K}_c$, let $\max_{i \in \mathcal{N}} (q_i \hat{v}_i; m)$ be the operator returning the $m$–th largest value in the set $\{q_1 \hat{v}_1, \ldots, q_N \hat{v}_N\}$, then $\theta^*$ is such that, for every $m \in \mathcal{K}_c$, the ad displayed at slot $s_m$ is

$$\alpha(m; \theta^*) \in \arg \max_{i \in \mathcal{N}} (q_i \hat{v}_i; m).$$  

(7.1)

This new formal definition simplifies the description of the efficient al-
location $\theta^*_i$ when $a_i$ is removed from $\mathcal{N}$. In fact, for any $i, j \in \mathcal{N}$, if $\pi(j; \theta^*) < \pi(i; \theta^*)$ (i.e., ad $a_j$ is displayed before $a_i$) then $\pi(j; \theta^*_{-i}) = \pi(j; \theta^*)$, while if $\pi(j; \theta^*) > \pi(i; \theta^*)$ then $\pi(j; \theta^*_{-i}) = \pi(j; \theta^*) - 1$ (i.e., ad $j$ is moved one slot upward), and w.l.o.g. we assume $\pi(i; \theta^*_{-i}) = N$. We conclude that the VCG payments can be expressed in the following simplified way:

$$
p^*_i(\hat{v}) = \begin{cases}
  \sum_{l=\pi(i;\theta^*)+1}^{K+1} (\Lambda_{l-1} - \Lambda_l) \max_{j \in \mathcal{N}} (q_j \hat{v}_j; l) & \text{if } \pi(i; \theta^*) \leq K, \\
  0 & \text{otherwise}
\end{cases}
$$

(7.2)

or as a per-slot payment in the following one:

$$
p^*_a(m; \theta^*)(\hat{v}) = \begin{cases}
  \sum_{l=m+1}^{K+1} (\Lambda_{l-1} - \Lambda_l) \max_{i \in \mathcal{N}} (q_i \hat{v}_i; l) & \text{if } m \leq K, \\
  0 & \text{otherwise}
\end{cases}
$$

(7.3)

In the sections of this chapter we study the problem of designing incentive compatible mechanisms under different conditions of lack of information over the parameters $q_i, i \in \mathcal{N}$, and $\Lambda_m, m \in \mathcal{K}$. Specifically, in Sec.7.1 we discuss the situations where the unique unknown parameter is the actual value of $q_i, i \in \mathcal{N}$. We will introduce an already known result over the regret on the revenue of the auctioneer [24] and then we complete the analysis providing a new result on the regret on the SW. In Sec.7.2 we assume that only the actual values of $\Lambda_m, m \in \mathcal{K}$, are unknown by the auctioneer. Specifically, in Sec.7.2.1 we design a mechanism DSIC in expectation w.r.t. the clicks realization and, in Sec.7.2.2 a mechanism DSIC in expectation w.r.t. its random component. We conclude, in Sec.7.2.3 the analysis of this environment with a consideration on DSIC mechanisms. Finally, in Sec.7.3 we assume that both $q_i, i \in \mathcal{N}$, and $\Lambda_m, m \in \mathcal{K}$, are unknown and we provide results on a mechanism in expectation w.r.t. its random component.

For simplicity, we also introduce two new definitions that will be used in the following two chapters: $\mathcal{F}$ is the allocation function set, while $\mathcal{F}_{-i}$ is the allocation function set when ad $a_i$ is excluded.

### 7.1 Unknown qualities $q_i, i \in \mathcal{N}$

In this section we assume that the qualities of the ads $q_i, i \in \mathcal{N}$, are unknown, while $\Lambda_m, m \in \mathcal{K}$, are known. We initially focus on DSIC mecha-
7.1. Unknown qualities $q_i, i \in \mathcal{N}$

nisms and subsequently we discuss mechanisms that are DSIC in expectation.

As in [11, 20], the authors of [24] formalized the problem as a multi–armed bandit problem and separated the exploration and exploitation phases, such that during the exploration phase they estimate the values of $q_i, i \in \mathcal{N}$, and during the exploitation phase they use the estimated qualities $\tilde{q}_i, i \in \mathcal{N}$. The pseudo code of the algorithm A–VCG1 (Adaptive VCG1) is given in Algorithm 6. The details of the algorithm follow.

Algorithm 6 Pseudo–code for the A–VCG1 mechanism.

**Input:** Length of exploration phase $\tau$, confidence $\delta$, prominences $\Lambda_m, m \in \mathcal{K}$

**Exploration phase**
for $t = 1, \ldots, \tau$ do
  Allocate ads according to (7.6)
  Ask for no payment
  Observe the clicks $\{\text{click}^i_{\pi(i; \theta_t)}(t)\}_{i=1}^N$
end for

Compute the estimated quality $\tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} \frac{\text{click}^i_{\pi(i; \theta_t)}(t)}{\Lambda_{\pi(i; \theta_t)}}$

Compute $\tilde{q}_i^+ = \tilde{q}_i + \eta$ where $\eta$ is given by (7.7)

**Exploitation phase**
for $t = \tau + 1, \ldots, T$ do
  Allocate ads according to $\tilde{f}$ defined in (7.9)
  For each ad $a_i$, ask for payment $\tilde{p}^e_i$ defined in (7.11)
end for

**Exploration phase** During the exploration phase, the algorithm receives as input the parameters $\Lambda_m, m \in \mathcal{K}$, and collects data to estimate the quality of each ad. This phase takes $\tau$ steps. At each step of the exploration phase, the algorithm collects $K$ samples (click or no–click events), one from each slot. Let $\theta_t$ (for $t \leq \tau$) be a sequence of (potentially arbitrary) allocations independent from the advertisers’ bids. Let set $B_i = \{t : \pi(i; \theta_t) \leq K, t \leq \tau\}$ contain all the steps when $a_i$ is allocated to a valid slot, so that $|B_i|$ corresponds to the total number of (click/no–click) samples available for ad $a_i$. Denote by $\text{click}^i_{\pi(i; \theta_t)}(t) \in \{0, 1\}$ the click event at time $t$ for ad $a_i$ when displayed in slot $s_{\pi(i; \theta_t)}$. Depending on the slot in which the click event occurs, the ad $a_i$ has different CTRs, thus the algorithm weigh each click sample by the probability of observation $\Lambda_m$ related to the slot in
which the ad was allocated. The estimated quality $\tilde{q}_i$ is computed as

$$\tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} \frac{\text{click}_i^{t} (t)}{\Lambda_{\pi (i; \theta_t)}}, \quad (7.4)$$

which is an unbiased estimate of $q_i$ (i.e., $\mathbb{E}_{\text{click}}[\tilde{q}_i] = q_i$, where $\mathbb{E}_{\text{click}}$ is the expectation w.r.t. the realization of the clicks). By applying the Hoeffding’s inequality [30], the authors of [24] obtained a bound over the error of the estimated quality $\tilde{q}_i$ for each ad $a_i$.

**Proposition 19.** [24] For any ad $a_i \in \mathcal{N}$

$$|q_i - \tilde{q}_i| \leq \sqrt{\left( \sum_{t \in B_i} \frac{1}{\Lambda_{\pi (i; \theta_t)}^2} \right) \frac{1}{2|B_i|^2} \log \frac{2N}{\delta}}, \quad (7.5)$$

with probability $1 - \delta$ (w.r.t. the click events).

During the exploration phase, at each step $t \in \{1, \ldots, \tau\}$, the authors of [24] adopt the following sequence of allocations

$$\theta_t = (a_{(t \mod N) + 1}, \ldots, a_{(t + N - 1 \mod N) + 1}), \quad (7.6)$$

thus obtaining $|B_i| = \left\lfloor \frac{K \tau}{N} \right\rfloor$ for all the ads $a_i$. Given that $\left\lfloor \frac{K \tau}{N} \right\rfloor \geq \frac{\tau K^3}{2N}$ [31], Eq. (7.5) becomes

$$|q_i - \tilde{q}_i| \leq \sqrt{\left( \sum_{m=1}^{K} \frac{1}{\Lambda_m^2} \right) \frac{2N}{K^2 \tau} \log \frac{2N}{\delta}} =: \eta. \quad (7.7)$$

During this phase, in order to guarantee DSIC, the advertisers cannot be charged with any payment, i.e., all the payments in steps $t \leq \tau$ are set to 0. In fact, as shown in [11], any bid–dependent payment could be easily manipulated by bidders with better estimates of the CTRs, thus obtaining a non–truthful mechanism, whereas bid–independent payments could lead to a non–IR mechanism to which bidders may prefer not to participate.

**Exploitation phase** Once the exploration phase is concluded, an upper–confidence bound over each quality is computed as

$$\tilde{q}_i^+ = \tilde{q}_i + \eta, \quad (7.8)$$

---

3 Notice that they realistically assume that all the ads have at least two samples to initialize their estimates $\tilde{q}_i^+$. This hypothesis allows to remove the floor notation in the bounds and, in the case of A–VCG1, it leads to an exploration time $\tau \geq 2N/K$. We will adopt the same convention in remaining of the thesis.
and the algorithm runs the exploitation phase for the remaining \( T - \tau \) steps. We define the estimated social welfare as:

\[
\tilde{SW}(\theta, \tilde{v}) = \sum_{i=1}^{N} \Lambda_{\pi(i;\theta)} \tilde{q}_i^+ \hat{v}_i,
\]

and we define \( \tilde{f} \) as the allocation function that displays ads in decreasing order of \( \tilde{q}_i^+ \hat{v}_i \). \( \tilde{f} \) returns the efficient allocation \( \tilde{\theta} \) on the basis of the estimated qualities as:

\[
\tilde{\theta} = \tilde{f}(\tilde{v}) \in \arg\max_{\theta \in \Theta} \tilde{SW}(\theta, \tilde{v}). \tag{7.9}
\]

The estimated efficient allocation \( \tilde{f} \) is adopted for all the steps of the exploitation phase. Notice that \( \tilde{f} \) is an affine maximizer, since

\[
\tilde{f}(\tilde{v}) \in \arg\max_{\theta \in \Theta} \sum_{i=1}^{N} \Lambda_{\pi(i;\theta)} \tilde{q}_i^+ \hat{v}_i = \arg\max_{\theta \in \Theta} \sum_{i=1}^{N} \frac{\tilde{q}_i^+}{q_i} \Lambda_{\pi(i;\theta)} q_i \hat{v}_i
\]

\[
= \arg\max_{\theta \in \Theta} \sum_{i=1}^{N} w_i \Lambda_{\pi(i;\theta)} q_i \hat{v}_i,
\]

where each weight \( w_i = \tilde{q}_i^+ / q_i \) is independent of the advertisers’ types \( v_i \). Hence, we can apply the WVCG (weighted–VCG) payments (here denoted by \( \tilde{p} \) because of the estimated parameters) satisfying the DSIC property. In particular, for any \( a_i \), we define the payment

\[
\tilde{p}_i(\tilde{v}) = \begin{cases} 
\frac{1}{w_i} \sum_{l=\pi(i;\tilde{\theta})+1}^{K+1} (\Lambda_{l-1} - \Lambda_{l}) \max_{j \in \mathcal{N}} (\tilde{q}_j^+ \hat{v}_j; l) & \text{if } \pi(i;\tilde{\theta}) \leq K \\
0 & \text{otherwise}
\end{cases}
\]

\[
= \begin{cases} 
\frac{q_i}{\tilde{q}_i} \sum_{l=\pi(i;\tilde{\theta})+1}^{K+1} (\Lambda_{l-1} - \Lambda_{l}) \max_{j \in \mathcal{N}} (\tilde{q}_j^+ \hat{v}_j; l) & \text{if } \pi(i;\tilde{\theta}) \leq K \\
0 & \text{otherwise}
\end{cases}.
\tag{7.10}
\]

It is evident that these payments cannot be computed by the auctioneer, since the actual \( q_i, i \in \mathcal{N} \), values are unknown. However, it is possible to resort to the pay–per–click payments

\[
\tilde{p}_i^c(\tilde{v}, \text{click}^i_{\pi(i;\tilde{\theta})}) = \frac{\tilde{p}_i(\tilde{v})}{\Lambda_{\pi(i;\tilde{\theta})} q_i}
\]
\[
\sum_{l=\pi(i, \tilde{\theta})+1}^{K+1} (\Lambda_{l-1} - \Lambda_l) \max_{j \in \mathcal{N}} (\tilde{q}^+_j \hat{v}_j, l) \text{click}^i_{\pi(i, \tilde{\theta})}.
\]

(7.11)

which in expectation, over the clicks, coincide with the payments \(\tilde{p}_i(\hat{v}) = \mathbb{E}[\tilde{p}_i^c(\hat{v}, \text{click}^i_{\pi(i, \tilde{\theta})})]\). Now, these payments can be computed simply relying on the estimates \(\tilde{q}^+_i\) and on the knowledge of the probabilities \(\Lambda_m\). The following properties hold for this mechanism.

**Proposition 20.** \([24]\) The A–VCG1 is DSIC, IR a posteriori, and WBB a posteriori.

The performance of A–VCG1 in terms of the auctioneer revenue regret the mechanism accumulates through the \(T\) steps, for which the mechanism is repeated, is bounded in the following way.

**Theorem 13.** \([24]\) Consider a sequential auction with \(N\) advertisers, \(K\) slots, and \(T\) steps with the separable CTR model with parameters \(\Lambda_m, m \in K\), and accuracy \(\eta\) as defined in Eq. (7.7). For any parameter \(\tau \in \{0, \ldots, T\}\) and \(\delta \in (0, 1)\), the A–VCG1 achieves a regret:

\[
R_T \leq v_{\max} \left( \sum_{m=1}^{K} \Lambda_m \right) \left( 2(T - \tau)\eta + \tau + \delta T \right).
\]

(7.12)

By setting the parameters to

\[
\delta = K^{-\frac{1}{3}} T^{-\frac{1}{3}} N^{\frac{1}{3}}
\]

\[
\tau = 2^{\frac{1}{3}} K^{-\frac{1}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} \Lambda_{\min}^{-\frac{2}{3}} \left[ \log \left( K^{\frac{1}{3}} T^{\frac{1}{3}} N^{\frac{2}{3}} \right) \right]^{\frac{1}{3}},
\]

where \(\Lambda_{\min} = \min_{m \in K} \Lambda_m > 0\) and \(v_{\max} = \max_{i \in N} v_i\), then the regret is

\[
R_T \leq 4 \cdot 2^{\frac{1}{3}} v_{\max} \Lambda_{\min}^{-\frac{2}{3}} K^{\frac{2}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} \left[ \log \left( K^{\frac{1}{3}} T^{\frac{1}{3}} N^{\frac{2}{3}} \right) \right]^{\frac{1}{3}}.
\]

(7.13)

Notice that unlike the standard bound for multi–armed bandit algorithms, the regret scales as \(\tilde{O}(T^{\frac{2}{3}})\) instead of \(\tilde{O}(T^{\frac{1}{2}})\). As pointed out in \([20]\) and \([11]\) this is the unavoidable price the bandit algorithm has to pay to be DSIC. In fact, the incentive compatibility hard constraint forces the learning mechanism to split the exploration and exploitation in two separate phases so that the quality estimates used during in exploitation rounds are completely independent from the bids.
In this dissertation, we do not solve two interesting problems when DSIC in expectation (w.r.t. the click realizations and/or realizations of the random component of the mechanism) is adopted: (i) whether or not it is possible to avoid the separation of the exploration and exploitation phases and (ii) whether it is possible to obtain a regret of $O(T^{\frac{1}{2}})$ as in the case of $K = 1$ [10]. Any attempt we tried to extend the result presented in [10] to the multi-slot case conducted us to a non-DSIC mechanism. We briefly provide some examples of adaptation to our framework of the two MAB presented in [10]. None of these attempts provided a monotone allocation function. We tried to extend the UCB1 in different ways, e.g., by introducing $N \cdot K$ estimators, one for each ad for each slot, or maintaining $N$ estimators weighting in different ways click obtained in different slots. The second MAB algorithm, called NewCB, is based on the definition of a set of active ads, i.e., the ones that can be displayed in the unique slot. We considered some extensions for the multi-slot setting (e.g., a single set for all the slots and multiple sets, one for each slot) without identifying monotone allocation algorithms.

Starting from the result of [24] on the revenue regret, we provide a new result identifying the bound for the regret over social welfare. Specifically, we observe that A–VCG1 is a no-regret algorithm even for the SW regret and $R_{SW}^T \leq \tilde{O}(T^{\frac{2}{3}})$.

**Theorem 14.** Let us consider a sequential auction with $N$ advertisers, $K$ slots, and $T$ steps with the separable CTR model with parameters $\Lambda_m, m \in \mathcal{K}$, and $\eta$ as defined in Eq. (7.7). For any parameter $\tau \in \{0, \ldots, T\}$ and $\delta \in (0, 1)$, the A–VCG1 achieves a regret:

$$R_{SW}^T \leq v_{\max} K \left(2 (T - \tau) \eta + \tau + \delta T\right).$$

(7.14)

By setting the parameters to

$$\delta = \left(\frac{\sqrt{2}}{\Lambda_{\min}}\right)^{\frac{2}{3}} K^{-\frac{1}{3}} N^{\frac{1}{3}} T^{-\frac{1}{3}},$$

$$\tau = \left(\frac{\sqrt{2}}{\Lambda_{\min}}\right)^{\frac{2}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} K^{-\frac{1}{3}} \left(\log 2^{\frac{2}{3}} \Lambda_{\min}^{\frac{2}{3}} N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{1}{3}}\right)^{\frac{1}{3}},$$

where $\Lambda_{\min} = \min_{m \in \mathcal{K}} \Lambda_m$, $\Lambda_{\min} > 0$, then the regret is

$$R_{SW}^T \leq 4 v_{\max} \left(\frac{\sqrt{2}}{\Lambda_{\min}}\right)^{\frac{2}{3}} K^{\frac{2}{3}} N^{\frac{1}{3}} T^{\frac{2}{3}} \left(\log 2^{\frac{2}{3}} \Lambda_{\min}^{\frac{2}{3}} N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{1}{3}}\right)^{\frac{1}{3}}.$$  

(7.15)
Notice that using $\tau$ and $\delta$ defined in Thm. 13, the bound for $R^T_{SW}$ is $\tilde{O}(T^{\frac{3}{2}})$, even if the parameters are not optimal for this second framework.

Before proving Thm. 14 we need to introduce two lemmas.

**Lemma 1.** Let $G$ be an arbitrary space of allocation functions, then for any $g \in G$, when $|qi - \tilde{q}_i^+| \leq \eta$ with probability $1 - \delta$, we have

$$0 \leq \left( \tilde{SW}(g(\hat{v}), \hat{v}) - SW(g(\hat{v}), \hat{v}) \right) \leq 2Kv_{\max}\eta,$$

with probability $1 - \delta$.

**Proof.** The first inequality follows from

$$SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v}) = \sum_{j: \pi(j;g(\hat{v})) \leq K} \Lambda_{\pi(j;g(\hat{v}))}(g(\hat{v}))\hat{v}_j \left( q_j - \tilde{q}_j^+ \right) \leq v_{\max} \sum_{j: \pi(j;g(\hat{v})) \leq K} \left( q_j - \tilde{q}_j^+ \right) \leq 0,$$

while the second inequality follows from

$$\tilde{SW}(g(\hat{v}), \hat{v}) - SW(g(\hat{v}), \hat{v}) = \sum_{j: \pi(j;g(\hat{v})) \leq K} \Lambda_{\pi(j;g(\hat{v}))}(g(\hat{v}))\hat{v}_j \left( \tilde{q}_j^+ - q_j \right) \leq v_{\max} \sum_{j: \pi(j;g(\hat{v})) \leq K} \left( \tilde{q}_j^+ - q_j \right) \leq 2Kv_{\max}\eta.$$

**Lemma 2.** Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, and a mechanism that separates the exploration ($\tau$ steps) and the exploitation phases ($T - \tau$ steps). Consider an arbitrary space of allocation functions $\mathcal{G}$, $\tilde{g} \in \arg\max_{g' \in \mathcal{G}} \tilde{SW}(g'(\hat{v}), \hat{v})$ and $|qi - \tilde{q}_i^+| \leq \eta$ with probability $1 - \delta$. For any $g \in \mathcal{G}$, an upper bound of the global regret over the $SW$ ($R^T_{SW}$) of the mechanism adopting $\tilde{g}$ instead of $g$ is:

$$R^T_{SW} \leq v_{\max}K [2(T - \tau)\eta + \tau + \delta T].$$
7.1. Unknown qualities \( q_i, i \in \mathcal{N} \)

**Proof.** We start providing a bound for the cumulative per–step regret during the exploitation phase.

\[
r = SW(g(\hat{v}), \hat{v}) - SW(\hat{g}(\hat{v}), \hat{v})
= SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v}) + \tilde{SW}(g(\hat{v}), \hat{v}) - \max_{g' \in G} \tilde{SW}(g'(\hat{v}), \hat{v})
\]

\[
\leq SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v}) + \tilde{SW}(\hat{g}(\hat{v}), \hat{v}) - SW(\hat{g}(\hat{v}), \hat{v})
\]

The two remaining terms \( r^1 \) and \( r^2 \) can be easily bounded by using Lemma 1

\[
r \leq r_1 + r_2 \leq 0 + 2Kv_{\text{max}}\eta = 2Kv_{\text{max}}\eta
\]

with probability \( 1 - \delta \).

Thus, we can conclude that:

\[
R_{SW}^T \leq v_{\text{max}}K \left[ 2(T - \tau)\eta + \tau + \delta T \right].
\]

We have now all the instruments to prove Thm. 14.

**Proof. Step 1: cumulative regret.** We apply Lemma 2 to separable CTR model with \( q_i, i \in \mathcal{N} \) unknowns, obtaining

\[
R_{SW}^T \leq v_{\text{max}}K \left[ 2(T - \tau)\eta + \tau + \delta T \right]
\]

\[
\leq v_{\text{max}}K \left[ 2(T - \tau)\sqrt{\frac{2}{\Lambda_{\text{min}}}} \sqrt{\frac{N}{K\tau}} \log \frac{2N}{\delta} + \tau + \delta T \right].
\]

**Step 2: parameter optimization.** In order to find values that optimize the bound over \( R_{SW}^T \), let \( e := \sqrt{\frac{2}{\Lambda_{\text{min}}}} \), then we first simplify the previous bound as

\[
R_{SW}^T \leq v_{\text{max}}K \left[ 2e \sqrt{\frac{N}{K\tau}} \log \frac{2N}{\delta} + \tau + \delta T \right].
\]

Taking the derivative of the previous bound w.r.t. \( \tau \) leads to

\[
v_{\text{max}}K \left( -\tau^{-\frac{3}{2}}eT \sqrt{\frac{N}{K}} \log \frac{2N}{\delta} + 1 \right) = 0,
\]

\[101\]
which leads to
\[ \tau = e^{\frac{2}{3} T^{\frac{2}{3}} N^{\frac{1}{3}} K^{-\frac{1}{3}}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}}. \]

Once replaced in the bound, we obtain
\[ R_{SW}^{SW} \leq \nu_{\text{max}} K \left[ 3e^{\frac{2}{3} T^{\frac{2}{3}} N^{\frac{1}{3}} K^{-\frac{1}{3}}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}} + \delta T \right]. \]

Finally, we choose \( \delta \) to optimize the asymptotic order by setting
\[ \delta = e^{\frac{2}{3} K^{-\frac{1}{3}} N^{\frac{1}{3}} T^{-\frac{1}{3}}} \]
given that \( \delta < 1 \) this imply that \( T > e^{2} K^{-1} N \).

The final bound is
\[ R_{SW}^{SW} \leq 4\nu_{\text{max}} e^{\frac{2}{3} K^{\frac{2}{3}} N^{\frac{1}{3}} T^{\frac{2}{3}}} \left( \log 2e^{-\frac{3}{2} N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{1}{3}}} \right)^{\frac{1}{3}}. \]

Notice that the choice of parameters \( \tau \) and \( \delta \), as in \([24]\), is obtained by a rough minimization of the upper–bound in Eq. 7.14. Each parameter can be computed by knowing the characteristics of the auction (number of steps \( T \), number of slots \( K \), number of ads \( N \), and \( \Lambda_{m} \)). Moreover, since the values are obtained optimizing an upper–bound of the regret and not directly the true cumulative regret, these values can provide a good guess for the parametrization, but there could be other values that better optimize the regret. Thus, in practice, the regret could be optimized by searching the space of the parameters around the values suggested in Thm. 14.

We can finally conclude that the both the regret on the SW and on the revenue of the auctioneer depends in the same way on the parameters \( N \), \( K \) and \( T \).

### 7.2 Unknown \( \Lambda_{m}, m \in \mathcal{K} \)

We now focus on the opposite situation, when the auctioneer knows \( q_{i}, i \in \mathcal{N} \), while \( \Lambda_{m}, m \in \mathcal{K} \), are unknown. By the definition of the separable CTR model, \( \Lambda_{m}, m \in \mathcal{K} \), are strictly non–increasing in \( m \). This property simplifies the allocation problem since the optimal allocation can be found without knowing the actual values of \( \Lambda_{m}, m \in \mathcal{K} \). Indeed, the allocation \( \theta^{*} \) such that \( \alpha(m; \theta^{*}) \in \arg \max_{i \in \mathcal{N}} (q_{i} \hat{v}_{i}; m) \), is optimal for all possible (feasible) \( \Lambda_{m}, m \in \mathcal{K} \). However, the lack of knowledge about \( \Lambda_{m}, m \in \mathcal{K} \) makes the
design of a truthful mechanism not straightforward because these parameters appear in the calculation of the payments.

Differently from what we presented in the previous section, here we initially focus on DSIC in expectation mechanisms, providing two mechanisms, the first DSIC in expectation w.r.t. the click realizations and the second DSIC in expectation w.r.t. the realizations of the random component of the mechanism, and subsequently we produce some considerations about DSIC mechanisms.

7.2.1 DSIC in expectation w.r.t. the click realizations mechanism

In this case, we do not need any estimation of the parameters $\lambda_m, m \in K$, and therefore we do not resort to the multi–armed bandit framework and the mechanism does not present separate phases. The pseudo code of the algorithm A–VCG2 (Adaptive VCG2) is given in Algorithm 7.

Algorithm 7 Pseudo–code for the A–VCG2 mechanism.

<table>
<thead>
<tr>
<th>Input: Qualities parameters $q_i, i \in N$</th>
</tr>
</thead>
<tbody>
<tr>
<td>for $t = 1, \ldots, T$ do</td>
</tr>
<tr>
<td>Allocate ads according to $f^*$ as prescribed by (7.1)</td>
</tr>
<tr>
<td>if Ad $a_i$ is displayed then</td>
</tr>
<tr>
<td>Ask for payment $p_i^c$ defined in (7.16)</td>
</tr>
<tr>
<td>end if</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

On the basis of the above considerations, we can adopt the allocatively efficient allocation function $f^*$ as prescribed by Eq. (7.1) even if the mechanism does not know the actual values of the parameters $\lambda_m, m \in K$. The problem is that the VCG payments defined in Eq. (7.2) cannot be computed, since $\lambda_m, m \in K$, are unknown. However, by resorting to execution–contingent payments [18, 26] (generalizing the pay–per–click approach$^4$), we can impose computable payments that, in expectation, are equal to Eq. (7.2). More precisely, the contingent payments are computed given the bids $\hat{v}$ and all click events over the slots and take the form:

$$p_i^c \left( \hat{v}, \{\text{click}_j \}_{j=1}^K \right) = \sum_{\pi(\cdot; \theta^*) \leq m \leq K} \text{click}_m^{\alpha(m; \theta^*)} \cdot \frac{q_{\alpha(m; \theta^*)}}{q_{\alpha(m; \theta^*)}} \cdot \hat{v}_{\alpha(m; \theta^*)}^{\alpha(m; \theta^*)} +$$

$^4$In pay–per–click payments, an advertiser pays only once its ad is clicked; in execution–contingent payments an advertiser pays when an event occurs and in our execution–contingent payments the events are clicks on the ads of other advertisers.
Chapter 7. Learning with Separable CTR

\[ - \sum_{\pi(i;\theta^*)<m\leq K} \text{click}_m^{\alpha(m;\theta^*)} \cdot \hat{v}_\alpha(m;\theta^*) \]  
(7.16)

Notice that the payment \( p_i^c \) depends not only on the click of ad \( a_i \), but also on the clicks of all the ads displayed in the slots below. In expectation, the two terms of \( p_i^c \) are:

\[
\mathbb{E}_{\text{click}} \left[ \sum_{\pi(i;\theta^*)\leq m\leq K} \text{click}_m^{\alpha(m;\theta^*)} \cdot \frac{q_{\alpha(m;\theta^*)} \cdot \hat{v}_\alpha(m;\theta^*)}{q_{\alpha(m;\theta^*)}} \right] = \sum_{\pi(j;\theta^*)\geq\pi(i;\theta^*)} \Lambda_{\pi(j;\theta^*)} q_j \hat{v}_j
\]

\[
\mathbb{E}_{\text{click}} \left[ \sum_{\pi(i;\theta^*)<m\leq K} \text{click}_m^{\alpha(m;\theta^*)} \cdot \hat{v}_\alpha(m;\theta^*) \right] = \sum_{\pi(j;\theta^*)>\pi(i;\theta^*)} \Lambda_{\pi(j;\theta^*)} q_j \hat{v}_j
\]

and therefore, in expectation, the payment equals to Eq. (7.2). Thus, we can state the following.

**Proposition 21.** The A–VCG2 is DSIC, IR, WBB in expectation (w.r.t. click realizations) and AE.

**Proof.** It trivially follows from the fact that the allocation function is AE and the payments in expectation equal the VCG payments.

We discuss further properties of the mechanism in what follows.

**Proposition 22.** The A–VCG2 is not DSIC a posteriori (w.r.t. click realizations).

**Proof.** The proof is by counterexample. Consider an environment with 3 ads \( a_i, i \in \mathcal{N} = \{1, 2, 3\} \) and 2 slots \( s_m m \in \mathcal{K} = \{1, 2\} \) s.t. \( q_1 = 0.5, v_1 = 4, q_2 = 1, v_2 = 1, q_3 = 1, v_3 = 0.5 \), which correspond to expected values of 2, 1, and 0.5.

The optimal allocation \( \theta^* \) consists in allocating \( a_1 \) in \( s_1 \) and \( a_2 \) in \( s_2 \). Consider a time \( t \) when both ad \( a_1 \) and \( a_2 \) are clicked, from Eq. (7.16) we have that the payment of \( a_2 \) is:

\[
p_2^c(v, \{\text{click}_{\pi(j;\theta^*)}^j\}_{j=1}^{K}) = \frac{1}{q_2} q_3 v_3 = 0.5
\]

If ad \( a_2 \) reports a value \( \hat{v}_2 = 3 \), the optimal allocation is now \( a_2 \) in \( s_1 \) e \( a_1 \) in \( s_2 \). In the case both \( a_1 \) and \( a_2 \) are clicked, the payment of \( a_2 \) is:

\[
p_2^c(\hat{v}, \{\text{click}_{\pi(j;\theta^*)}^j\}_{j=1}^{K}) = \frac{1}{q_2} q_1 v_1 + \frac{1}{q_1} q_3 v_3 - v_1 = 2 + 1 - 4 = -1
\]
Given that, in both cases, the utility is \( u_2(\cdot,\cdot) = v_2 - p_c^2(\hat{v}, \{\text{click}_{\pi(\cdot;\theta^*)}^j\}_{j=1}^K) \), reporting a non–truthful value is optimal. Thus, we can conclude that the mechanism is not DSIC.

**Proposition 23.** The A–VCG2 is IR a posteriori (w.r.t. click realizations).

**Proof.** Rename the ads \( a_i, i \in \mathcal{N} \), such that \( q_1 v_1 \geq q_2 v_2 \geq \ldots \geq q_N v_N \). We can write payments in Eq. (7.16) as:

\[
p_c^i (v, \{\text{click}_{\pi(j;\theta^*)}^j\}_{j=1}^K) = \sum_{j=i+1}^{K} \frac{\text{click}_j^i}{q_j} q_{j+1} v_{j+1} - \sum_{j=i+1}^{K} \text{click}_j^i v_j
\]

Thus, the utility for advertiser \( a_i \) is:

\[
u_i(v, \theta^*) = \text{click}_j^i v_i + \sum_{j=i+1}^{K} \frac{\text{click}_j^i}{q_j} q_{j+1} v_{j+1} - \sum_{j=i}^{K} \frac{\text{click}_j^i}{q_j} q_j v_j
\]

\[
= \sum_{j=i}^{K} \text{click}_j^i v_j - \sum_{j=i}^{K} \frac{\text{click}_j^i}{q_j} q_{j+1} v_{j+1}
\]

\[
= \sum_{j=i}^{K} \left( \frac{\text{click}_j^i q_j}{q_j} v_j - \frac{\text{click}_j^i}{q_j} q_{j+1} v_{j+1} \right)
\]

\[
= \sum_{j=i}^{K} \frac{\text{click}_j^i}{q_j} (q_j v_j - q_{j+1} v_{j+1}).
\]

Since \( \frac{\text{click}_j^i}{q_j} \geq 0 \) by definition and \( q_j v_j - q_{j+1} v_{j+1} \geq 0 \) because of the chosen ordering of the ads, then the utility is always positive and we can conclude that the mechanism is IR a posteriori.

**Proposition 24.** The A–VCG2 is not WBB a posteriori (w.r.t. click realizations).

**Proof.** The proof is by counterexample. Consider an environment with 3 ads \( a_i, i \in \mathcal{N} = \{1, 2, 3\} \), and 2 slots \( s_m, m \in \mathcal{K} = \{1, 2\} \), s.t. \( q_1 = 1, v_1 = 2, q_2 = 0.5, v_2 = 1, q_3 = 1, v_3 = 0.1 \).

The optimal allocation \( \theta^* \) consists in allocating \( a_1 \) in \( s_1 \) and \( a_2 \) in \( s_2 \). Consider step \( t \) when both ad \( a_1 \) and \( a_2 \) are clicked, their payments are:

\[
p_c^1 (v, \{\text{click}_{\pi(j;\theta^*)}^j\}_{j=1}^K) = \frac{1}{q_1} q_2 v_2 + \frac{1}{q_2} q_3 v_3 - v_2
\]

105
$\sum_{i=1}^{3} p_i^c \left( v, \{ \text{click}^{j}_{\pi(j; \theta^*)} \}_{j=1}^{K} \right) = 0.4 - 0.5 < 0$, and we can conclude that the mechanism is not WBB \textit{a posteriori}.

Now we state the following theorem, whose proof is straightforward.

\textbf{Theorem 15.} \textit{Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, with separable CTR model with parameters $\Lambda_m$, $m \in K$. The $A$–$VCG2$ achieves an auctioneer’s revenue expected regret $R_T = 0$.}

An important property of this mechanism is that the expected payments are exactly the VCG payments for the optimal allocation when all the parameters are known. Moreover, the absence of an exploration phase allows us to obtain a per–step expected regret of zero and, thus, the cumulative regret over the $T$ steps of auction $R_T = 0$. Similar considerations can be applied to the study of the regret over the social welfare, obtaining the following.

\textbf{Corollary 2.} \textit{The $A$–$VCG2$ has an expected regret over the social welfare $R_{SW}^T = 0$.}

\subsection*{7.2.2 DSIC in expectation w.r.t. random component realizations mechanism}

As for the previous mechanism, here we have only the exploitation phase, indeed the algorithm does not estimate parameters. Differently from $A$–$VCG2$, the mechanism we present in this section has a random component as proposed in [10]. The mechanism, called $A$–$VCG2'$, is reported in Algorithm 8. It is obtained applying the approach described in [10] to allocation function $f^*$.

\begin{algorithm}
\caption{Pseudo–code for the $A$–$VCG2'$ mechanism.}
\begin{algorithmic}
\Input Qualities parameters $q_i$, $i \in N$
\For {$t = 1, \ldots, T$} 
\State Allocate ads according to $f'^*$ as prescribed by Algorithm 9
\State For each ad $a_i$, ask for payment $p_i^{B,*c}$ defined in (7.18)
\EndFor
\end{algorithmic}
\end{algorithm}
Since $f^*$ is monotone and the problem is with single-parameter and linear utilities, as we have seen in Sec. 2.1.3 payments assuring DSIC can be written as Myerson payments:

$$p^*_i(\hat{v}) = \Lambda_{\pi(i;f^*(\hat{v}))} q_i \hat{v}_i - \int_0^{\hat{v}_i} \Lambda_{\pi(i;f^*(\hat{v}, u))} q_i du,$$

which coincide with the VCG payments. This is justified by the fact that when a mechanism is AE, IR and WBB the only payments that lead to a DSIC mechanism are the VCG payments with Clacke’s pivot [29], thus Eq. (7.17) must coincide. However, also in this case the problem is that these payments are not directly computable, because parameters $\Lambda_m, m \in K$, in the integral are unknown and, as in the case discussed in Section 7.2.1 we cannot replace them by empirical estimates. We can obtain these payments in expectation by using execution-contingent payments associated with non-optimal allocations where the report $\hat{v}_i$ is modified between 0 and the actual value. This can be obtained by resorting to the approach proposed in [10], which takes in input a generic allocation function $f$ and introduces a randomized component to it, thus producing a new (random) allocation function that we denote by $f'$. This technique, at the cost of reducing the efficiency of the mechanism, allows the computation of the allocation and the payments at the same time even when payments described in [5] cannot be computed directly.

We apply this approach to $f^*$ obtaining a new allocation function $f^{*'}$. With $f^{*'}$, the advertisers’ reported values $\hat{v}_i, i \in N$ are modified, each with a (small) probability $\mu$. The (potentially) modified values are then used to compute the allocation (using $f^*$) and the payments. More precisely, with a probability of $(1 - \mu)^N$, $f^*$ returns the same allocation as $f^*$, while it defines a different allocation with probability of $1 - (1 - \mu)^N$. The reported values $\hat{v}_i, i \in N$, are modified through the canonical self-resampling procedure (cSRP) described in [10] that generates two samples: $x_i(\hat{v}_i, \omega_i)$ and $y_i(\hat{v}_i, \omega_i)$, where $\omega_i$ is the random seed. We sketch the result of cSRP:

$$(x_i, y_i) = cSRP(\hat{v}_i) = \begin{cases} (\hat{v}_i, \hat{v}_i) & \text{w.p. } 1 - \mu \\ (\hat{v'}_i, \hat{v'}_i) & \text{otherwise} \end{cases},$$

where $\hat{v'}_i \sim U([0, \hat{v}_i])$ and $\hat{v''}_i = \text{rec}(\hat{v'}_i)$. rec($\hat{v'}_i$) is a recursive algorithm returning $\hat{v'}_i$ with probability $1 - \mu$, otherwise it samples a value in $U([0, \hat{v}_i])$ and recursively call rec on the new value. More details can be found in [10].

Algorithm 9 shows how $f^{*'}$ works when the original allocation function is $f^*$. The reported values $\hat{v}_i, i \in N$, are perturbed through the canonical
Chapter 7. Learning with Separable CTR

Algorithm 9 \( f^*(\hat{v}) \)

1: for all \( i \in N \) do
2: \((x_i, y_i) = cSRP(\hat{v}_i)\)
3: \( x = (x_1, \ldots, x_N) \)
4: end for
5: \( \theta = f^*(x) \)

self–resampling procedure (Step 2) and then the allocation is chosen by applying the original allocation function \( f^* \) to the new values \( x \) (Step 5). Finally, the payments are computed as

\[
p_{i}^{B,*}(x, click^i_{\pi(i; f^*(x))}) = \begin{cases} 
p_{i}^{B,*}(x, y; \hat{v}) &= \text{if } click^i_{\pi(i; f^*(x))} = 1 \\
0 &= \text{otherwise} 
\end{cases}
\]

\[
= \begin{cases} 
\hat{v}_i - \frac{1}{\mu} \hat{v}_i &= \text{if } y_i < \hat{v}_i \\
0 &= \text{otherwise,} 
\end{cases}
\text{if } click^i_{\pi(i; f^*(x))} = 1 \\
\end{cases}
\]

(7.18)

where

\[
p_{i}^{B,*}(x, y; \hat{v}) = \Lambda_{\pi(i; f^*(x))}q_i \hat{v}_i - \begin{cases} 
\frac{1}{\mu} \Lambda_{\pi(i; f^*(x))}q_i \hat{v}_i &= \text{if } y_i < \hat{v}_i \\
0 &= \text{otherwise} 
\end{cases}
\]

(7.19)

with \( y = (y_1, \ldots, y_N) \). The expectation, w.r.t. the randomization of the mechanism, of the payments in Eq. (7.18) coincides with the payments defined in [5] instantiated for the randomized allocation function \( f^{*'} \). The result presented in [10] assures that the resulting mechanism is DSIC in expectation w.r.t. the click realizations.

We state the following results on the properties of the above mechanism.

**Theorem 16.** Let us consider an auction with \( N \) advertisers, \( K \) slots, and \( T \) steps, with the separable CTR model with parameters \( \Lambda_m, m \in \mathcal{K} \). The A–VCG2’ achieves an expected regret \( R_T \leq 2K^2 \mu v_{\text{max}} T \).

**Proof.** **Step 1: payments and additional notation.** We recall that according to [5] and [29] the expected VCG payments can be written, as in Eq. (7.17), in the form

\[
p_i^*(\hat{v}) = \Lambda_{\pi(i; f^*(\hat{v}))}q_i \hat{v}_i - \int_0^{\hat{v}_i} \Lambda_{\pi(i; f^*(\hat{\nu}_i, u))}q_i du,
\]

108
while the A–VCG2' mechanism prescribes contingent payments as in Eq. (7.18), which leads to expected payments

$$p_i^*(\hat{\mathbf{v}}) = \mathbb{E}_\mathbf{x}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_\mathbf{x}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}_i, u] q_i du.$$  

(7.20)

Given the randomness of the allocation function of A–VCG2', we need to introduce the following additional notation:

- $\mathbf{z} \in \{0, 1\}^N$ is a vector where each element $z_i$ denotes whether the $i$–th bid has been preserved or it has been modified by the self–resampling procedure, i.e., if $x_i = \hat{v}_i$ then $z_i = 1$, otherwise if $x_i < \hat{v}_i$ then $z_i = 0$. Notice that $\mathbf{z}$ does not provide information about the actual modified values $x$;

- $\mathbb{E}_{\mathbf{x}|\mathbf{z}}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}]$ is the expected value of prominence associated with the slots allocated to ad $a_i$ conditioned on the declared bids $\hat{\mathbf{v}}$ being perturbed as in $\mathbf{z}$.

Let $Z = \{z|\pi(i; f^*(\hat{\mathbf{v}})) \leq K + 1 \Rightarrow z_i = 1 \ \forall i \in N\}$ be all the realizations where the self–resampling procedure does not modify the bids of the first $K + 1$ ads, i.e., the $K$ ads displayed applying $f^*$ to the true bids $\hat{\mathbf{v}}$ and the first non–allocated ad.

**Step 2: cumulative regret.** We proceed by studying the per–ad regret $r_i(\hat{\mathbf{v}}) = p_i^*(\hat{\mathbf{v}}) - p_i^\nu(\hat{\mathbf{v}})$. Given the previous definitions, we rewrite the expected payments $p_i^\nu(\hat{\mathbf{v}})$ as

$$p_i^\nu(\hat{\mathbf{v}}) = \left(\mathbb{P}[\mathbf{z} \in Z] \Lambda_{\pi(i;f^*(\hat{\mathbf{v}}))} + \mathbb{P}[\mathbf{z} \not\in Z] \mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}]\right) q_i \hat{v}_i +$$

$$\quad - \int_0^{\hat{v}_i} \left(\mathbb{P}[\mathbf{z} \in Z] \Lambda_{\pi(i;f^*(\hat{\mathbf{v}}), u)} + \mathbb{P}[\mathbf{z} \not\in Z] \mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}_i, u]\right) q_i du$$

$$= \mathbb{P}[\mathbf{z} \in Z] \left(\Lambda_{\pi(i;f^*(\hat{\mathbf{v}}))} q_i \hat{v}_i - \int_0^{\hat{v}_i} \Lambda_{\pi(i;f^*(\hat{\mathbf{v}}), u)} q_i du\right) +$$

$$\quad + \mathbb{P}[\mathbf{z} \not\in Z] \left(\mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}_i, u] q_i du\right)$$

$$= \mathbb{P}[\mathbf{z} \in Z] p_i^*(\hat{\mathbf{v}}) +$$

$$\quad + \mathbb{P}[\mathbf{z} \not\in Z] \left(\mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_{\mathbf{x}|\mathbf{z} \not\in Z}[\Lambda_{\pi(i;f^*(\mathbf{x}))}|\hat{\mathbf{v}}_i, u] q_i du\right),$$

where in the last expression we used the expression of the VCG payments in Eq. (7.17) according to [5] and [29]. The per–ad regret is

$$r_i(\hat{\mathbf{v}}) = p_i^*(\hat{\mathbf{v}}) - p_i^\nu(\hat{\mathbf{v}})$$
Chapter 7. Learning with Separable CTR

\begin{align*}
&= p_i^*(\hat{v}) - \mathbb{P}[z \in Z] p_i^*(\hat{v}) + \\
&- \mathbb{P}[z \not\in Z] \left( \mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}] q_i \hat{v}_i - \int_{0}^{\hat{v}_i} \mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}_i, u] q_i du \right) \\
&= \mathbb{P}[z \not\in Z] p_i^*(\hat{v}) + \\
&- \mathbb{P}[z \not\in Z] \left( \mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}] q_i \hat{v}_i - \int_{0}^{\hat{v}_i} \mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}_i, u] q_i du \right).
\end{align*}

Since we have that \( u \leq \hat{v}_i \) in the integral and since the allocation function defined in [10] is monotone, we have that

\[
\mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}_i, u] \leq \mathbb{E}_{x \mid z \not\in Z} [\Lambda_{\pi(i; f^*(x))} | \hat{v}],
\]

which implies that \( r_{i,1}^B \) is non-negative. Thus the regret \( r_i^B \) can be bounded as

\[
r_i^B(\hat{v}) = \mathbb{P}[z \not\in Z] p_i^*(\hat{v}) - \mathbb{P}[z \not\in Z] r_{i,1}^B \\
\leq \mathbb{P}[z \not\in Z] p_i^*(\hat{v}) \leq \mathbb{P} [\exists j : z_j = 0 \land \pi(j; f^*(\hat{v})) \leq K + 1] v_{\text{max}} \\
\leq \sum_{j \in N: \pi(j; f^*(\hat{v})) \leq K + 1} \mathbb{P}[z_j = 0] v_{\text{max}} = (K + 1) \mu v_{\text{max}} \leq 2K \mu v_{\text{max}}.
\]

(7.21)

We can now compute the bound on the global regret \( R_T \). Since this mechanism does not require any estimation phase, the regret is simply

\[ R_T \leq 2K^2 \mu v_{\text{max}} T. \]

**Step 3: parameters optimization.** In this case, the bound would suggest to choose a \( \mu \to 0 \), but it is necessary to consider that with \( \mu \to 0 \) the variance of the payment goes to infinity.

Adopting \( \mu = \frac{\epsilon}{K^2 T} \) we obtain that \( R_T \to 0 \) as \( \epsilon \to 0 \), but, given that \( \mu \) cannot be 0 due to Eq. (7.18), we conclude \( R_T = O(1) \). The problem associated with the choice of \( \mu \) is that, as we will show in Chapter 9, the smaller \( \mu \) the larger the variance of the payments. We provide a similar result for the regret over the social welfare.

**Theorem 17.** Let us consider an auction with \( N \) advertisers, \( K \) slots, and \( T \) steps, with separable CTR model with parameters \( \Lambda_m, m \in \mathcal{K} \). The A–VCG2′ achieves an expected regret \( R_T^{SW} \leq K^2 \mu v_{\text{max}} T \).
The bound over the global regret on the social welfare \( R^{SW}_T \) can be easily derived considering that each bid is modified by the self–resampling procedure with a probability of \( \mu \). Thus we can define \( Z' = \{z'|z' \in \{0, 1\}^N, \pi(i; f^*(\hat{v})) \leq K \Rightarrow z'_i = 1\} \), i.e., the set of the random realizations where the self–resampling procedure does not modify the bids of the ads displayed when the allocation function \( f^* \) is applied to the true bids \( \hat{v} \). Thus we have:

\[
R^{SW}_T \leq T \left( \mathbb{P}[z \in Z'] \cdot 0 + \mathbb{P}[z \notin Z'] K v_{\text{max}} \right) \leq K^2 \mu v_{\text{max}} T.
\]

\[\Box\]

### 7.2.3 Considerations about DSIC mechanisms

At the cost of worsening the regret, one may wonder whether there exists a no–regret DSIC mechanism. In what follows, resorting to the same arguments used in \cite{49}, we show that the answer to such question is negative.

**Theorem 18.** Let us consider an auction with \( N \) advertisers, \( K \) slots, and \( T \) steps, with the separable CTR model with parameters \( \Lambda_m, m \in \mathcal{K} \), whose value are unknown. Any online learning DSIC a posteriori (w.r.t. click realizations) mechanism achieves an expected regret \( R_T = \Theta(T) \).

**Proof.** (sketch) Basically, the A–VCG2 mechanism is only DSIC in expectation (and not DSIC) because it adopts execution–contingent payments in which the payment of advertiser \( a_i \) depends also on the clicks over ads different from \( a_i \). The above payment technique—i.e., payments reported in Eq. (7.16)—is necessary to obtain in expectation the values \( SW(\theta^*_{-i}, \hat{v}_{-i}) \) and \( SW_{-i}(\theta^*, \hat{v}) \), since parameters \( \Lambda_m, m \in \mathcal{K} \) are not known. In order to have DSIC a posteriori, we need payments \( p_i \) that are deterministic w.r.t. the clicks over other ads different from \( a_i \) (i.e., pay–per–click payments are needed).

We notice that even if \( \Lambda_m \) have been estimated (e.g., in an exploration phase), we cannot have payments leading to DSIC. Indeed, with estimates \( \tilde{\Lambda}_m \), the allocation function maximizing \( \tilde{SW} \) (computed with \( \tilde{\Lambda}_m \)) is not an affine maximizer and therefore the adoption of WVCG mechanism would not guarantee DSIC. As a result, only mechanisms with payments defined as in \cite{5} can be used. However, these payments, if computed exactly (and not estimated in expectation), require the knowledge about the actual \( \Lambda_m \) related to each slot \( s_m \) in which an ad can be allocated for each report \( \hat{v} \leq v \).
To prove the theorem, we provide a characterization of DSIC mechanisms. Exactly, we need a monotone allocation function and the Mayer-son payments defined in \cite{5}. These payments, as said above, require the knowledge about the actual $\Lambda_m$ related to the slot $s_m$ in which an ad can be allocated for each report $\hat{v} \leq v$. Thus we have two possibilities:

- **In the first case**, the ads are partitioned and each partition is associated with a single slot and the ad with the largest expected valuation is chosen at each slot independently. In other words, an ad can be allocated only in one given specific slot and its report determines only whether it is displayed or not (and not where). This case is equivalent to multiple separate–single slot auctions and therefore each auction is DSIC as shown in \cite{20}. However, as shown in \cite{49}, this mechanism would have a regret $\Theta(T)$.

- **In the second case**, the ads are partitioned and each partition is associated with multiple slots and for each partition an auction is carried out. In other words, an ad can be allocated in more than one slot on the basis of its report. In this case, to compute the payments, it would be necessary to know the exact CTRs of the ad for each possible slot, but this is possible only in expectation either by using the above execution–contingent as we do in Sec. 7.2.1 or by generating non–optimal allocation as we do in Sec. 7.2.2.

Thus, in order to have DSIC, we need to adopt the class of mechanisms described in the first case, obtaining $R_T = \Theta(T)$. \hfill $\square$

### 7.3 Unknown $\Lambda_m, m \in K$, and $q_i, i \in N$

In order to complete the study of the separable CTR model, it remains to study the scenario in which both $q_i, i \in N$, and $\Lambda_m, m \in K$, are unknown. From the results discussed in the previous section, we know that adopting DSIC as solution concept we would obtain $R_T = \Theta(T)$. Thus, we focus only on DSIC in expectation.

First of all, we remark that the mechanisms presented in Sec. 7.1 and 7.2 cannot be adopted here, but the study of a new mechanism is required. By combining A–VCG1 and A–VCG2', we obtain the algorithm A–VCG3 (Adaptive VCG3) illustrated in Algorithm 10. As in the case in which only $q_i, i \in N$, are unknown, we formalize the problem as a multi–armed bandit where the exploration and exploitation phases are separate and where, during the exploration phase, we estimate the values of $q_i, i \in N$. Details of the algorithm follow.
Algorithm 10 Pseudo–code for the A–VCG3 mechanism.

**Input:** Length of exploration phase $\tau$, confidence $\delta$

**Exploration phase**
for $t = 1, \ldots, \tau$ do
  Allocate ads according to (7.6)
  Ask for no payment
  Observe the clicks $\{click(t)\}_{i=1}^{N}$
end for

Compute the estimated quality $\tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} click(t)$
Compute $\tilde{q}_i^+ = \tilde{q}_i + \eta$ where $\eta$ is given by (7.22)

**Exploitation phase**
for $t = \tau + 1, \ldots, T$ do
  Allocate ads according to $\tilde{f}'$ as prescribed by Algorithm 9 adopting $\tilde{f}$ instead of $f^*$
  For each ad $a_i$, ask for payment $\tilde{p}_{B,c}^{B,c}$ defined in (7.23)
end for

Exploration phase During the first $\tau$ steps of the auction, the algorithm computes $q_i$, $i \in \mathcal{N}$. We adopt the same exploration policy of Sec. 7.1, but the estimations are computed just using samples from the first slot, since $\Lambda_m$ with $m > 1$ are unknown. (In the following, we report some considerations about the case in which also the samples from the slots below the first are considered.) Define $B_i = \{t : \pi(i; \theta_t) = 1, t \leq \tau\}$ the set of steps $t \leq \tau$ where $a_i$ is displayed in the first slot, the number of samples collected for $a_i$ is $|B_i| = \lfloor \frac{\tau}{N} \rfloor \geq \frac{\tau}{2N}$. The estimated value of $q_i$ is:

$$\tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} click(t).$$

such that $\tilde{q}_i$ is an unbiased estimate of $q_i$ (i.e., $\mathbb{E}_{\text{click}}[\tilde{q}_i] = q_i$). By applying the Hoeffding’s inequality we obtain an upper bound over the error of the estimated quality $\tilde{q}_i$ for each ad $a_i$.

**Proposition 25.** For any ad $a_i$, $i \in \mathcal{N}$

$$|q_i - \tilde{q}_i| \leq \sqrt{\frac{1}{2|B_i|} \log \frac{2N}{\delta}} \leq \sqrt{\frac{N}{\tau} \log \frac{2N}{\delta}} =: \eta,$$

with probability $1 - \delta$ (w.r.t. the click events).

After the exploration phase, an upper–confidence bound over each quality is computed as $\tilde{q}_i^+ = \tilde{q}_i + \eta$. 

113
Exploitation phase We first focus on the allocation function. During the exploitation phase we want to use an allocation $\tilde{\theta} = \tilde{f}(\hat{\nu})$ maximizing the estimated social welfare with estimated $\tilde{q}_i^+, i \in N$, and the parameters $\Lambda_m, m \in K$. Since the actual parameters $\Lambda_m, m \in K$, are monotonically non-increasing we can use an allocation $\tilde{\theta}$, where

$$\alpha(m; \tilde{\theta}) \in \arg\max_{i \in N}(\tilde{q}_i^+ \hat{v}_i; m) = \arg\max_{i \in N}(\tilde{q}_i^+ \Lambda_m \hat{v}_i; m).$$

Notice that the allocation function $\tilde{f}$ is an affine maximizer (due to weights depending on $\tilde{q}_i$ as in Sec. 7.1), but WVCG payments cannot be computed given that parameters $\Lambda_m, m \in K$ are unknown. Neither the adoption of execution-contingent payments, like in Eq. (7.16), is allowed, given that $q_i$ is unknown and only estimates $\tilde{q}_i$ are available.

Thus, we resort to implicit payments as in Sec. 7.2.2. More precisely, we use the same exploitation phase we used in Sec. 7.2.2 except that we adopt $\tilde{f}$ in place of $f^*$. In this case, we have that the per-click payments are:

$$\tilde{p}^B_{i,c}(x, \text{click}^i_{\pi; f(x)}) = \begin{cases} \tilde{p}^B_{i,x,y; \hat{\nu}} (\pi(i; f(x)))q_i \hat{v}_i & \text{if click}^i_{\pi; f(x)} = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} \hat{v}_i - \left\{ \begin{array}{ll} \frac{1}{\mu} \hat{v}_i & \text{if } y_i < \hat{v}_i \\ 0 & \text{otherwise,} \end{array} \right. \text{if } \text{click}^i_{\pi; f(x)} = 1 \\ 0 & \text{otherwise} \end{cases}$$

(7.23)

where

$$\tilde{p}^B_{i,x,y; \hat{\nu}} = \Lambda_{\pi(i; f(x))}q_i \hat{v}_i - \begin{cases} \frac{1}{\mu} \Lambda_{\pi(i; f(x))}q_i \hat{v}_i & \text{if } y_i < \hat{v}_i \\ 0 & \text{otherwise,} \end{cases}$$

(7.24)

We can state the following.

**Theorem 19.** The A–VCG3 is DSIC and WBB in expectation (w.r.t. the realizations of the random component of the mechanism) and IR a posteriori (w.r.t. the random component of the mechanism). These properties hold a posteriori w.r.t. the click realizations.

**Proof.** The proof of DSIC in expectation and WBB in expectation easily follows from the definition of the adopted mechanism as discussed in [10]. The proof of IR a posteriori is similar to the proof of Proposition [23]. The fact that the properties hold a posteriori w.r.t. the click realizations follows from [10].

\[\square\]
Now we want to analyze the performance of the mechanism in terms of regret cumulated through $T$ steps. Notice that in this case we have to focus on two different potential sources of regret: the adoption of a sub–optimal (randomized) allocation function and the estimation of the unknown parameters.

**Theorem 20.** Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, with the separable CTR model with parameters $\Lambda_m, m \in \mathcal{K}$. For any parameter $\tau \in \{0, \ldots, T\}$ and $\delta \in (0, 1)$, the A–VCG3 achieves a regret

$$R_T \leq v_{\max} \left( T - \delta \right) \left( 2\eta + 2\mu N \right) + \tau + \delta T$$

By setting the parameters to

- $\mu = N^{-2/3} T^{-1/3}$. $\mu$ is always $\leq 1$
- $\delta = N^{1/3} T^{-1/3}$. $\delta \leq 1$, thus $T \geq N$
- $\tau = T^{2/3} N^{1/3} \left( \log \frac{2N}{\delta} \right)^{1/3}$

then the regret is

$$R_T \leq 6v_{\max} KT^{2/3} N^{1/3} \left( \log \left( 2N^{2/3} T^{1/3} \right) \right)^{1/3} \quad (7.25)$$

**Proof.** **Step 1: payments and the regret.** Similar to the proof of Thm. [16], we use the form of the VCG payments as in Eq. (7.17):

$$p_i^*(\hat{\mathbf{v}}) = \Lambda_{\pi(i); f^*(\hat{\mathbf{v}})} q_i \hat{v}_i - \int_0^{\hat{v}_i} \Lambda_{\pi(i); f^*(\hat{\mathbf{v}}-u)} q_i du,$$

while A–VCG3 uses the contingent payments in Eq. (7.23), which in expectation become

$$\tilde{p}_i^!(\hat{\mathbf{v}}) = \mathbb{E}_{x} \left[ \Lambda_{\pi(i); \tilde{f}(x)} | \hat{\mathbf{v}} \right] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_{x} \left[ \Lambda_{\pi(i); \tilde{f}(x)} | \hat{\mathbf{v}} - u, u \right] q_i du. \quad (7.26)$$

We also need to introduce the expected payments

$$\tilde{p}_i(\hat{\mathbf{v}}) = \Lambda_{\pi(i); \hat{f}(\hat{\mathbf{v}})} q_i \hat{v}_i - \int_0^{\hat{v}_i} \Lambda_{\pi(i); \hat{f}(\hat{\mathbf{v}}-u)} q_i du,$$

which correspond to the VCG payments except from the use of the estimated allocation function $\hat{f}$ instead of $f^*$. 
Initially, we compute an upper bound over the per–ad regret $r_i = p_i^* - p_i'$ for each step of the exploitation phase and we later use this result to compute the upper bound for the regret over the whole time interval ($R_T$). We divide the per–ad regret in two different components:

$$r_i(\tilde{v}) = p_i^*(\tilde{v}) - p_i'(\tilde{v}) = p_i^*(\tilde{v}) - p_i^{*'}(\tilde{v}) + p_i^{*'}(\tilde{v}) - p_i'(\tilde{v}) = r_i^B(\tilde{v}) + r_i^L(\tilde{v}),$$

where

- $r_i^B(\tilde{v})$ is the regret due to the use of the approach proposed in \[10\] instead of the VCG payments, when all the parameters are known;

- $r_i^L(\tilde{v})$ is the regret due to the uncertainty on the parameters when the payments defined in \[10\] are considered.

For the definitions of $z$ and $\mathbb{E}_{x|z}[\Lambda(\pi(i; f(x)))|\tilde{v}]$ refer to the proof of Thm. \[16\].

**Step 2: the per–ad per–step cSRP regret.** We can reuse the result obtained in the proof of Thm. \[16\]. In particular, we can use the bound in Eq. (7.21), i.e., $r_i^B(\tilde{v}) \leq (K + 1) \mu v_{\max}$. Given that we have assumed $N > K$, in the remaining parts of this proof we will use the following upper bound: $r_i^B(\tilde{v}) \leq (K + 1) \mu v_{\max} \leq N \mu v_{\max}$.

**Step 3: the per–ad per–step learning regret.** Similar to the previous step, we write the learning expected payments based on the cSRP in Eq. (7.26) as

$$\tilde{p}_i'(\tilde{v}) = \mathbb{P}[z = 1] \tilde{p}_i(\tilde{v}) + \mathbb{P}[z \neq 1] \left( \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f(x)))|\tilde{v}] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f(x)))|\tilde{v} - u] q_i du \right).$$

Then the per–ad regret is

$$r_i^L(\tilde{v}) = p_i^{*'}(\tilde{v}) - \tilde{p}_i'(\tilde{v}) = \mathbb{P}[z = 1] \left( p_i^*(\tilde{v}) - \tilde{p}_i(\tilde{v}) \right) + \mathbb{P}[z \neq 1] \left( \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f^*(x)))|\tilde{v}] q_i \hat{v}_i - \int_0^{\hat{v}_i} \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f^*(x)))|\tilde{v} - u] q_i du + \right.$$

$$\leq v_{\max}$$

$$\left. - \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f(x)))|\tilde{v}] q_i \hat{v}_i + \int_0^{\hat{v}_i} \mathbb{E}_{x|z \neq 1} [\Lambda(\pi(i; f(x)))|\tilde{v} - u] q_i du \right) = -r_i^B \leq 0.$$
\[7.3. \text{Unknown } \Lambda_m, m \in K, \text{ and } q_i, i \in N\]

\[\leq p_i^*(\hat{\mathbf{v}}) - \tilde{p}_i(\hat{\mathbf{v}}) + \mathbb{P} [\exists j : z_j = 0] v_{\text{max}} \]

\[\leq p_i^*(\hat{\mathbf{v}}) - \tilde{p}_i(\hat{\mathbf{v}}) + \sum_{j \in N} \mathbb{P} [z_j = 0] v_{\text{max}} = p_i^*(\hat{\mathbf{v}}) - \tilde{p}_i(\hat{\mathbf{v}}) + N \mu v_{\text{max}}.\]

We now simply notice that payments \(\tilde{p}_i\) are WVCG payments corresponding to the estimated allocation function \(\tilde{f}\) and can be written as

\[\tilde{p}_i(\hat{\mathbf{v}}) = \frac{q_i}{\hat{q}_i^+} \left[ \tilde{SW}(f_{-i}(\hat{\mathbf{v}}), \hat{\mathbf{v}}) - \tilde{SW}_{-i}(f(\hat{\mathbf{v}}), \hat{\mathbf{v}}) \right],\]

then, using the result

\[r \leq \sum_{m=1}^{K} \sum_{l=1}^{K} v_{\text{max}} \Delta_l \left( q_{\alpha(m;\tilde{\theta})}^+ - q_{\alpha(m;\tilde{\theta})} \right) \leq v_{\text{max}} \sum_{m=1}^{K} \left( \frac{q_{\alpha(m;\tilde{\theta})}^+}{\hat{q}_i} - \frac{q_{\alpha(m;\tilde{\theta})}}{\hat{q}_i} \right) \sum_{l=1}^{K} \Delta_l,\]

in the proof of Thm. 3.2 (our Thm. 13) in [24], we can conclude that

\[\sum_{i : \pi(i;f^*(\hat{\mathbf{v}}) \leq K)} (p_i^*(\hat{\mathbf{v}}) - \tilde{p}_i(\hat{\mathbf{v}})) \leq 2v_{\text{max}} \eta \left( \sum_{m=1}^{K} \Lambda_m \right) \leq 2K v_{\text{max}} \eta.\]

**Step 4: cumulative regret.** We now bring together the two per–step regrets obtaining that at each step of the the exploitation phase, the mechanism suffers of a regret \(r = \sum_{i=1}^{N} r_i\). We first notice that the expected per–step regret \(r_i\) for each ad \(a_i\) is defined as the difference between the VCG payment \(p_i^*(\hat{\mathbf{v}})\) and the (expected) payments computed by the estimated randomized mechanism \(p_i'(\hat{\mathbf{v}})\). We notice that \(p_i^*(\hat{\mathbf{v}})\) can be strictly positive only for the \(K\) displayed ads, while \(p_i'(\hat{\mathbf{v}}) \geq 0 \forall i \in N\), due to the mechanism randomization. Thus, \(p_i^*(\hat{\mathbf{v}}) - p_i'(\hat{\mathbf{v}}) > 0\) only for at most \(K\) ads. Thus we obtain the per–step regret

\[r \leq \sum_{i : \pi(i;f^*(\hat{\mathbf{v}}) \leq K)} r_i = \sum_{i : \pi(i;f^*(\hat{\mathbf{v}}) \leq K)} (r_i^B + r_i^L) \]

\[\leq KN \mu v_{\text{max}} + \sum_{i : \pi(i;f^*(\hat{\mathbf{v}}) \leq K)} (p_i^*(\hat{\mathbf{v}}) - \tilde{p}_i(\hat{\mathbf{v}}) + N \mu v_{\text{max}}) \]

\[\leq KN \mu v_{\text{max}} + 2K v_{\text{max}} \eta + KN \mu v_{\text{max}} = 2K v_{\text{max}} \eta + 2KN \mu v_{\text{max}}.\]

Finally, the global regret becomes

\[R_T \leq v_{\text{max}} K \left[ (T - \tau) \left( 2\sqrt{\frac{N}{\tau}} \log \frac{2N}{\delta} + 2\mu N \right) + \tau + \delta T \right].\]
Step 5: parameters optimization. We first simplify further the previous bound as
\[
R_T \leq v_{\text{max}} K \left[ T \left( 2\sqrt{\frac{N}{\tau}} \log \frac{2N}{\delta} + 2\mu N \right) + \tau + \delta T \right].
\] (7.28)

Then we optimize the value of \( \tau \), taking the derivative of the previous bound w.r.t. \( \tau \), setting it to zero and obtaining
\[
v_{\text{max}} K \left( -\tau^{-\frac{3}{2}} T \sqrt{N \log \frac{2N}{\delta} + 1} \right) = 0,
\]
which leads to
\[
\tau = T^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}}.
\]
Once replaced into Eq. (7.28) we obtain
\[
R_T \leq v_{\text{max}} K \left[ 3T^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}} + 2T \mu N + \delta T \right].
\]

The optimization of the asymptotic order of the bound can then be obtained by setting \( \mu \) and \( \delta \) so as to equalize the second and third term in the bound. In particular, by setting
\[
\mu = T^{-\frac{1}{3}} N^{-\frac{2}{3}} \quad \text{and} \quad \delta = T^{-\frac{1}{3}} N^{\frac{1}{3}},
\]
we obtain the final bound
\[
R_T \leq 6v_{\text{max}} KT^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log \left( 2N^{\frac{2}{3}} T^{\frac{1}{3}} \right) \right)^{\frac{1}{3}}.
\]

We can summarise the results of this last theorem with the fact that up to numerical constants and logarithmic factors, the previous bound is
\[
R_T \leq \tilde{O}(T^{\frac{2}{3}} K N^{\frac{1}{3}}).\]
Notice that we succeed in matching the lowest possible complexity for the parameter \( T \) when exploration and exploitation phases are separate, i.e., \( \tilde{O}(T^{\frac{2}{3}}) \). Hence, since the per–step regret \( (R_T / T) \) decreases to 0 as \( T^{-\frac{1}{3}} \), the proposed mechanism is a no–regret algorithm, meaning that, asymptotically, our mechanism achieves the same performances of VGC (when all the parameter are known).

We add two additional consideration comparing this bound with the one of Sec. 7.1. The dependence of the cumulative regret in the parameter \( K \).
is augmented by a factor $K^{1/3}$. The reason resides in the exploration phase. In fact, here we cannot take advantage of the samples collected over all the slots, given that we estimate the qualities only on the basis of their visualization in the first slot. Instead, the dependency on $N$ is the same of the one in the case studied in Sec. 7.1.

We conclude the analysis on the regret over the revenue of the auctioneer leaving two open questions.

The first one regards whether or not it is possible to avoid the separation of the exploration and exploitation phases preserving IC in expectation (in some form) and whether or not, in that case, it is possible to obtain a regret of $O(T^{1/2})$. We conjecture that, if it is possible to have $R_T = O(T^{1/2})$ when only $q_i, i \in \mathcal{N}$ are unknown, then it is possible to have $R_T = O(T^{1/2})$ also when both $q_i, i \in \mathcal{N}$ and $\Lambda_m, m \in \mathcal{K}$ are unknown.

The second question is whether it is possible to exploit the samples from the slots below the first one to improve the accuracy of the estimates and to reduce the length of the exploration phase. The critical issue here is that the samples from those slots are about the product of two random variables, i.e., $\Lambda_s$ and $q_i$, and it is not trivial to find a method to use these samples to improve the estimates. However, we notice that the exploitation of these additional samples would correspond to a reduction of the regret bound of at most $K^{1/3}$, given that the dependency from $K$ cannot be better than in the case discussed in Sec. 7.1 (i.e., $O(K^{2/3})$).

We can also prove an upper-bound for the regret on the social welfare of A–VCG3. The derivation is not straightforward w.r.t. the bound over the regret on the payments, but, using the value of the parameters identified in Thm. 20, the bound is $\tilde{O}(T^{2/3})$. Optimizing the parameters w.r.t. to the regret over the social welfare, we obtain the following.

**Theorem 21.** Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, with the separable CTR model with parameters $\Lambda_m, m \in \mathcal{K}$. For any parameter $\tau \in \{0, \ldots, T\}$ and $\delta \in (0, 1)$, the A–VCG3 achieves a regret

$$R_{SW}^T \leq v_{\max} K \left[ (T - \tau)(2\eta + N\mu) + \tau + \delta T \right]$$

$$\leq v_{\max} K \left[ (T - \tau) \left( 2 \sqrt{\frac{N}{\tau} \log \frac{2N}{\delta} + N\mu} \right) + \tau + \delta T \right]$$

For $T > \frac{N}{K^{3/2}}$ we can set the parameters to

$$\mu = K^{-1}N^{1/3}T^{-1/3}$$
\[ \delta = N^{1/3}T^{-1/3} \]
\[ \tau = T^{2/3}N^{1/3} \left( \log \frac{2N}{\delta} \right)^{1/3} \]

and then the regret is
\[ R_T^{SW} \leq 5v_{\text{max}}KN^{1/3}T^{2/3} \left( \log N^{2/3}T^{3} \right)^{1/3}. \]

**Proof.** **Step 1: per-step regret.** We start computing the per-step regret over the SW during the exploitation phase.

First of all we introduce the following definition: \( Z' = \{ z' | z' \in \{0, 1\}^N, \pi(i; f^*(\hat{v})) \leq K \Rightarrow \pi'_i = 1 \} \), i.e., the set of the random realization where the self-resampling procedure does not modify the bids of the ads displayed when the allocation function is \( f^* \) is applied to the true bids \( \hat{v} \).

We now provide the bound over the regret.

\[
\begin{align*}
r & = SW(f^*(\hat{v}), \hat{v}) - E_{\pi} \left[ SW(\tilde{f}(x), \hat{v}) | \hat{v} \right] \\
& = \mathbb{P}[z \in Z'] \left( SW(f^*(\hat{v}), \hat{v}) - E_{\pi|z \in Z'} \left[ SW(\tilde{f}(x), \hat{v}) | \hat{v} \right] \right) + \\
& \quad + \mathbb{P}[z \notin Z'] \left( SW(f^*(\hat{v}), \hat{v}) - E_{\pi|z \notin Z'} \left[ SW(\tilde{f}(x), v) | \hat{v} \right] \right) \\
& \leq SW(f^*(\hat{v}), \hat{v}) - E_{\pi|z \in Z'} \left[ SW(\tilde{f}(x), \hat{v}) | \hat{v} \right] + \\
& \quad + K_{\mu} \left( SW(f^*(\hat{v}), \hat{v}) - E_{\pi|z \notin Z'} \left[ SW(\tilde{f}(x), v) | \hat{v} \right] \right) \\
& \leq SW(f^*(\hat{v}), \hat{v}) - E_{\pi|z \in Z'} \left[ SW(f^*(x), \hat{v}) | \hat{v} \right] + \\
& \quad + E_{\pi|z \in Z'} \left[ SW(f^*(x), \hat{v}) | \hat{v} \right] - E_{\pi|z \in Z'} \left[ \tilde{SW}(\tilde{f}(x), \hat{v}) | \hat{v} \right] + \\
& \quad + E_{\pi|z \in Z'} \left[ \tilde{SW}(\tilde{f}(x), \hat{v}) | \hat{v} \right] - E_{\pi|z \in Z'} \left[ SW(\tilde{f}(x), \hat{v}) | \hat{v} \right] + v_{\text{max}} \mu K^2 \\
& \leq \max_{f \in \mathcal{F}} \left( E_{\pi|z \in Z'} \left[ \tilde{SW}(f(x), \hat{v}) - SW(f(x), \hat{v}) | \hat{v} \right] \right) + v_{\text{max}} \mu K^2
\end{align*}
\]
7.3. Unknown $\Lambda_m, m \in \mathcal{K}$, and $q_i, i \in \mathcal{N}$

\[
\leq \max_{f \in \mathcal{F}} \left( \sum_{j: \pi(j; f(x)) \leq K} \Lambda_{\pi(j; f(x))} v_j (\tilde{q}_j - q_j) \right) + \nu_{\max} \mu K^2 \\
\leq \nu_{\max} \max_{f \in \mathcal{F}} \left( \sum_{j: \pi(j; f(x)) \leq K} (\tilde{q}_j - q_j) \right) + \nu_{\max} \mu K^2 \\
\leq 2 \nu_{\max} K \eta + \nu_{\max} \mu K^2 = \nu_{\max} K (2 \eta + K \mu).
\]

We provide a brief intuition of bounds $r_1$ and $r_2$. The bound $r_1$ can be explained noticing that when the bids of the ads displayed in $f^* (\hat{v})$ are not modified we have that $\alpha(m; f^* (\hat{v})) = \alpha(m; f^* (x))$ where $m \leq K$ and $x$ s.t. $z \in Z'$. The bound for $r_2$ can be understood noticing that when the bids of the ads s.t. $\pi(j; f^* (x)) \leq K$ are not modified and $x_i \leq \hat{v}_i \forall i \in \mathcal{N}$, we obtain $\hat{SW}(f^* (x), \hat{v}) = \hat{SW}(f^* (x), x) \leq \max_{\theta \in \Theta} \hat{SW}(\theta, x) = \hat{SW}(\tilde{f}(x), x) \leq \hat{SW}(\tilde{f}(x), \hat{v})$.

**Step 2: cumulative regret.** We can now compute the upper bound for the global regret

\[
R_{SW}^T \leq \nu_{\max} K \left[ (T - \tau)(2 \eta + K \mu) + \tau + \delta T \right] \\
\leq \nu_{\max} K \left[ (T - \tau) \left( 2 \sqrt{\frac{N}{\tau}} \log \frac{2N}{\delta} + K \mu \right) + \tau + \delta T \right].
\]

**Step 3: parameter optimization.** We first simplify the previous bound as

\[
R_{SW}^T \leq \nu_{\max} K \left[ 2 T \sqrt{\frac{N}{\tau}} \log \frac{2N}{\delta} + K \mu T + \tau + \delta T \right].
\]

Taking the derivative of the previous bound w.r.t. $\tau$ leads to

\[
\nu_{\max} K \left( -\tau^{-\frac{3}{2}} T \sqrt{N \log \frac{2N}{\delta} + 1} \right) = 0,
\]

which leads to

\[
\tau = N^{\frac{1}{3}} T^{\frac{2}{3}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}}.
\]

Once replaced $\tau$ in the bound, we obtain

\[
R_{SW}^T \leq 3 \nu_{\max} K N^{\frac{1}{3}} T^{\frac{2}{3}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}} + \mu K^2 \nu_{\max} T + \delta \nu_{\max} K T.
\]
Finally, we choose $\delta$ and $\mu$ to optimize the asymptotic order by setting

$$\delta = N^{\frac{1}{3}} T^{-\frac{1}{3}}$$

$$\mu = K^{-1} T^{-\frac{1}{3}} N^{\frac{1}{3}}$$

given that $\delta < 1$ this imply that $T > N$ and, given that $\mu < 1$ we have that $T > \frac{N}{K^3}$.

The final bound is

$$R_{SW}^T \leq 5 \cdot v_{\text{max}} K N^{\frac{1}{3}} T^{\frac{2}{3}} \left( \log 2N^{\frac{2}{3}} T^{\frac{1}{3}} \right)^{\frac{1}{3}}.$$
After having devoted a chapter to the case of separable CTR model, we are now ready to move a step up, considering the more complicated Cascade Model, where both position– and ad–dependent externalities are present. Specifically, in Sec. 8.1 we analyze the problem of designing a DSIC mechanism when only the qualities of the ads are unknown. We provide results holding for the CFNE_s_a(K) model that thus is applicable also to the Cascade Model. Then, in Sec. 8.2 we study situations where there is lack of information over other parameters and we observe which new problems rise up.

8.1 Unknown quality $q_i$, $i \in \mathcal{N}$

In this first section we analyze the problem where the only unknown parameters are the qualities $q_i$, $i \in \mathcal{N}$, of the ads and the CFNE_s_a(K) model is adopted. Given we suppose the ad– and position–dependent parameters are known, the bound we provide can be applied to both the situation where ad– and position–dependent externalities are not separate (CFNE_s_a(K)) and when they are (Cascade Model). Notice that, for the same problem, in [24] the authors provide a DSIC mechanism with a bound over the regret
of the auctioneer of \( O(T^{2/3}NK^{2/3}) \), but this bound did not match with their numerical simulations and they conjecture a bound of \( O(T^{2/3}N^{1/3}K^{3/3}) \). Here, with our new proof, we show the conjecture is actually correct.

As we do in Section 7.1, we focus on DSIC mechanisms and we leave open the question whether better bounds over the regret can be found by employing DSIC in expectation. Therefore we study MAB algorithms that separate the exploration and exploitation phases. The structure of the mechanism we propose, called PAD–A–VCG, is similar to the A–VCG1 and is reported in Fig. 11.

Algorithm 11 Pseudo–code for the PAD–A–VCG mechanism.

Input: Length of exploration phase \( \tau \), confidence \( \delta \), ad/position–dependent externalities \( \gamma_{m,i}, m \in K \) and \( i \in N \).

Exploration phase

for \( t = 1, \ldots, \tau \) do
    Allocate ads according to (7.6)
    Ask for no payment
    Observe the clicks \( \{\text{click}^i_{\pi(i;\theta_t)}(t)\}_{i=1}^N \)
end for

Compute the estimated quality \( \tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} \frac{\text{click}^i_{\pi(i;\theta_t)}(t)}{\Gamma_i(\theta_t)} \)

Compute \( \tilde{q}_i^+ = \tilde{q}_i + \eta \) where \( \eta \) is given by (8.1)

Exploitation phase

for \( t = \tau + 1, \ldots, T \) do
    Allocate ads according to \( \tilde{f} \)
    if Ad \( a_i \) is clicked then
        Ask for payment \( \tilde{p}_i^c \) defined in (8.2)
    end if
end for

Exploration phase  During the exploration phase with length \( \tau \leq T \) steps the algorithm collects \( K \) samples of click or no–click events. Given a generic exploration policy \( \{\theta_t\}_{t=0}^\tau \), the estimate quality \( \tilde{q}_i \) is computed as:

\[
\tilde{q}_i = \frac{1}{|B_i|} \sum_{t \in B_i} \frac{\text{click}^i_{\pi(i;\theta_t)}(t)}{\Gamma_i(\theta_t)},
\]

where we identify the set \( B_i = \{t : \pi(i;\theta_t) \leq K, t \leq \tau\} \). Since the explorative allocations \( \theta_t \) impact on the discount \( \Gamma_i(\theta_t) \), we use a variation
of Proposition 19 in which Eq. (7.5) is replaced by:

$$|q_i - \tilde{q}_i| \leq \sqrt{\left( \sum_{t \in B_i} \frac{1}{\Gamma_i(\theta_i)^2} \right) \frac{1}{2|B_i|^2} \log \frac{2N}{\delta}}.$$ 

For each exploration policy such that $|B_i| = \lfloor K\tau/N \rfloor \geq K\tau$ for all $i \in \mathcal{N}$, for instance the policy defined in Eq. (7.6), we redefine $\eta$ as

$$|q_i - \tilde{q}_i| \leq \frac{1}{\Gamma_{\text{min}}} \sqrt{\frac{N}{2K\tau} \log \frac{N}{\delta}} =: \eta, \quad (8.1)$$

where $\Gamma_{\text{min}} = \min_{\theta \in \Theta, i; i \in K, \pi(i;\theta) \leq K} \Gamma_i(\theta)$. We define the upper–confidence bound $\tilde{q}_i^+ = \tilde{q}_i + \eta$. During the exploration phase, in order to preserve the DSIC property, the allocations $\{\theta_t\}_{t=0}^\tau$ do not depend on the reported values of the advertisers and no payments are imposed to the advertisers.

**Exploitation phase** We define the estimated social welfare as

$$\tilde{SW}(\theta, \hat{v}) = \sum_{i=1}^N \Gamma_i(\theta)\tilde{q}_i^+\hat{v}_i = \sum_{m=1}^K \Gamma_{\alpha(m;\theta)}(\theta)\tilde{q}_{\alpha(m;\theta)}^+\hat{v}_{\alpha(m;\theta)}.$$ 

We denote by $\tilde{\theta}$ the allocation maximizing $\tilde{SW}(f(\hat{v}), \hat{v})$ and by $\tilde{f}$ the allocation function returning $\tilde{\theta}$:

$$\tilde{\theta} = \tilde{f}(\hat{v}) \in \arg \max_{\theta \in \Theta} \tilde{SW}(\theta, \hat{v}).$$

Once the exploration phase is over, the ads are allocated on the basis of $\tilde{f}$. Since $\tilde{f}$ is an affine maximizer, the mechanism can impose WVCG payments to the advertisers satisfying the DSIC property. In a pay–per–click fashion the advertiser $a_i$ is charged

$$\tilde{p}_i^c(\hat{v}, click_i^{\pi(i;\theta)} = \frac{\tilde{SW}(\tilde{\theta}_{-i}, \hat{v}_{-i}) - \tilde{SW}(\tilde{\theta}, \hat{v})}{\Gamma_i(\theta)\tilde{q}_i^+}click_i^{\pi(i;\theta)} (8.2)$$

which corresponds, in expectation, to the WVCG payment $E\left[ \tilde{p}_i^c(\hat{v}, click_i^{\pi(i;\theta)}) \right] = \tilde{p}_i(\hat{v})$.

We are interested in bounding the regret of the auctioneer’s revenue due to PAD–A–VCG compared to the auctioneer’s revenue of the VCG mechanism when all the parameters are known.
Chapter 8. Learning with Cascade Model

**Theorem 22.** Let us consider an auction with \( N \) advs, \( K \) slots, and \( T \) steps. The auction has position/ad–dependent externalities and cumulative discount factors \( \Gamma_i(\theta), i \in N, \) and \( \eta \) defined as in Eq. (8.1). For any parameter \( \tau \in \{0, \ldots, T\} \) and \( \delta \in (0, 1) \), the PAD–A–VCG achieves a regret:

\[
R_T \leq v_{\text{max}} K \left[ (T - \tau) \left( \frac{3\sqrt{2n}}{\Gamma_{\text{min}} q_{\text{min}}} \sqrt{\frac{N}{K\tau}} \log \frac{N}{\delta} \right) + \tau + \delta T \right], \quad (8.3)
\]

where \( q_{\text{min}} = \min_{i \in N} q_i \). By setting the parameters to

\[
\delta = K^{\frac{1}{3}} N^{\frac{1}{3}} \left( \frac{5}{\sqrt{2 \Gamma_{\text{min}}}} \right)^{\frac{2}{3}} T^{-\frac{1}{3}},
\]

\[
\tau = \left( \frac{5}{\sqrt{2 \Gamma_{\text{min}}}} \right)^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log N \right)^{\frac{1}{3}},
\]

the regret is

\[
R_T \leq 4v_{\text{max}} K^{\frac{4}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} \frac{5^{\frac{2}{3}}}{2^{\frac{1}{3}} \Gamma_{\text{min}}^{\frac{2}{3}} q_{\text{min}}} \left( \log \frac{2^{\frac{1}{3}} \Gamma_{\text{min}}^{\frac{2}{3}} N^{\frac{2}{3}} T^{\frac{1}{3}}}{K^{\frac{1}{3}} 5^{\frac{1}{3}}} \right)^{\frac{1}{3}}. \quad (8.4)
\]

Before deriving the proof of Thm. 22, we have to introduce and prove two lemmas, where the second one is just the extension of Lemma 1 to the case of CFNE_{sa}(K).

**Lemma 3.** Let \( \mathcal{G} \) be an arbitrary space of allocation functions, then for any \( g \in \mathcal{G} \), when \( |q_i - \hat{q}_i^+| \leq \eta \) with probability \( 1 - \delta \), we have

\[
-2Kv_{\text{max}} \eta \leq SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v}) q_i \hat{q}_i^+ \leq \frac{2Kv_{\text{max}}}{q_{\text{min}}} \eta,
\]

with probability \( 1 - \delta \).

**Proof.** By using the definition of \( SW \) and \( \tilde{SW} \) we have the following sequence of inequalities

\[
\tilde{SW}(g(\hat{v}), \hat{v}) q_i \hat{q}_i^+ - SW(g(\hat{v}), \hat{v}) = \sum_{j: \pi(j; g(\hat{v})) \leq K} \Gamma_j(g(\hat{v})) \hat{v}_j \left( \hat{q}_j^+ q_i q_j - q_i^+ \right)
\]
8.1. Unknown quality \( q_i, i \in \mathcal{N} \)

\[
\begin{align*}
&\leq v_{\text{max}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left( \frac{q_{j}^{+}}{q_{i}^{+}} - q_{j} \right) \\
&\leq v_{\text{max}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left( \tilde{q}_{j}^{+} - q_{j} \right) \leq 2K v_{\text{max}} \eta.
\end{align*}
\]

The second statement follows from

\[
SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v}) \frac{q_{i}}{\hat{q}_{i}^{+}}
\]

\[
\begin{align*}
&\leq \sum_{j: \pi(j; g(\hat{v})) \leq K} \Gamma_{j}(g(\hat{v})) \hat{\nu}_{j} \left( q_{j} - \tilde{q}_{j}^{+} \frac{q_{i}}{\hat{q}_{i}^{+}} \right) \\
&\leq v_{\text{max}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left( q_{j} - q_{j} \frac{q_{i}}{\hat{q}_{i}^{+}} + q_{j} \frac{q_{i}}{\hat{q}_{i}^{+}} - \tilde{q}_{j}^{+} \frac{q_{i}}{\hat{q}_{i}^{+}} \right) \\
&= v_{\text{max}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left[ q_{j} \left( \frac{\tilde{q}_{i}^{+} - q_{i}}{\hat{q}_{i}^{+}} \right) + \left( q_{j} - \tilde{q}_{j}^{+} \right) \frac{q_{i}}{\hat{q}_{i}^{+}} \right] \\
&\leq \frac{v_{\text{max}}}{q_{\text{min}}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left( \tilde{q}_{i} - q_{i} + \eta \right) \leq \frac{2K v_{\text{max}} \eta}{q_{\text{min}}}. \quad \square
\end{align*}
\]

**Lemma 4.** Let \( G \) be an arbitrary space of allocation functions, then for any \( g \in G \), when \(|q_{i} - \tilde{q}_{i}^{+}| \leq \eta \) with probability \( 1 - \delta \), we have

\[
0 \leq \left( \tilde{SW}(g(\hat{v}), \hat{v}) - SW(g(\hat{v}), \hat{v}) \right) \leq 2K v_{\text{max}} \eta,
\]

with probability \( 1 - \delta \).

**Proof.** The first inequality follows from

\[
SW(g(\hat{v}), \hat{v}) - \tilde{SW}(g(\hat{v}), \hat{v})
\]

\[
\begin{align*}
&= \sum_{j: \pi(j; g(\hat{v})) \leq K} \Gamma_{j}(g(\hat{v})) \hat{\nu}_{j} \left( q_{j} - \tilde{q}_{j}^{+} \right) \\
&\leq v_{\text{max}} \sum_{j: \pi(j; g(\hat{v})) \leq K} \left( q_{j} - \tilde{q}_{j}^{+} \right) \leq 0,
\end{align*}
\]

while the second inequality follows from

\[
\tilde{SW}(g(\hat{v}), \hat{v}) - SW(g(\hat{v}), \hat{v})
\]

127
Chapter 8. Learning with Cascade Model

\[ = \sum_{j: \pi(j; g(\hat{v})) \leq K} \Gamma_j(g(\hat{v})) \hat{v}_j (\hat{q}_j - q_j) \]

\[ \leq v_{\max} \sum_{j: \pi(j; g(\hat{v})) \leq K} (\hat{q}_j^* - q_j) \]

\[ = v_{\max} \sum_{j: \pi(j; g(\hat{v})) \leq K} (\hat{q}_j + \eta - q_j) \leq 2Kv_{\max}\eta. \]

We are now ready to proceed with the proof of Thm. 22.

**Proof.** **Step 1: per–ad per–step regret.** We first compute the per–step per–ad regret \( r_i = p_i^*(\hat{v}) - \tilde{p}_i(\hat{v}) \) at each step of the exploitation phase for each ad \( a_i \). According to the definition of payments we have

\[
r_i = SW(f_{-i}(\hat{v}), \hat{v}) - \tilde{SW}(\tilde{f}_{-i}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} + \tilde{SW}(\tilde{f}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} - SW_{-i}(f^*(\hat{v}), \hat{v}).
\]

We bound the first term through Lemma 3 and the following inequalities

\[
r_i^1 = SW(f_{-i}(\hat{v}), \hat{v}) - \tilde{SW}(f_{-i}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} + \tilde{SW}(f_{-i}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} - \tilde{SW}(\tilde{f}_{-i}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*}
\]

\[
\leq \max_{f \in \mathcal{F}_{-i}} \left( SW(f(\hat{v}), \hat{v}) - \tilde{SW}(f(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} \right) +
\]

\[
+ \left( \tilde{SW}(f_{-i}(\hat{v}), \hat{v}) - \max_{f \in \mathcal{F}_{-i}} \tilde{SW}(\tilde{f}(\hat{v}), \hat{v}) \right) \frac{q_i}{q_i^*}
\]

\[
\leq 2Kv_{\max} \frac{\eta}{\eta_{\min}},
\]

with probability \( 1 - \delta \). We rewrite \( r_i^2 \) as

\[
r_i^2 = \left( SW(\tilde{f}(\hat{v}), \hat{v}) - \Gamma_i(\tilde{f}(\hat{v})) \tilde{q}_i + \hat{v}_i \right) \frac{q_i}{q_i^*} - SW(f^*(\hat{v}), \hat{v}) + \Gamma_i(f^*(\hat{v})) q_i \hat{v}_i
\]

\[
= \tilde{SW}(\tilde{f}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} - SW(f^*(\hat{v}), \hat{v}) + \left( \Gamma_i(f^*(\hat{v})) - \Gamma_i(\tilde{f}(\hat{v})) \right) q_i \hat{v}_i.
\]

We now focus on the term \( r_i^3 \) and use Lemma 3 to bound it as

\[
r_i^3 = \tilde{SW}(\tilde{f}(\hat{v}), \hat{v}) \frac{q_i}{q_i^*} - SW(\tilde{f}(\hat{v}), \hat{v}) + SW(\tilde{f}(\hat{v}), \hat{v}) - \max_{f \in \mathcal{F}} SW(f(\hat{v}), \hat{v}) \leq 0
\]

128
8.1. Unknown quality $q_i, i \in \mathcal{N}$

$$\leq \max_{f \in \mathcal{F}} \left( \tilde{SW}(f(\hat{\nu}), \hat{\nu}) \frac{q_i}{\tilde{q}_i} - SW(f(\hat{\nu}), \hat{\nu}) \right)$$

$$\leq 2Kv_{\max}\eta.$$ 

**Step 2: exploitation and cumulative regret.** We define $I = \{i \in \mathcal{N} | \pi(i; f^*(\hat{\nu})) \leq K \lor \pi(i; \hat{f}(\hat{\nu})) \leq K\}, |I| \leq 2K$. It is clear that only the ads $a_i$ s.t. $i \in I$ have a regret $r_i \neq 0$. The other ads, $i \notin I$, have both $p^*_i(\hat{\nu}) = 0$ and $\tilde{p}_i(\hat{\nu}) = 0$. Thus, we can bound the regret $r$, at each exploitative step, in the following way

$$r = \sum_{i \in I} (r^1_i + r^2_i)$$

$$\leq \sum_{i \in I} \left( \frac{2Kv_{\max}}{q_{\min}} \eta + 2Kv_{\max}\eta \right) + \sum_{i \in I} \left( \Gamma_i(f^*(\hat{\nu})) - \Gamma_i(\hat{f}(\hat{\nu})) \right) q_i \hat{\nu}_i$$

$$= \sum_{i \in I} \left( \frac{2Kv_{\max}}{q_{\min}} \eta + 2Kv_{\max}\eta \right) + \sum_{i = 1}^N \left( \Gamma_i(f^*(\hat{\nu})) - \Gamma_i(\hat{f}(\hat{\nu})) \right) q_i \hat{\nu}_i$$

$$\leq 8K^2v_{\max}\eta + \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) - SW(\hat{f}(\hat{\nu}), \hat{\nu})$$

$$= \frac{8K^2v_{\max}}{q_{\min}} \eta + \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) - \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) +$$

$$+ \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) - \max_{f \in \mathcal{F}} \tilde{SW}(f) + \tilde{SW}(\hat{f}(\hat{\nu}), \hat{\nu}) - SW(\hat{f}(\hat{\nu}), \hat{\nu}) \leq 0$$

$$\leq 8K^2v_{\max}\eta + \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) - \tilde{SW}(f^*(\hat{\nu}), \hat{\nu}) +$$

$$+ \tilde{SW}(\hat{f}(\hat{\nu}), \hat{\nu}) - SW(\hat{f}(\hat{\nu}), \hat{\nu}).$$

The remaining terms $r^1$ and $r^2$ can be easily bounded using Lemma 4 as

$$r^1 \leq 0 \quad \text{and} \quad r^2 \leq 2Kv_{\max}\eta.$$ 

Summing up all the terms we finally obtain

$$r \leq \frac{10K^2v_{\max}}{q_{\min}} \eta$$

with probability $1 - \delta$. Now, considering the per–step regret of the exploration and exploitation phases, we obtain the final bound on the cumulative
regret $R_T$ as follows

$$R_T \leq v_{\text{max}} K \left[ (T - \tau) \left( \frac{10 K}{\Gamma_{\text{min}} q_{\text{min}}} \sqrt{\frac{N}{2 K \tau}} \log \frac{N}{\delta} \right) + \tau + \delta T \right].$$

**Step 3: parameter optimization.** Let $c := \frac{5}{\sqrt{2 \Gamma_{\text{min}} q_{\text{min}}}}$, then we first simplify the previous bound as

$$R_T \leq v_{\text{max}} K \left[ 2 c T \sqrt{NK \log \frac{N}{\delta}} + \tau + \delta T \right].$$

Taking the derivative w.r.t. $\tau$ leads to

$$v_{\text{max}} K \left( - \tau^{-\frac{3}{2}} c T \sqrt{N K \log \frac{N}{\delta}} + 1 \right) = 0,$$

which leads to

$$\tau = c^2 T^2 \frac{2}{3} K \frac{1}{3} N^{\frac{1}{3}} \left( \log \frac{N}{\delta} \right)^{\frac{1}{3}}.$$

Once replaced in the bound, we obtain

$$R_T \leq v_{\text{max}} K \left[ 3 T^2 \frac{2}{3} c^2 N^{\frac{1}{3}} K^{\frac{1}{3}} \left( \log \frac{N}{\delta} \right)^{\frac{1}{3}} + \delta T \right].$$

Finally, we choose $\delta$ to optimize the asymptotic order by setting

$$\delta = K^{\frac{1}{3}} N^{\frac{1}{3}} c^2 T^{-\frac{1}{3}},$$

which leads to the final bound

$$R_T \leq 4 v_{\text{max}} K^{\frac{4}{3}} c^2 T^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log \frac{N^{\frac{2}{3}} T^{\frac{1}{3}}}{K^{\frac{1}{3}} c^{\frac{2}{3}}} \right)^{\frac{1}{3}}.$$

Notice that this bound imposes constraints on the value of $T$, indeed, $T > \tau$, thus $T > c^3 K^{\frac{1}{3}} T^{\frac{2}{3}} N^{\frac{1}{3}} \left( \log \frac{N^{\frac{2}{3}}}{\delta} \right)^{\frac{1}{3}}$ and $\delta < 1$, thus $T > c^2 K N$, leading to:

$$T > c^2 K N \max \left\{ \log \frac{N}{\delta}, 1 \right\}.$$

The problem associated with the previous bound is that $\tau$ and $\delta$ depends on $q_{\text{min}}$, which is an unknown quantity. Thus actually choosing this values to optimize the bound may be unfeasible. An alternative choice of $\tau$ and $\delta$
8.1. Unknown quality $q_i, i \in \mathcal{N}$

is obtained by optimizing the bound removing the dependency on $q_{\text{min}}$. Let $d := \frac{5}{\sqrt{2T_{\text{min}}}}$, then we choose

$$\tau = d^2 K^\frac{1}{2} T^\frac{2}{3} N^\frac{1}{3} \left( \log \frac{N}{\delta} \right)^{\frac{1}{3}},$$

and

$$\delta = K^{\frac{1}{3}} N^{\frac{1}{3}} d^2 T^{-\frac{1}{3}},$$

which leads to the final bound

$$R_T \leq 4v_{\text{max}} K^\frac{4}{3} T^\frac{2}{3} N^\frac{1}{3} \frac{d^2}{q_{\text{min}}} \left( \log \frac{N^\frac{2}{3} T^{\frac{1}{3}}}{K^\frac{1}{3} d^2} \right)^{\frac{1}{3}}$$

under the constraint that $T \geq KN d^2$. \hfill \Box

Up to constants and logarithmic factors, the previous bound is $R_T \leq \tilde{O}(T^\frac{2}{3} N^\frac{1}{3} K^\frac{4}{3})$. We first notice that moving from position– to position/ad–dependent externalities does not change the dependency of the regret on both the number of steps $T$ and the number of ads $N$. Thus, the per–step regret still decreases to 0 as $T$ increases. The main difference w.r.t. the bound in Thm. 13 is in the dependency on $K$ and on the smallest quality $q_{\text{min}}$. We believe that the augmented dependence in $K$ is mostly due to an intrinsic difficulty of the position/ad–dependent externalities. The intuition is that now, in the computation of the payment for each ad $a_i$, the errors in the quality estimates cumulate through the slots (unlike the separable CTR case where they are scaled by $\Lambda_k - \Lambda_{k+1}$). This cumulated error should impact only on a portion of the ads (i.e., those which are actually displayed according to the optimal and the estimated optimal allocations) whose cardinality can be upper–bounded by $2K$. Thus we observe that the bound shows a super–linear dependency in the number of slots. The other main difference is that now the regret has an inverse dependency on the smallest quality $q_{\text{min}}$. Inspecting the proof, this dependency appears because the error of a quality estimation for an ad $a_i$ might be amplified by the inverse of the quality itself $1/q_i$. This dependency might follow from that fact the we have a distribution–free (or worst–case) bound, i.e the bound holds for any set of advertisers. This generality comes at the price that the bound could be inaccurate for some specific sets of advertisers. On the other hand, distribution–dependent bounds (see e.g., the bounds of UCB [8]), where $q$ and $v$ appear explicitly, would be more accurate in predicting the behavior of the algorithm. Nonetheless, they could not be
used to optimize the parameters $\delta$ and $\tau$, since they would then depend on unknown quantities. In in Section [9] through numerical simulations, we investigate whether this dependency on $q_{\min}$ is an artifact of the proof or it is intrinsic in the algorithm.

We also notice that the above bound does not reduce to the bound in Eq. (7.13) in which only position–dependent externalities are present even disregarding the constant terms. Indeed, the dependency on $K$ is different in the two bounds: in Eq. (7.13) we have $K^{\frac{2}{3}}$ while in Eq. (8.4) we have $K^{\frac{4}{3}}$. This means that the bound in Eq. (8.4) over–estimates the dependency on $K$ whenever the auction has only position–dependent externalities. It is an interesting open question whether it is possible to derive an auction–dependent bound where the specific values of the discount factors $\gamma_{m,i}$ explicitly appear in the bound and which reduces to Eq. (7.13) for position–dependent externalities.

Our final remark is that even if we are considering an environment where $q_i, i \in \mathcal{N}$, are unknown, if, at least, a guess about the value of $q_{\min}$ is available, it could be used to better tune $\tau$ by multiplying it by $(q_{\min})^{\frac{2}{3}}$, thus reducing the regret from $\mathcal{O}((q_{\min})^{-1})$ to $\mathcal{O}((q_{\min})^{-\frac{2}{3}})$.

Adopting the same mechanism described before, it is also possible to derive an upper–bound over the cumulative regret over the social welfare of the allocation (as in [10]). We obtain the same dependence over $T$, as for the regret on the payment. Thus $R_{SW}^T \leq \tilde{O}(T^{\frac{2}{3}})$. In particular notice that PAD–A–VCG is a zero–regret algorithm.

**Theorem 23.** Let us consider an auction with $N$ advs, $K$ slots, and $T$ steps. The auction has position/ad–dependent externalities and cumulative discount factors $\Gamma_i(\theta), i \in \mathcal{N}$, and $\eta$ defined as in Eq. (8.1). For any parameter $\tau \in \{0, \ldots, T\}$ and $\delta \in (0, 1)$, the PAD–A–VCG achieves a regret:

$$R_{SW}^T \leq v_{\max}K \left[ (T - \tau) \frac{2}{\Gamma_{\min}} \sqrt{\frac{N}{2K\tau \log \frac{N}{\delta}}} + \tau + \delta T \right],$$

By setting the parameters to

$$\delta = \frac{\sqrt{2}}{\Gamma_{\min}} K^{-\frac{1}{3}} N^{\frac{1}{3}} T^{-\frac{1}{3}}$$

$$\tau = \frac{\sqrt{2}}{\Gamma_{\min}} T^{\frac{2}{3}} N^{\frac{1}{3}} K^{-\frac{1}{3}} \left( \log \frac{2N}{\delta} \right)^{\frac{1}{3}},$$
the regret is
\[
R_{SW}^{ST} \leq 4v_{\max}\left(\frac{\sqrt{2}}{\Gamma_{\min}}\right)^{\frac{2}{3}} K^{\frac{2}{3}} N^{\frac{1}{3}} T^{\frac{2}{3}} \left(\log 2^{\frac{2}{3}} \Gamma_{\min}^{-\frac{2}{3}} N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{1}{3}}\right)^{\frac{1}{3}}.
\] (8.6)

Proof. Step 1: cumulative regret. We apply Lemma 2 to the model with position– and ad–dependent externalities with \( q_i, i \in N \), unknowns, obtaining
\[
R_{SW}^{ST} \leq v_{\max} K [2(T - \tau) \eta + \tau + \delta T]
\]
\[
\leq v_{\max} K \left[2(T - \tau) \frac{\sqrt{2}}{\Gamma_{\min}} \sqrt{\frac{N}{K_\tau}} \log \frac{2N}{\delta} + \tau + \delta T\right].
\]

Step 2: parameter optimization. First we notice that adopting the value of the parameters identified in Thm. 22 we obtain an upper bound \( \tilde{O}(T^{\frac{2}{3}}) \) for the global regret \( R_{SW}^{ST} \).

In order to find values that better optimize the bound over \( R_{SW}^{ST} \), it is possible to use the procedure followed in the proof of Thm. 14 with \( \epsilon := \frac{\sqrt{2}}{\Gamma_{\min}} \):
\[
R_{SW}^{ST} \leq 4v_{\max} \epsilon^{\frac{2}{3}} K^{\frac{2}{3}} N^{\frac{1}{3}} T^{\frac{2}{3}} \left(\log 2 e^{-\frac{2}{3}} N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{1}{3}}\right)^{\frac{1}{3}}.
\]

When using \( \tau \) and \( \delta \) defined in Thm. 22, the bound for \( R_{SW}^{ST} \) is \( \tilde{O}(T^{\frac{2}{3}}) \), even if the parameters are not optimal for this second framework. We notice that unlike the bound on the revenue regret, in this case \( R_{SW}^{ST} \) does not display any dependency on \( q_{\min} \), which suggests that the problem of minimizing the social welfare regret may be easier. Roughly speaking, this is due to the fact that the accuracy of the estimated qualities is only used to determine the allocation \( \tilde{f} \) but they do not determine the performance of \( \tilde{f} \) itself, which is measured according to its actual social welfare. On the other hand, in the computation of the revenue regret, the qualities \( \tilde{q}_i^{+} \) are used to determine the payments and this may lead to an additional error which is reflected in the presence of \( q_{\min} \) in the bound.

8.2 Further extensions

In this section we provide a negative result, in terms of regret, under DSIC truthfulness when the parameter \( \gamma_{m,i} \) depends only on the ad \( a_i \), i.e., when
we are considering the only ad–dependent model and the parameters $\gamma_i$, $i \in \mathcal{N}$, are the only unknown ones.

We focus on the exploitation phase, supposing the exploration phase has produced the estimates $\tilde{\gamma}_i^+$, $i \in \mathcal{N}$, for the continuation probabilities $\gamma_i$, $i \in \mathcal{N}$. The allocation function $f$ presented in [35] is able to compute the optimal allocation when $\gamma_i$, $i \in \mathcal{N}$, values are known, but it is not an affine maximizer when applied to the estimated values $\tilde{\gamma}_i^+$, $i \in \mathcal{N}$. This allocation $\tilde{f}$ is defined as:

$$\tilde{f}(\hat{v}) \in \arg \max_{\theta \in \Theta} \sum_{m=1}^{K} q_{\alpha(m;\theta)} \hat{v}_{\alpha(m;\theta)} \prod_{h=1}^{m-1} \tilde{\gamma}_{\alpha(h;\theta)}^+. \quad (8.7)$$

In this case, a weight depending only on a single ad cannot be isolated and therefore we cannot formulate $\tilde{f}$ as an affine maximizer. Furthermore, we show also that this allocation function is not monotone.

**Proposition 26.** The allocation function $\tilde{f}$ is not monotone.

**Proof.** The proof is by counterexample. Consider an environment with 3 ads and 2 slots such that:

<table>
<thead>
<tr>
<th>ad</th>
<th>$v_i$</th>
<th>$\tilde{\gamma}_i^+$</th>
<th>$\gamma_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>0.85</td>
<td>1</td>
<td>0.89</td>
</tr>
<tr>
<td>$a_2$</td>
<td>1</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>$a_3$</td>
<td>1.4</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

and $q_i = 1 \ \forall i \in \mathcal{N}$. The optimal allocation $\hat{\theta}$ found by $\tilde{f}$ when agents declare their true values $v$ is: ad $a_2$ is allocated in the first slot and $a_3$ in the second one. We have $CTR_3(\hat{\theta}) = 0.9$.

If advertiser $a_3$ reports a larger value: $\hat{v}_3 = 1.6$, in the allocation $\hat{\theta}$ found by $\tilde{f}(\hat{v}_3, v_{-3})$, ad $a_1$ is displayed into the first slot and $a_3$ into the second one. In this case $CTR_3(\hat{\theta}) = 0.89 < CTR_3(\hat{\theta})$, thus the allocation function $\tilde{f}$ is not monotone. \qed

On the basis of the above result, we can state the following theorem.

**Theorem 24.** Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, with the only ad–dependent model with parameters $\gamma_i$, $i \in \mathcal{N}$, whose value are unknown. Any online learning DSIC mechanism achieves an expected regret $R_T^{SW} = \Theta(T)$ over the social welfare.
8.2. Further extensions

**Proof.** Call $f(\hat{v}|\gamma)$ the allocation function maximizing the social welfare given parameters $\gamma$. As shown above, $f(\hat{v}|\tilde{\gamma})$ cannot be adopted in the exploitation phase, the mechanism would not be DSIC otherwise. However, it can be easily observed that a necessary condition to have a no–regret algorithm is that the allocation function used in the exploitation phase, say $g(\hat{v}|\tilde{\gamma})$, is such that $g(\hat{v}|\gamma) = f(\hat{v}|\gamma)$ for every $\hat{v}$ and $\gamma$ (that is, they always return the same allocation) given that $\tilde{\gamma}$ are consistent estimates and $\tilde{\gamma} \to \gamma$ as $T \to +\infty$. Otherwise, since allocations are finite and the difference between the values of the allocations is generically strictly positive, the algorithm would suffer from a strictly positive regret when $T \to +\infty$ and therefore it would not be a no–regret mechanism. However, any such a $g$ would not be monotone and therefore it cannot be adopted in a DSIC mechanism. As a result, any online learning DSIC mechanism is not a no–regret mechanism. To complete the proof, we need to provide a mechanism with regret $\Theta(T)$. Such a mechanism can be easily obtained by partitioning ads in groups such that in each group the ads compete only for a single slot. Therefore, each ad can appear in only one slot.

The above result shows that no approach similar to the approach described in [10] can be adopted even for DSIC in expectation. Indeed, the approach described in [10] requires in input a monotone allocation function. This would suggest a negative result in terms of regret also with DSIC in expectation. However, here, we leave the study of this case open.

Finally, we provide a result on the regret over the auctioneer’s revenue, whose proof is straightforward given that the (W)VCG cannot be adopted due to the above result and therefore the regret over the payments cannot go to zero as $T$ goes to $\infty$.

**Theorem 25.** Let us consider an auction with $N$ advertisers, $K$ slots, and $T$ steps, with the only ad–dependent model with parameters $\gamma_i$, $i \in \mathcal{N}$ whose value are unknown. Any online learning DSIC mechanism achieves an expected regret over the auctioneer’s revenue $R_T = \Theta(T)$.

Given that the only ad–dependent model is a specialization of the Cascade Model, the negative results of this section can be directly extended to the Cascade Model.
We close this second part of the thesis, devoted to learning mechanisms, providing a thorough numerical analysis to validate the theoretical bounds over the regret of the auctioneer’s revenue presented in the previous two chapters. In particular, we analyze the accuracy with which our bounds predict the dependency of the regret w.r.t. the main parameters of the auctions such as $T$, $N$, $K$, and $q_{\text{min}}$. In all the experimental settings, we generate the parameters related to the ads in the same way. The qualities $q_i$, $i \in \mathcal{N}$, are drawn from a uniform distribution in $[0.01, 0.1]$, while the values $v_i$, $i \in \mathcal{N}$, are randomly drawn from a uniform distribution on $[0, 1] (v_{\text{max}} = 1)$. Since the main objective is to evaluate the accuracy of the bounds, we report the relative regret

$$R_T = \frac{R_T}{B(T, K, N, q_{\text{min}}, \Gamma_{\text{min}})},$$

where $B(T, K, N, q_{\text{min}}, \Gamma_{\text{min}})$ is the value of the bound for the specific setting (i.e., Eq. (7.13) and Eq. (7.25) for separable CTR model, and Eq. (8.4) for position/ad–dependent externalities, CFNE$^\text{sa}(K)$ and Cascade Model). We analyze the accuracy of the bound w.r.t. each specific parameter, changing only its value and keeping the values of all the others fixed. Since $B$ is proved to be an upper–bound on the real regret $R_T$, we expect the relative
regret $\overline{R}_T$ to be always smaller than 1. All the results presented in the following sections are obtained by setting $\tau$ and $\delta$ as suggested by the bounds derived in the previous sections and, where it is not differently specified, by averaging over 100 independent runs.

9.1 Learning with Separable CTR

9.1.1 Unknown $\Lambda_m, m \in \mathcal{K}$

![Figure 9.1: Position–dependent externalities with unknown $\Lambda_m, m \in \mathcal{K}$. Dependency of the relative regret on $T$ and $K$.](image)

We start the numerical simulation section investigating the accuracy of the bound derived for algorithm A–VCG2' presented in Sec. 7.2.2. We used several probability distributions to generate the values of $\lambda_m, m \in \mathcal{K}$. We observed that, when they are drawn uniformly from the interval $[0.98, 1.00]$, the numerical simulations confirm our bound (as we show below), whereas, in our studies we observed that the bound seems to overestimate the dependences over $K$ and $\mu$ when the support of the probability distribution is wider (e.g., $[\xi, 1.00]$ with $\xi \ll 0.98$).

The left plot of Fig. 9.1 shows the dependence of the ratio $\overline{R}_T$ w.r.t. $T$ when $\mu = 0.01$. Despite the noise, the ratio seems not to be affected by the variation of $T$, confirming our bound. In the right plot of Fig. 9.1 we observe that when $T = 10^5$ and $\mu = 0.01$ the behavior of the ratio as $K$ changes is essentially the same for different values of $N$. Furthermore, we observe that the bound is accurate except that it seems to overestimate the dependence when $K$ assumes small values (as it happens in practice). In the left plot of Fig. 9.2 the ratio $\overline{R}_T$ seems to be constant as $\mu$ varies when $T = 10^5$.

We conclude our analysis studying the variance of the payments as $\mu$
9.1 Learning with Separable CTR

Figure 9.2: Separable CTR model with unknown $\Lambda_m, m \in \mathcal{K}$. Dependency of the relative regret on $\mu$. Variance of the revenue of the auctioneer.

varies. The bound over $R_T$, provided in Sec. 7.2.2, suggests to choose a $\mu \to 0$ in order to reduce the regret. Nonetheless, the regret bounds are obtained in expectation w.r.t. all the sources of randomization (including the mechanism) and do not consider the possible deviations. Thus in the right plot of Fig. 9.2 we investigate the variance of the payments. The variance is excessively high for small values of $\mu$, making the adoption of these value inappropriate. Thus, the choice of $\mu$ should consider both these two dimensions of the problem: the regret and the variance of the payments.

9.1.2 Unknown $\Lambda_m, m \in \mathcal{K}$, and $q_i, i \in \mathcal{N}$

In this section we analyze the bound provided in Sec. 7.3 for position-dependent auctions where both the prominences and the qualities are unknown. For these simulations we generate $\lambda_m, m \in \mathcal{K}$ samples from a uniform distribution over $[0.5, 1]$. In the simulations we adopted the values of $\tau, \delta$ and $\mu$ derived for the bound. In particular, in order to balance the increase of variance of the payments when $\mu$ decreases, the number of steps is not constant, but it changes as a function of $\mu$ as $\frac{1000}{\mu}$. This means that, in expectation, the bid of a generic ad $a_i$ is modified 1000 times over the number of the steps.

In the plots of Fig. 9.3, we show that the bound in Eq. (7.25) accurately predicts the dependence of the regret w.r.t. the parameters $T$ and $N$. Indeed, except for the noise due to the high variance of the payments based on the cSRP, the two plots show that fixing the other parameters, the ratio $R_T$ is constant as $T$ and $N$ increase, respectively.

The plot in Fig. 9.4 represents the dependency of the relative regret w.r.t. the parameter $K$. We can deduce that the bound $R_T$ over-estimate the
dependency on $K$ for small values of the parameters, while, with larger values, the bound accurately predicts the behavior, the curves being flat.

### 9.2 Learning with Cascade Model

We now move to the analysis of the bound provided in Sec. 8.1 for auctions with position–dependent and ad–dependent externalities where only the qualities are unknown.

In the bound provided in Thm. 22 the regret $R_T$ presents a linear dependency on $N$ and an inverse dependency on the smallest quality $q_{\text{min}}$. The relative regret $\overline{R}_T$ is now defined as $R_T/B$ where $B$ is the bound Eq. (8.4). In the left plot of Fig. 9.5 we report $\overline{R}_T$ as $T$ increases. As it can be observed, the bound accurately predicts the behavior of the regret w.r.t. $T$ as in the case of position–dependent externalities. In the right plot of Fig. 9.5...
9.2. Learning with Cascade Model

![Figure 9.5: Dependency on T and q_{min} in auctions with position/ad–dependent externalities.](image)

![Figure 9.6: Dependency of the relative regret R_T on N.](image)

we report \( R_T \) as we change \( q_{min} \). According to the bound in Eq. (8.4) the regret should decrease as \( q_{min} \) increases (i.e., \( R_T \leq \tilde{O}(q_{min}^{-1}) \)) but it is clear from the plot that \( R_T \) has a much smaller dependency on \( q_{min} \), if any. Indeed from this experiment is not clear whether \( R_T = \tilde{O}(q_{min}) \), thus implying that \( R_T \) does not depend on \( q_{min} \) at all, or \( R_T \) is sublinear in \( q_{min} \), which would correspond to a dependency \( R_T = \tilde{O}(q_{min}^{-f}) \) with \( f < 1 \). Finally, in Fig. 9.6, we study the dependency on \( N \). In this case \( R_T \) slightly increases and then it tends to flat as \( N \) increases. This result suggests that the, theoretically derived, \( N^{\frac{1}{3}} \) dependency of \( R_T \) w.r.t. the number of ads might be correct. We do not report results on \( K \) since the complexity of finding the optimal allocation \( f^* \) becomes intractable for values of \( K \) larger than 8, as we have shown in [23], making the empirical evaluation of the bound impossible.
Conclusions and Future Works

In the dissertation we focused on the online pay–per–click/visit advertising problems, specifically, our goal was to study them from all the four main perspectives that are crucial for the success of an economic mechanism: computational complexity of finding the optimal solution, incentive compatibility, exact and theoretically bounded approximation algorithms, and online learning. With this goal in mind, in the previous chapters we studied different advertising scenarios each characterised by a particular way the user interacts with the slots where ads are displayed.

Different applications require different user models, for this reason, in Chapter 3 we analysed the weaknesses of the models currently available in the state of the art (separable CTR and Cascade Model) and we proposed three new models: CFNE\textsubscript{aa}, CFNE\textsubscript{sa} and CFNE–q\textsubscript{i,m}. We also highlighted the relations among the different models, relations that we used when providing computational complexity and online learning results.

In the CFNE\textsubscript{aa} model we introduced a contextual graph that generalises the ad–dependent externality of the Cascade Model, characterised by a parameter \(\gamma_i\). The limitation of \(\gamma_i\) is that the CTR discount applied on the following allocated ad \(a_j\) depends only on the ad \(a_i\) and not on the one allocated in the next slot \((a_j)\). The introduction of the contextual graph over-
Chapter 10. Conclusions and Future Works

comes this limitation. A different limitation of the Cascade Model is the
fact that the ad– and position–dependent externalities are considered inde-
pendent, while in some real applications these two factors are not separated,
but simultaneously condition a single parameter $\gamma_{m,i}$. The CFNE$_{sa}$ captures
exactly this situation. Finally, we noticed that, in the Cascade Model, the
quality of an ad is independent w.r.t. the slot in which it is allocated. In
some application, different from the SSAs, this could be a limitation, in-
deed there are cases where the quality depends also on the slot. We recall
the example of advertising, like coupons, related to shops, where the qual-
ity of the ad depends on the distance place where user receives the coupon
and the location of the corresponding shop. This last limitation is overcome
by the CFNE$_{q_{i,m}}$ model.

We also highlighted an additional limitation of the Cascade Model. In
this model the memory of the user is supposed to be unlimited, i.e., an ad
influences the CTR of all the ads allocated after it. The fact that the memory
of the user is limited is supported by some experimental studies. Thus, for
the models CFNE$_{aa}$ and CFNE$_{sa}$, the two main models we focused on in
the thesis, we introduced this new dimension, allowing the length of the
memory to be a parameter of the models.

The last model we introduced is the Mobile Advertising model. It is
an economic model for mobile geo–location advertising. Specifically, we
designed a user mobility model whereby the user moves along one of sev-
eral paths, and where the quality of ads depends on the actual path the user
follows and on the node of the path at which the user receives the ad. In
addition, we captured ad fatigue by decreasing ad relevance as the user re-
ceives more ads.

In Chapters 4 and 5, we provided computational complexity and algo-
rithmic results on the CFNE$_{aa}(c)$ and CFNE$_{sa}(c)$ models. In particular, in
Chapter 4 we studied the case where the user memory $c$ is strictly lower
than $K$ and in Chapter 5 the case $c = K$.

We proved that finding the optimal allocation of the class of models
CFNE$_{sa}(c), c = O(1)$, is a polynomial task, while the class CFNE$_{aa}(c)$–
{nr, r}, $c = O(1)$, is APX–hard. For CFNE$_{aa}(c)$–r we also proved the
completeness. Moreover, we concluded that CFNE$_{aa}(c)$–tr is in APX.

When the length of the user memory is $K$, we proved that CFNE$_{aa}(K)$–
nr is poly–APX–complete, but the designed approximation algorithm is
not monotone, thus, in order to achieve the DSIC property, we have to
resort to an algorithm providing a weaker guarantee. Finally, we proved
that CFNE$_{aa}(K)$–{r, tr} is APX–complete.

We scratched the surface of the problem, we provided answers to open
questions, but, introducing new models, we also contributed to the generation of new questions. The main question left open by our work on $\text{CFNE}_{aa}(K)$ is to asymptotically close the gap on the approximation ratio of truthful mechanisms for $\text{CFNE}_{aa}(K)$–nr.

Moreover, the approximation algorithms with theoretical guarantee should be compared w.r.t. simpler greedy algorithms, e.g., based on dynamic programming, to evaluate the best approximation ratios that can be obtained in practice. This task is currently under work.

In Chapter 6, we then studied the Mobile Advertising model, focusing on the allocation problem. In addition to the optimal algorithms, we developed polynomial–time approximate algorithms with theoretical bounds and we experimentally evaluated them in terms of the trade–off between suboptimality of the allocation and computational time. In particular, we observed that, in the single–path case, the optimal solution can be found for large instances and that the average–case performance of the approximation algorithm is significantly better than the worst–case theoretical bound. With multi–path cases, finding the optimal allocation quickly becomes intractable, and approximation algorithms are necessary. In future, we will focus on the problems of proving the computational complexity of the problem and designing more efficient approximation algorithms.

We now move to the problem of learning the CTRs of ads in SSAs with truthful mechanisms. While almost all the literature focused on single–slot scenarios, here we focused on multi–slot scenarios. With multiple slots it is necessary to adopt a user model to characterize the valuations of the advertisers over the different slots. We provided results for the separable CTR model, the Cascade Model and even for the new $\text{CFNE}_{sa}(K)$ model. Specifically, in our study we faced different scenarios with different hypothesis on the lack of information and for each scenario we designed a truthful learning mechanism, studied its economic properties, derived an upper bound over the regret, and, for some mechanisms, also a lower bound. In particular, we focused on two different metrics for evaluating the performance of the mechanisms: the regret over the auctioneer’s revenue and the regret over social welfare.

For the separable CTR model, we showed that there exists a DSIC no–regret learning mechanism for the general case in which all the parameters are unknown. Our mechanism presents a regret $\tilde{O}(T^{\frac{2}{3}})$ and it is DSIC in expectation w.r.t. the random component of the mechanism. However, it remains open whether or not it is possible to obtain a regret $\tilde{O}(T^{\frac{1}{2}})$. Studying the Cascade Model we showed that it is possible to design a DSIC mechanism with a regret $\tilde{O}(T^{\frac{2}{3}})$ when only the qualities are unknown. Instead,
even considering the only ad–dependent model when only $\gamma_i$s are unknown, it is not possible to obtain a no–regret DSIC mechanism. The proof of this result would seem to suggest that the same result holds also when truthfulness is in expectation. However, we did not produce any proof for that, leaving it for future works.

Finally, we mention two additional open problems. The first question concerns the study of a lower bound for the case the separable CTR model for different notions of DSIC in expectation, e.g., both in expectation w.r.t. the click realizations and in expectation w.r.t. the random component of the mechanism. Notice that it is an open problem whether the separation of exploration and exploitation phases is necessary and, in the negative case, whether it is possible to obtain a regret $\tilde{O}(T^{1/2})$. It is interesting to analyse the same problems also in the case of the only ad–dependent model.
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Relation between $CTR_{a_i}$ and $\hat{v}_{a_i}$</td>
<td>22</td>
</tr>
<tr>
<td>2.2</td>
<td>Parameters of the counterexample used in the proof of Proposition 1</td>
<td>22</td>
</tr>
<tr>
<td>3.1</td>
<td>Summary of computational results</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Running example: graph (a), paths with starting vertex $t_1$ (b), tree of paths (c)</td>
<td>33</td>
</tr>
<tr>
<td>6.1</td>
<td>Example of the single–path case with allocated ads (a), and of a multi–path case with allocated ads (b)</td>
<td>76</td>
</tr>
<tr>
<td>6.2</td>
<td>Average run time (left) and approximation ratio (right) as $K$ varies. $\lambda = 0.5$ (top) and $\lambda = 0.8$ (bottom)</td>
<td>90</td>
</tr>
<tr>
<td>6.3</td>
<td>Average run time (left) and approximation ratio (right) as $</td>
<td>\Psi_i</td>
</tr>
<tr>
<td>9.1</td>
<td>Position–dependent externalities with unknown $\Lambda_m$, $m \in \mathcal{K}$, Dependency of the relative regret on $T$ and $K$</td>
<td>138</td>
</tr>
<tr>
<td>9.2</td>
<td>Separable CTR model with unknown $\Lambda_m$, $m \in \mathcal{K}$. Dependency of the relative regret on $\mu$. Variance of the revenue of the auctioneer</td>
<td>139</td>
</tr>
<tr>
<td>9.3</td>
<td>Separable CTR model with unknown $q_i$, $i \in \mathcal{N}$, and $\Lambda_m$, $m \in \mathcal{K}$. Dependency of the relative regret on $T$, $N$</td>
<td>140</td>
</tr>
<tr>
<td>9.4</td>
<td>Separable CTR model with unknown $q_i$, $i \in \mathcal{N}$, and $\Lambda_m$, $m \in \mathcal{K}$. Dependency of the relative regret on $K$</td>
<td>140</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>9.5</td>
<td>Dependency on $T$ and $q_{\text{min}}$ in auctions with position/ad–dependent externalities.</td>
<td>141</td>
</tr>
<tr>
<td>9.6</td>
<td>Dependency of the relative regret $\overline{R}_T$ on $N$.</td>
<td>141</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Known results on regret bounds for SSA. We remark with bold font the results provided in this dissertation. . . . . . 38

6.1 Two scenarios used as examples. . . . . . . . . . . . . . . . . . . 77


